Efficient Identification of the Highest Diversity Gain Object

Dimitris Sacharidis[1] and Timos Sellis[2]

[1] Technische Universität Wien, Austria,
`dimitris@ec.tuwien.ac.at`
[2] RMIT University, Australia,
`timos.sellis@rmit.edu.au`

Abstract. Diversification has recently attracted a lot of attention, as a means to retrieve objects that are both relevant to a query and sufficiently dissimilar to each other. Since it is a computationally expensive problem, greedy techniques that iteratively identify the most promising objects are typically used. We focus on the sub-task within one iteration and formalize it as the highest diversity gain problem. We show that it is possible to optimally solve such problems, by appropriately defining a novelty function and identifying the object with the highest novelty. Furthermore, we are able to determine parts of the search space than cannot contain promising objects. Based on these results, we propose a greedy diversification algorithm that iteratively invokes a procedure to determine the most novel object. This procedure uses an index to guide the search towards promising objects, and computes bounds to prune large parts of the space. As a result, the procedure is shown to be I/O optimal, under certain conditions, and experimental studies on real and synthetic data demonstrate its efficiency.

1 Introduction

Conventional information retrieval systems return a set of objects that has the highest relevance to a given user query. However, in various situations, such a result set can be of little help to the user, e.g., when the universe of objects is huge and objects contain overlapping or duplicate information, when the query terms are vague and the actual intent of the user is unknown. To increase the usefulness of the result set, a better approach, termed *diversification*, suggests returning objects that are both *relevant* to the query and *diverse*, i.e., dissimilar to each other.

Although they come in various flavors, relevance and diversity generally pose contradictory objectives. The former favors objects similar to the query, while the latter favors objects dissimilar to each other and thus to the query. Therefore, diversification aims to strike a balance between them. In the most common interpretations, the weighted sum of relevance and diversity is defined to be the target optimization function, and the *diversification problem* is stated as finding a set of objects that maximizes this function.

Diversification is significantly more expensive than top-k ranking, and it is shown to be NP-hard. Therefore, a simple greedy approach is universally adopted in the literature. Starting with an empty result set, iteratively insert in the set the highest diversity gain object, i.e., the one that maximizes the optimization function computed on the set after the insertion. We segregate the sub-task within one iteration of the greedy approach, and refer to it as the *highest diversity gain problem.*

While there has been an abundance of works on diversification problems, surprisingly little effort has been devoted to studying the highest diversity gain problem. For the majority of works, efficiency is not an issue, and the highest gain object is simply identified after retrieving and exhaustively examining all relevant objects. On the other hand, this work attempts to address this overlooking and aims to attract further attention to the highest diversity gain problem, similarly to recent works in top-k diversification [8, 16].

Our work focuses on a broad class of diversification problems, formalizing different definitions of relevance and diversity, computed over sets of attributes, which may or may not overlap. We show that, for this class, it is possible to define a *novelty function* that assigns a score to each object, such that the most novel object is the one with the highest diversity gain. Then, we study this function to gather insight on the possible location of the highest diversity gain object.

Building upon our findings, we propose a simple algorithm, termed DIV, that iteratively invokes the Novelty (NOV) procedure for solving the highest diversity gain problem. NOV utilizes a multidimensional index storing all objects, in order to guide the search towards the most novel object, while avoiding parts of the space that contain unpromising objects. Particularly, NOV computes an upper and lower bound on the novelty of all object within an index node, and makes simple observations to eliminate index subtrees. Therefore, without scanning the entire collection of objects, NOV is able to quickly identify the most novel one. In fact, we show that under certain conditions NOV is *I/O optimal*, i.e., it performs the fewest possible node accesses among any other algorithm that solves the highest diversity gain problem and operates on the same index. Experimental results on real and synthetic data, verify the efficiency of DIV, and show that it is at least an order of magnitude faster than a greedy diversification method [8] and a simple linear scan-based algorithm.

The remainder of this paper is structured as follows. Section 2 reviews related work. Section 3 introduces all concepts and formally states the diversification and highest diversity gain problems. Section 4 presents our approach to the aforementioned problems and details the NOV procedure and the DIV algorithm. Section 5 presents our experimental study, and Section 6 concludes this paper.

2 Related Work

The need for diversification arises in information retrieval (IR) systems, where for example [3] proposes a reranking approach in order to boost the utility of the search results. Their greedy approach assigns a score to each document, termed *maximal marginal relevance* (MMR), which is used to incrementally retrieve documents. MMR is a weighted combination of relevance and diversity and plays the role of a novelty function (similar to Equation 2 with the difference that the first term does not consider the diversity of the set $\mathcal{O}$). The authors, however, do not define a global score for a collection of documents, and do not propose any algorithm for finding the MMR.

Subsequent works address different variations of IR diversification. The work of [1] addresses the ambiguity in a user's query, assuming a taxonomy that models the information within queries and documents is available. The relevance of documents is defined with standard metrics, whereas the diversity is computed using the taxonomy. The proposed greedy algorithm returns a set of documents that cover various topics of interest from the taxonomy. In a similar spirit, [19] proposes a method to diversify query results in online shopping applications. In this scenario, the products are classified, and the goal is to design an efficient method to return diverse products with respect to an ordering among the attributes of the classification. To improve result diversification, [5] defines diversity using multiple criteria: anchor texts, query logs, search result clusters and hosts. Their reranking approach defines a composite similarity function over these diversity criteria. The work of [2] targets performance in diversity-aware search. They avoid reranking all relevant documents, by proposing a data access prioritization scheme, in the spirit of [7], that cleverly alternates among five sequential and random access methods.

A useful review of diversification problems is made in [10], where the authors define eight axioms that any diversification system should be expected to satisfy, and prove that no objective function can satisfy all of them. They also make a categorization of common diversification objectives into three classes, MaxSum, MaxMin and mono-objective. The first two classes are related to dispersion problems [11, 17], which are shown to be NP-hard. On the other hand, the last class is related to MMR and is easy to process in an incremental manner.

Another study focusing on the performance of various diversification methods, in terms of effectiveness and efficiency, appears in [20]. Their problem formulation is similar to ours, in that they define a global score for a set of objects as the weighted sum of its relevance and diversity. They study local search methods that insert/remove objects from the result set, clustering techniques, as well as algorithms introduced in [3] and [10]. They also propose two novel algorithms, a probabilistic approach, and a greedy method that progressively inserts the most promising object to the result set, in a manner similar to our framework. However, unlike this work, all these algorithms must examine the entire dataset to determine at each iteration which object to insert or remove from the result set.

Diversification concepts have also appeared in various other domains; see also the survey of [6]. In the context of recommender systems, the goal is to recommend diverse items to the user. The work of [21] measures the diversity among items based on the dissimilarity of their explanations. An item explanation is the set of similar items that the user has highly rated in the past, or the set of similar users that have highly rated this item. Keyword search in structured databases entails constructing a ranked list of structured queries, representing the possible interpretations of the user's intent. The work of [4] uses a probabilistic ranking model that also takes into account the diversity of the query results. In the context of graph databases, [15] finds the top-k diversified "prestige" nodes in information networks using vertex-reinforced random walks. [9] presents a method for diversifying the results of keyword search in graphs.

The diversification problem is closely related to top-k query processing [12]. The work of [13] introduces the k-nearest diverse neighbor problem, whose goal is to return a set of k objects that are as close as possible to a given query point, and at the same time no two objects have diversity below a given threshold. They propose a greedy algorithm that performs a conventional nearest neighbor (NN) search around the query, and iteratively insert the next nearest object if its distance from the current result set is above the threshold. Since it is a variation of the NN problem, the algorithm avoids visiting the entire dataset. However, theirs is a much easier problem that does not apply to our formulation. A similar problem appears in [14], where a query point is specified in a spatial space, and the goal is to maximize an objective function that tradeoffs relevance and diversity. In this setting, relevance is defined as the distance to the query, whereas diversity is defined either as the smallest distance or the angle similarity to an object in the result. The authors propose a greedy incremental method, which however avoids a complete scan of the dataset only for the angle-based diversity. Another related problem is the spatial cohesion query [18], where the goal is to efficiently find the object that balances the attraction to a set of (point or area) attractors and the repulsion from a set of (point or area) repellers. In essense, that problem is a combination of nearest and farthest neighbor search.

The work in [16] defines the problem of top-k diversification as a variation of conventional top-k, adding the restriction that the result set must not contain objects with similarity less than a user defined threshold. Note that this problem definition is similar to [13], but different than ours, as we do not impose a user defined threshold on the diversity of the objects. Another work [8] is more related to our problem formulation, and we discuss it in more detail next.

The SPP algorithm. The work of [8] assumes objects are embedded in a vector space and solves a problem similar to ours, i.e., their objective function is Equation 1. The diversity is defined as in this work, but the relevance of an object is given by an unknown function, which is not related to the embedded vector space. In contrast, we define the relevance of an object using the distance of an object to a given query.

Similar to our most novel object approach, the authors apply a greedy approach to the diversification problem by identifying in each iteration the most promising object. For this reason, they define a slightly different than ours (Equation 2) novelty function. However, this function does not guarantee that they find the best object in each iteration (i.e., Theorem 1 does not hold).

The proposed algorithm, termed SPP, retrieves objects by sorted access according to (1) their relevance, and (2) their distance to any arbitrary probing location in the vector space. The authors show that best probing locations are the vertices of the bounded Voronoi diagram computed on the diversified set of objects found in the previous iterations.

In each SPP iteration, the most promising object is identified. In particular, SPP retrieves multiple objects via sorted access based on either relevance or distance. Among the objects seen, SPP maintains the one with the highest novelty found, and also computes a local upper bound on the novelty of the unseen objects in each probing location. An iteration of SPP terminates when the maximum of these upper bounds is lower than the highest seen novelty.

3 Definitions

Consider a finite set of objects $\mathcal{U}$, termed the universe. An object $\mathbf{o} \in \mathcal{U}$ is defined over a set of numerical attributes $\mathcal{A} = \mathbb{R}^{|\mathcal{A}|}$. We assume that the set of attributes $\mathcal{A}$ is partitioned into two (not necessarily disjoint) sets, $\mathcal{A}^r$ and $\mathcal{A}^v$, definining the relevance and diversity spaces, respectively. We denote as $\mathbf{o}^r$ (resp. $\mathbf{o}^v$) the projection of the object $\mathbf{o}$ in the relevance (resp. diversity) space.

Given a query $\mathbf{q}$ defined in the relevance space, the *relevance* of an object $\mathbf{o}$ is the opposite of Euclidean distance between the projection of the object in the relevance space and $\mathbf{q}$, i.e., $rel(\mathbf{o}|\mathbf{q}) = -d(\mathbf{o}^r, \mathbf{q}) = -\|\mathbf{o}^r - \mathbf{q}\|_2$. The smaller the distance of an object from $\mathbf{q}$, the greater its relevance.

The *diversity* of two objects $\mathbf{o}_1$, $\mathbf{o}_2$ is the Euclidean distance between their projections in the diversity space, i.e, $div(\mathbf{o}_1, \mathbf{o}_2) = d(\mathbf{o}_1^v, \mathbf{o}_2^v) = \|\mathbf{o}_1^v - \mathbf{o}_2^v\|_2$. The larger the distance between two objects, the greater their diversity.

The aforementioned definitions can be extended to the case of a set of objects. Given a query $\mathbf{q}$, the relevance of a set of objects $\mathcal{O} \subseteq \mathcal{U}$ is defined as:

$$rel(\mathcal{O}|\mathbf{q}) = \sum_{\mathbf{o} \in \mathcal{O}} rel(\mathbf{o}|\mathbf{q}) = -\sum_{\mathbf{o} \in \mathcal{O}} d(\mathbf{o}^r, \mathbf{q}).$$

The diversity of a set of objects $\mathcal{O}$ is defined as:

$$div(\mathcal{O}) = \min_{\mathbf{o}_i \neq \mathbf{o}_j \in \mathcal{O}} div(\mathbf{o}_i, \mathbf{o}_j) = \min_{\mathbf{o}_i \neq \mathbf{o}_j \in \mathcal{O}} d(\mathbf{o}_i^v, \mathbf{o}_j^v).$$

Given a query $\mathbf{q}$, the *score* of a set of objects $\mathcal{O}$ is the weighted sum of the set's diversity and relevance, i.e.,

$$\begin{aligned} s(\mathcal{O}|\mathbf{q}) &= \alpha \cdot div(\mathcal{O}) + \beta \cdot rel(\mathcal{O}|\mathbf{q}) \\ &= \alpha \cdot \min_{\mathbf{o}_i \neq \mathbf{o}_j \in \mathcal{O}} div(\mathbf{o}_i, \mathbf{o}_j) + \beta \cdot \sum_{\mathbf{o} \in \mathcal{O}} rel(\mathbf{o}|\mathbf{q}). \end{aligned} \tag{1}$$

Note that the values α, β should not only reflect the relative weight between diversity and relevance, but also account for normalization (e.g., in this formulation relevance is the sum of $|\mathcal{O}|$ distances, whereas diversity is a single distance). In the remainder of this paper, we set $\alpha = \beta = 1$ to aid readability. All formulas and algorithms can be trivially extended to arbitrary weight values. Therefore, the score of $\mathcal{O}$ is defined as:

$$s(\mathcal{O}|\mathbf{q}) = \min_{\mathbf{o}_i \neq \mathbf{o}_j \in \mathcal{O}} d(\mathbf{o}_i^v, \mathbf{o}_j^v) - \sum_{\mathbf{o} \in \mathcal{O}} d(\mathbf{o}^r, \mathbf{q}).$$

We next formalize the diversification and highest diversity gain problems. Note that given a set of objects $\mathcal{O} \subseteq \mathcal{U}$, the notation $\mathbf{o} \notin \mathcal{O}$ refers to an object $\mathbf{o} \in \mathcal{U} \setminus \mathcal{O}$.

Problem 1. [**k-Diversification**] Find a set $\mathcal{O}_k^* \subseteq \mathcal{U}$ of k objects with the highest score among all other sets of equal size, i.e.,

$$\mathcal{O}_k^* = \operatorname*{argmax}_{\mathcal{O}_k \subseteq \mathcal{U}, |\mathcal{O}_k| = k} s(\mathcal{O}_k|\mathbf{q}).$$

Problem 2. [**Highest Diversity Gain**] Given a query $\mathbf{q}$ and a set of objects $\mathcal{O}$, find an object $\mathbf{o}^* \notin \mathcal{O}$ such that the set $\mathcal{O} \cup \{\mathbf{o}^*\}$ has the greatest score, i.e.,

$$\mathbf{o}^* = \operatorname*{argmax}_{\mathbf{o} \notin \mathcal{O}} s(\mathcal{O} \cup \{\mathbf{o}\}|\mathbf{q}).$$

The object $\mathbf{o}^*$ is called the *highest diversity gain object*, with respect to $\mathcal{O}$ and $\mathbf{q}$. Note that it is possible that more than one objects maximize the score; to simplify presentation, we assume that $\mathbf{o}^*$ is any of them.

A well-known greedy approach, followed by several works (e.g. [3, 10, 8]), for solving the k-diversification problem is to solve k instances of the highest diversity gain problem as follows. Define a sequence of sets of objects $\{\mathcal{O}_i\}$, for $0 \leq i \leq k$. The first term $\mathcal{O}_0$ is the empty set. Then, the i-th term $\mathcal{O}_i$ includes the $(i-1)$-th term $\mathcal{O}_{i-1}$ and the highest diversity gain object with respect to $\mathcal{O}_{i-1}$ and $\mathbf{q}$, i.e., $\mathcal{O}_i = \mathcal{O}_{i-1} \cup \{\mathbf{o}_i^*\}$.

4 Methodology

We present our methodology for solving the highest gain and diversification problem. Section 4.1 introduces the novelty function and Section 4.2 presents important observations for eliminating unpromising objects. Then, Sections 4.3 and 4.4 describe the algorithms for solving the two problems. Section 4.5 discusses the case of nonidentical relevance and diversity spaces. Table 1 gathers the most important symbols used throughout this paper.

4.1 The Novelty Function

Given a query $\mathbf{q}$, a set of objects $\mathcal{O}$, we define the *novelty* of an object $\mathbf{o} \notin \mathcal{O}$ as:

$$n(\mathbf{o}|\mathcal{O}, \mathbf{q}) = \min\{div(\mathcal{O}), \min_{\mathbf{o}' \in \mathcal{O}} d(\mathbf{o}^v, \mathbf{o}'^v)\} + rel(\mathbf{o}|\mathbf{q}). \tag{2}$$

The following theorem shows the importance of the novelty function. It implies that to solve the highest gain problem, it suffices to find the *most novel object*, i.e., the one with the largest novelty.

Table 1. Notation

Symbol	Definition
$\mathcal{U}$	universe of objects
$\mathcal{A}$, $\mathcal{A}^r$, $\mathcal{A}^v$	set of all, relevance, diversity attributes
$\mathbf{o}$	an object
$\mathbf{q}$	the query
$d(\mathbf{x}, \mathbf{y})$	Euclidean distance ($\lVert\mathbf{x} - \mathbf{y}\rVert_2$)
$rel(\mathbf{o}\vert\mathbf{q})$	relevance of $\mathbf{o}$ w.r.t. $\mathbf{q}$
$div(\mathbf{o}_1, \mathbf{o}_2)$	diversity of $\mathbf{o}_1$ and $\mathbf{o}_2$
$\mathcal{O}$	a set of objects
$rel(\mathcal{O}\vert\mathbf{q})$	relevance of $\mathcal{O}$ w.r.t. $\mathbf{q}$
$div(\mathcal{O})$	diversity of $\mathcal{O}$
$s(\mathcal{O}\vert\mathbf{q})$	score of $\mathcal{O}$ w.r.t. $\mathbf{q}$
$n(\mathbf{o}\vert\mathcal{O}, \mathbf{q})$	novelty of $\mathbf{o}$ w.r.t. $\mathcal{O}$ and $\mathbf{q}$
δ	diversity of $\mathcal{O}$ w.r.t. $\mathbf{q}$
τ	a novelty value
$\mathbf{o}_{NN}$	nearest neighbor of $\mathbf{o} \notin \mathcal{O}$ in $\mathcal{O}$
T, N	R*-Tree indexing the universe, a node of T
$n^+(N)$, $n^-(N)$	upper, lower bound of novelty of objects in N

Theorem 1. *The most novel object is the highest gain object, i.e., for any object $\mathbf{o} \notin \mathcal{O}$ the following holds $\mathbf{o}^* = \operatorname{argmax} n(o|\mathcal{O}, \mathbf{q}) = \operatorname{argmax} s(\mathcal{O} \cup \{\mathbf{o}\}|\mathbf{q})$.*

Proof. We prove by contradiction. Let $\mathbf{o}^n \neq \mathbf{o}^*$ be the object that has the largest novelty, so that $n(o^*|\mathcal{O}, \mathbf{q}) < n(o^n|\mathcal{O}, \mathbf{q})$.

Consider the score of the set $\mathcal{O} \cup \{\mathbf{o}^*\}$ and observe that:

$$\begin{aligned} s(\mathcal{O} \cup \{\mathbf{o}^*\}) =& \min\left\{ \min_{\mathbf{o}_i \neq \mathbf{o}_j \in \mathcal{O}} d(\mathbf{o}_i^v, \mathbf{o}_j^v), \min_{\mathbf{o}' \in \mathcal{O}} d(\mathbf{o}^{*v}, \mathbf{o}'^v) \right\} \\ & - \left(\sum_{\mathbf{o} \in \mathcal{O}} d(\mathbf{o}^r, \mathbf{q}) + d(\mathbf{o}^{*r}, \mathbf{q}) \right) \\ =& n(\mathbf{o}^*|\mathcal{O}, \mathbf{q}) - \sum_{\mathbf{o} \in \mathcal{O}} d(\mathbf{o}^r, \mathbf{q}). \end{aligned}$$

Similarly, when the object $\mathbf{o}^n$ is included:

$$s(\mathcal{O} \cup \{\mathbf{o}^n\}) = n(\mathbf{o}^n|\mathcal{O}, \mathbf{q}) - \sum_{\mathbf{o} \in \mathcal{O}} d(\mathbf{o}^r, \mathbf{q}).$$

Since $n(\mathbf{o}^*|\mathcal{O}, \mathbf{q}) < n(\mathbf{o}^n|\mathcal{O}, \mathbf{q})$, we obtain that $s(\mathcal{O} \cup \{\mathbf{o}^*\}) < s(\mathcal{O} \cup \{\mathbf{o}^n\})$, which is a contradiction as $\mathbf{o}^*$ maximizes the score. □

4.2 Observations

Over the next sections, we assume that diversity and relevance are defined over the same space, i.e., $\mathcal{A}^r = \mathcal{A}^v = \mathcal{A}$. Therefore, we drop all r and v superscripts.

Later, in Section 4.5, we lift this restriction. Also note that for illustration purposes, all examples assume that $\mathcal{A} = \mathbb{R}^2$, i.e., the Euclidean plane.

Moreover, we simplify notation by introducing the following concepts. We denote by δ the diversity of the set $\mathcal{O}$, i.e., $\delta = div(\mathcal{O})$. Given an object $\mathbf{o}$, we define $\mathbf{o}_{NN}$ to be its nearest neighbor within $\mathcal{O}$, i.e., $\mathbf{o}_{NN} = \operatorname{argmin}_{\mathbf{o}' \in \mathcal{O}} d(\mathbf{o}, \mathbf{o}')$. Since the set $\mathcal{O}$ and query $\mathbf{q}$ are fixed, we drop the $\mathcal{O}, \mathbf{q}$ designation for the novelty of an object $\mathbf{o}$, and thus Equation 2 simplifies to:

$$n(\mathbf{o}) = \min\{\delta, d(\mathbf{o}, \mathbf{o}_{NN})\} - d(\mathbf{o}, \mathbf{q}). \tag{3}$$

We present two observations for eliminating objects that have novelty less than a known value.

Observation 1. *Given a set of objects $\mathcal{O}$, a query $\mathbf{q}$, and a known novelty value τ, any object $\mathbf{o} \notin \mathcal{O}$ such that $d(\mathbf{o}, \mathbf{q}) > \delta - \tau$ has novelty less than τ.*

Proof. The novelty of an object $\mathbf{o}$ satisfying this criterion is $n(\mathbf{o}) = \min\{\delta, d(\mathbf{o}, \mathbf{o}_{NN})\} - d(\mathbf{o}, \mathbf{q}) < \min\{\delta, d(\mathbf{o}, \mathbf{o}_{NN})\} + \tau - \delta$. Since $\delta \geq \min\{\delta, d(\mathbf{o}, \mathbf{o}_{NN})\}$, we obtain $n(\mathbf{o}) < \tau$. $\square$

Observation 2. *Given a set of objects $\mathcal{O}$, a query $\mathbf{q}$, an object $\mathbf{o}' \in \mathcal{O}$ and a novelty value τ, any object $\mathbf{o} \notin \mathcal{O}$ such that $d(\mathbf{o}, \mathbf{o}') - d(\mathbf{o}, \mathbf{q}) < \tau$ has novelty less than τ.*

Proof. For any object $\mathbf{o} \notin \mathcal{O}$ and its nearest neighbor $\mathbf{o}_{NN}$ within $\mathcal{O}$, it holds that $d(\mathbf{o}, \mathbf{o}') \geq d(\mathbf{o}, \mathbf{o}_{NN})$. Moreover, since $d(\mathbf{o}, \mathbf{o}_{NN}) \geq \min\{\delta, d(\mathbf{o}, \mathbf{o}_{NN})\}$, the novelty of object $\mathbf{o}$ is $n(\mathbf{o}) \leq d(\mathbf{o}, \mathbf{o}') - d(\mathbf{o}, \mathbf{q})$. Therefore, any object $\mathbf{o}$ satisfying the criterion must have $n(\mathbf{o}) < \tau$. $\square$

Note that Observation 2 holds, and is in fact stronger, when we substitute $\mathbf{o}'$ with $\mathbf{o}$'s nearest neighbor $\mathbf{o}_{NN}$ within $\mathcal{O}$.

4.3 Finding the Most Novel Object

This section introduces the *Novelty* (NOV) procedure for solving the highest gain problem. The key idea is to use an index in order to direct the search towards the most promising object while pruning groups of unpromising objects.

We build an R*-Tree T on the Euclidean space $\mathcal{A}$ indexing the universe of objects $\mathcal{U}$. Each node N corresponds to a disk page, is associated with a rectangle $N.mbr$, and contains a number of child nodes. A leaf node N represents an object $\mathbf{o}$, and thus its rectangle is the point in $\mathcal{A}$ corresponding to this object. The rectangle of an internal node N is the minimum bounding rectangle (MBR) of (i.e., the smallest rectangle that encloses) all children rectangles.

We say that an object $\mathbf{o}$ is in a node N, denoted as $\mathbf{o} \in N$, if $\mathbf{o}$ is represented by a leaf node in the subtree rooted at N. Furthermore, given a point $\mathbf{x}$, we define $\texttt{mindist}(N, \mathbf{x})$ (resp. $\texttt{maxdist}(N, \mathbf{x})$) to be the smallest (resp. largest) possible distance to $\mathbf{x}$ of any point within the MBR of N.

In the following, we present a set of observations regarding the novelty of objects in a given non-leaf node.

Novelty bounds. Given a non-leaf node N, but not the objects within its subtree, it is possible to compute bounds on the novelty of any object in N.

Lemma 1. *Given a set of objects $\mathcal{O}$ and a non-leaf node N, the novelty of an object $\mathbf{o}$ in N cannot be more than $n^+(N) = \delta - \texttt{mindist}(N, \mathbf{q})$ if $|\mathcal{O}| > 1$, and cannot be more than $n^+(N) = \texttt{maxdist}(N, \mathbf{o}') - \texttt{mindist}(N, \mathbf{q})$ if $\mathcal{O} = \{\mathbf{o}'\}$.*

Proof. Assume $|\mathcal{O}| > 1$, when δ is defined. Clearly, for any object $\mathbf{o} \in N$ it holds that $\delta \geq \min\{\delta, d(\mathbf{o}, \mathbf{o}_{NN})\}$. By the definition of $\texttt{mindist}$ and since any object $\mathbf{o} \in N$ lies within $N.mbr$, we have $\texttt{mindist}(N, \mathbf{q}) \leq d(\mathbf{o}, \mathbf{q})$. Combining the two inequalities, we derive $n^+(N) \geq n(\mathbf{o})$ for any object $\mathbf{o} \in N$ and $|\mathcal{O}| > 1$.

Assume $\mathcal{O} = \{\mathbf{o}'\}$. By the definition of $\texttt{maxdist}$ and since any object $\mathbf{o} \in N$ is within $N.mbr$, we have $\texttt{maxdist}(N, \mathbf{o}') \leq d(\mathbf{o}, \mathbf{o}')$. Also, we have $\texttt{mindist}(N, \mathbf{q}) \leq d(\mathbf{o}, q)$. Combining the two inequalities, we derive $n^+(N) \geq n(\mathbf{o})$ for any object $\mathbf{o} \in N$ and $\mathcal{O} = \{\mathbf{o}'\}$. $\square$

Lemma 1 implies that the non-leaf node with the highest $n^+()$ value is more likely to contain the most novel object, and thus provides the means to direct the search.

Lemma 2. *Given a set of objects $\mathcal{O}$ and a non-leaf node N, the novelty of an object $\mathbf{o}$ in N cannot be less than $n^-(N) = -\texttt{maxdist}(N, \mathbf{q})$.*

Proof. For any object $\mathbf{o} \in N$ it holds that $\min\{\delta, d(\mathbf{o}, \mathbf{o}_{NN})\} \geq 0$ and $d(\mathbf{o}, \mathbf{q}) \leq \texttt{maxdist}(N, \mathbf{q})$. Therefore, its novelty is $n(\mathbf{o}) \geq -\texttt{maxdist}(N, \mathbf{q})$. $\square$

Lemma 2 provides a lower bound on the novelty of the most novel object: $n(\mathbf{o}^*) \geq n^-(N)$ for any non-leaf node N.

Applying Observation 1. The following lemma applies Observation 1 for a node N.

Lemma 3. *Given a novelty value τ, a node N contains objects with novelty less than τ, if $\texttt{mindist}(N, \mathbf{q}) > \delta - \tau$.*

Proof. Any object $\mathbf{o} \in N$ has $d(\mathbf{o}, \mathbf{q}) \geq \texttt{mindist}(N, \mathbf{q})$. From the condition of the lemma, we obtain $d(\mathbf{o}, \mathbf{q}) > \delta - \tau$. Thus, Observation 1 applies for all objects $\mathbf{o} \in N$. $\square$

Applying Observation 2. The following lemmas apply Observation 2 for a node N.

Lemma 4. *Given a novelty value τ, a node N contains objects with novelty less than τ, if there exists an object $\mathbf{o}' \in \mathcal{O}$ such that $\texttt{maxdist}(N, \mathbf{o}') - \texttt{mindist}(N, \mathbf{q}) < \tau$.*

Proof. For any object $\mathbf{o} \in N$ the following conditions hold: $d(\mathbf{o}, \mathbf{o}') \leq \texttt{maxdist}(N, \mathbf{o}')$ and $d(\mathbf{o}, \mathbf{q}) \geq \texttt{mindist}(N, \mathbf{q})$. It also holds that $d(\mathbf{o}, \mathbf{o}') - d(\mathbf{o}, \mathbf{q}) < \tau$ and thus Observation 2 applies for all objects within N. $\square$

Algorithm description. Algorithm 1 presents the pseudocode of the NOV procedure. NOV takes as input the R*-Tree T indexing all objects in the universe, the query q, and the set of objects $\mathcal{O}$, and returns the most novel object $\mathbf{o}^*$ with respect to $\mathcal{O}$ and $\mathbf{q}$.

NOV maintains a novelty value τ, initially set to $-\infty$, which corresponds to a lower bound of the highest possible novelty. It also computes the diversity of $\mathcal{O}$ and initializes H (line 1). NOV directs the search using the heap H, which contains nodes sorted descending on their novelty upper bound (Lemma 1).

NOV performs a number of iterations (lines 3–15), where at the end of each iteration the node N_x at the top of the heap is popped (line 15); for the first iteration N_x is the root node of T (line 2). NOV terminates when node N_x is a leaf, in which case the object corresponding to this node is the most novel object $\mathbf{o}^*$ (line 16).

Assuming that N_x is not a leaf, NOV reads this node from disk (line 4) and checks if Lemmas 3, 4 apply for its children (lines 5–14). Particularly, it first checks if Lemma 3 applies for a child N (lines 7–8). If not, NOV examines all objects within the set $\mathcal{O}$ (lines 9–11). For each such object $\mathbf{o}'$, the algorithm checks if Lemma 4 (lines 10–11) applies for node N. If neither lemma applies (lines 12–14), then N is pushed in the heap (line 14), and the novelty value is appropriately updated (line 13) according to Lemma 2.

Correctness and optimality. The next theorems prove the correctness and I/O optimality of NOV.

Theorem 2. *The NOV procedure returns the most novel object.*

Proof. We show that NOV cannot miss the most novel object $\mathbf{o}^*$. NOV prunes nodes of the R*-Tree based on Lemmas 2–4. Therefore, by the correctness of these lemmas, $\mathbf{o}^*$ cannot be in any pruned node.

NOV terminates when it pops from the heap a leaf corresponding to object $\mathbf{o}_x$. Since the heap contains nodes sorted by the upper bound of Lemma 1, it holds that $n(\mathbf{o}_x) \geq n^+(N)$ for all nodes $N \in H$. By the correctness of the lemma, $\mathbf{o}_x$ has higher novelty than any object within any of the nodes in the heap. $\square$

Intuitively, I/O optimality means that an R*-Tree-based algorithm only visits nodes that *may contain* the most novel object. To formalize this, we introduce the notion of the search frontier.

Given the most novel object $\mathbf{o}^*$, define the *search frontier* (SF) to be the part of the space that contains points with novelty more than $n(\mathbf{o}^*)$. Since $\mathbf{o}^*$ has the highest possible novelty among objects in $\mathcal{U} \setminus \mathcal{O}$, the SF contains no object. However, it holds that any R*-Tree-based algorithm, which finds the most novel object, must access all nodes that intersect the SF. This is true even for a node

Algorithm 1: NOV

```
Input: R*-Tree $T$; objects $\mathcal{O}$; query $\mathbf{q}$
Output: $\mathbf{o}^*$ the most novel object
Variables: $H$ a heap with nodes sorted by $n^+()$; novelty value $\tau$
1  $\tau \leftarrow -\infty$; $\delta \leftarrow div(\mathcal{O})$; $H \leftarrow \varnothing$
2  $N_x \leftarrow N_{root}$                                  ▷ root node of $T$
3  while $N_x$ is non-leaf do
4      read node $N_x$
5      foreach child $N$ of $N_x$ do
6          $pruned \leftarrow false$
7          if $\texttt{mindist}(N, \mathbf{q}) > \delta - \tau$ then          ▷ Lemma 3
8              $pruned \leftarrow true$
9          foreach $\mathbf{o}' \in \mathcal{O}$ do
10             if $\texttt{maxdist}(N, \mathbf{o}') - \texttt{mindist}(N, \mathbf{q}) < \tau$ then   ▷ Lemma 4
11                 $pruned \leftarrow true$
12         if not pruned then
13             if $n^-(N) > \tau$ then $\tau \leftarrow n^-(N)$         ▷ Lemma 2
14             push$(H, N)$
15     $N_x \leftarrow$ pop$(H)$
16 $\mathbf{o}^* \leftarrow N_x.mbr$
```

N that does not contain $\mathbf{o}^*$; the reason is that the algorithm cannot determine this containment unless it retrieves the contents of N.

Therefore, I/O optimality means that an algorithm only accesses nodes that intersect with the SF, and never visits the same node twice.

To prove I/O optimality for NOV, we must ensure that the upper bound on the novelty of a node is *tight.* In other words, there must exist a point within the node's MBB (not necessarily an object in $\mathcal{U}$) such that its novelty is equal to the upper bound. The upper bound of Lemma 1 is not tight. However, given a node it is possible to compute a tight upper bound in an analytical way. The key idea is to find a point within the node's MBB that maximizes the novelty function. This point must be one of the critical points where all partial derivatives of the novelty function become zero.

Theorem 3. *The NOV procedure with a tight upper bound on novelty is I/O optimal.*

Proof. First, NOV never visits the same node twice. The reason is that a node is only inserted once in the heap, and NOV always removes from the heap the node it visits.

We next prove that NOV with a tight upper bound on novelty $n^+()$ never accesses a node that does not intersect with the SF, defined by the most novel object $\mathbf{o}^*$. Assume the contrary, i.e., that NOV visits a node N that does not intersect with the SF. Since N is outside the SF, for any point (not necessarily

Algorithm 2: DIV

```
    Input: R*-Tree $T$; query $\mathbf{q}$
    Output: $\mathcal{O}$ the $k$-diversified set of objects
    Variables: $\mathbf{o}^*$ current most novel object
1   $\mathcal{O} \leftarrow \varnothing$
2   for $i \leftarrow 1$ to $k$ do
3       $\mathbf{o}^* \leftarrow \text{NOV}(T, \mathcal{O}, \mathbf{q})$
4       $\mathcal{O} \leftarrow \mathcal{O} \cup \{\mathbf{o}^*\}$
```

an object in $\mathcal{U}$) $\mathbf{p} \in N$ it holds that $n(\mathbf{p}) < n(\mathbf{o}^*)$. By the property of the tight upper bound, there exists a point $\mathbf{p}' \in N$ such that $n(\mathbf{p}') = n^+(N)$. Moreover, since N is visited it must hold that $n^+(N) > n(\mathbf{o}^*)$. Therefore, $n(\mathbf{p}') > n(\mathbf{o}^*)$, which contradicts the fact that N is outside the SF. □

4.4 Solving the Diversification Problem

We next present a simple algorithm, termed DIV, that solves the k-diversification problem by invoking k times the NOV procedure. Algorithm 2 shows the pseudocode of DIV. The algorithm takes as input the R*-Tree T indexing all objects in the universe, the query $\mathbf{q}$, and returns the k-diversified set of objects $\mathcal{O}$.

DIV initializes the set of diversified objects $\mathcal{O}$ to be empty (line 1). Then it calls k times the NOV procedure (lines 2–4). In each invocation, DIV obtains the next most novel object $\mathbf{o}^*$ (line 3) and inserts it into the set $\mathcal{O}$ (line 4).

4.5 Generalization for Nonidentical Relevance and Diversity Spaces

So far, we have assumed that relevance and diversity are defined over the same set of attributes. This section discusses the general case when $\mathcal{A}^r \subseteq \mathcal{A}$ and $\mathcal{A}^v \subseteq \mathcal{A}$. Note that the query $\mathbf{q}$ takes values only for the relevance attributes $\mathcal{A}^r$, while an object $\mathbf{o}$ takes values on all attributes. To illustrate the concepts, we assume a three-dimensional example where $\mathcal{A}^r = \{A_1, A_2\}$ and $\mathcal{A}^v = \{A_2, A_3\}$.

We define the *extension* of the query $\mathbf{q}$, denoted by $\rho(\mathbf{q})$, as the set of points in $\mathcal{A}$ such that $\rho(\mathbf{q}).A_x = \mathbf{q}.A_x$, for all $A_x \in \mathcal{A}^r$, and $\rho(\mathbf{q}).A_y = [-\infty, +\infty]$, for all $A_y \notin \mathcal{A}^r$. The extension has the property that for any object $\mathbf{o} \in \mathcal{U}$, it holds that $rel(\mathbf{o}|\mathbf{q}) = -d(\mathbf{o}^r, \mathbf{q}) = -d(\mathbf{o}, \rho(\mathbf{q}))$, i.e., the relevance of $\mathbf{o}$ is given by the distance between $\rho(\mathbf{q})$ and $\mathbf{o}$ computed *over all attributes*. In our three-dimensional example, $\rho(\mathbf{q})$ represents a line perpendicular to the A_1, A_2 plane, and the relevance of $\mathbf{o}$ is its (perpendicular) distance to this line.

Similarly, we define the *restriction* of an object $\mathbf{o}$, denoted by $\pi(\mathbf{o})$, as the set of points in $\mathcal{A}$ such that $\pi(\mathbf{o}).A_x = \mathbf{o}.A_x$, for all $A_x \in \mathcal{A}^v$, and $\pi(\mathbf{o}).A_y = [-\infty, +\infty]$, for all $A_y \notin \mathcal{A}^v$. The extension has the property that for any other object $\mathbf{o}' \in \mathcal{U}$, it holds that $div(\mathbf{o}', \mathbf{o}) = d(\mathbf{o}'^v, \mathbf{o}^v) = d(\mathbf{o}', \pi(\mathbf{o}))$, i.e., the diversity of $\mathbf{o}'$ and $\mathbf{o}$ is given by the distance between $\mathbf{o}'$ and the restriction of $\mathbf{o}$ computed *over all attributes*. In our three-dimensional example, $\pi(\mathbf{o})$ represents a line perpendicular to the A_2, A_3 plane, and the diversity of $\mathbf{o}$ and another object $\mathbf{o}'$ is the (perpendicular) distance of $\mathbf{o}'$ to this line.

Using the previous transformations and in analogy to Equation 3, the novelty of an object $\mathbf{o}$ becomes

$$n(\mathbf{o}) = \min\{\delta, d(\mathbf{o}, \pi(\mathbf{o}_{NN}))\} - d(\mathbf{o}, \rho(\mathbf{q})).$$

More importantly, the two main results of Section 4.2 hold, if we substitute $\mathbf{q}$ with $\rho(\mathbf{q})$ and $\mathbf{o}'$ with $\pi(\mathbf{o}')$. Given the aforementioned transformations, adapting NOV is straightforward. Observe that all Lemmas 1–4 hold, as long as we substitute $\mathbf{q}$ with $\rho(\mathbf{q})$ and $\mathbf{o}'$ with $\pi(\mathbf{o}')$.

5 Experimental Evaluation

Section 5.1 describes the setting, while Section 5.2 contains the results.

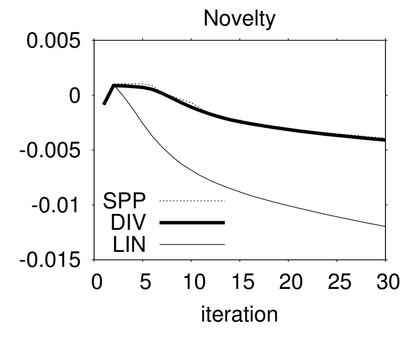
5.1 Setting

Methods. We implement our proposed diversification method DIV, which is based on the NOV procedure. We also implement the SPP algorithm from [8], and adapt it to consider at each iteration the same novelty function with DIV. Recall that SPP requires a module that provides sorted access on the universe of objects based on their distance from a given location. In our implementation, this module performs nearest neighbor search using the same R*-Tree as in DIV. Furthermore, as a baseline, we implement a greedy algorithm denoted as LIN. In each iteration, LIN performs an exhaustive linear scan over the universe of objects to identify the most novel one, and inserts it into the result set. All algorithms are implemented in C++ and executed on a 2GHz machine, whose disk page size is 4096 bytes.

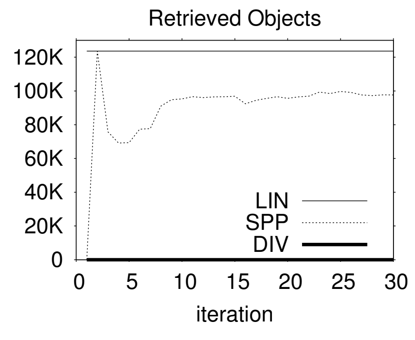
Datasets. We use a real and two synthetic datasets in our evaluation. The real dataset[3], denoted as NE, is a two-dimensional collection of 125,000 postal addresses in three metropolitan areas, New York, Philadelphia and Boston. The synthetic dataset UNI contains independent and uniformly distributed random objects. The synthetic dataset CLU contains objects that are randomly distributed around 1,000 cluster centers. The probability with which a cluster center attracts objects is drawn from a Zipfian distribution with degree 0.8.

Parameters and metrics. We study the effect on the performance of the algorithms of three parameters: number of objects in the universe, number of attributes and value of k. Particularly, for the NE dataset, the number of objects is fixed to $|\mathcal{U}| = 125\text{K}$ and the dimensionality to $|\mathcal{A}| = 2$. Table 2 shows the range of values examined for each parameter. In each experiment, we vary a single parameter and set the others to their default values.

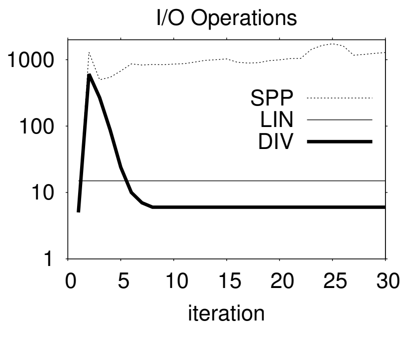
[3] Available at `http://www.rtreeportal.org`



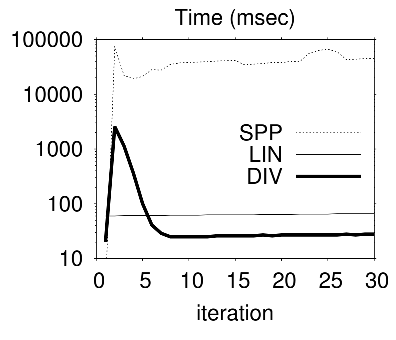
(a) Highest novelty



(b) Number of retrieved objects



(c) I/O cost



(d) Processing time

Fig. 1. NE, per iteration analysis

To quantify performance, we measure the number of retrieved objects, the number of I/O operations, and the total processing time. The reported values are the averages of 10 distinct executions with queries uniformly selected at random from the space. We emphasize that the objective of this work is to improve the efficiency of solving the diversification problem, and not on the quality of the constructed result set. In fact, all implemented methods follow the same iterative heuristic and thus construct the same result set (save for differences due to ties) with identical global scores.

Table 2. Parameters

Symbol	**Values**	**Default**
$\lvert\mathcal{U}\rvert$	100K, 500K, 1M, 5M, 10M	1M
$\lvert\mathcal{A}\rvert$	2, 3, 4, 5, 6	2
k	[1, 50]	20

5.2 Results

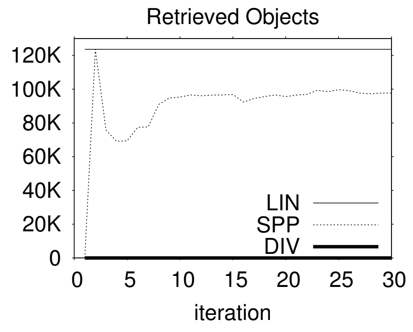
Per iteration analysis. In the first experiment, we study the performance of all algorithms as they iteratively find promising objects. For this setting, we use the NE dataset and set $k = 30$. Figure 1 contains the results, where the bold, regular, dotted line corresponds to DIV, LIN, SPP, respectively.

Figure 1a depicts the novelty value of the most novel object found at each iteration. Observe that at the first iteration, all methods select the nearest neighbor to the query whose novelty is equal to minus the distance to the query. At the second iteration, all methods select objects that have the same novelty. However, these two initial objects may differ among methods, depending on how they solve ties in novelty. In fact after this point, at each iteration, the selection of the object among those with equal novelty may affects the best novelty values in subsequent iterations. This phenomenon is observed in Figure 1a, where the novelty graphs vary slightly between DIV and SPP, and a bit more with respect to LIN. The fact that LIN's highest novelty is significantly lower is attributed to unfortunate choices when breaking ties. Note that the highest novelty of all methods decreases because the relevance term of the novelty function dominates the diversity one, while objects farther from the query are inserted.

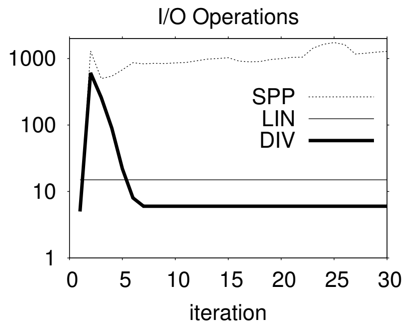
Figure 1b shows the number of objects retrieved at each iteration. LIN needs to scan the entire dataset at each iteration, hence it performs the maximum number of retrievals (125K). On the other hand, DIV always guides search towards the most novel object, and thus performs a single retrieval at each iteration. SPP performs a single retrieval in the first iteration to identify the nearest to the query object. However, at all other iterations, SPP must examine a very large set of objects. Even though SPP prunes the search space, it cannot guide the search towards the non-pruned space. The main reason is that the set of probing locations remains fixed throughout an iteration.

Figure 1c shows the number of I/O operations performed at each iteration. LIN retrieves constantly 16 pages from disk. DIV performs 5 I/Os to retrieve the first object and constantly 6 I/Os to retrieve objects after the seventh iteration. For iterations 2 and 3 DIV requires more than 100 I/Os to retrieve the most novel object. The reason is that, in the beginning, the set of retrieved objects is too small to prune large parts of the space. Regarding SPP, note that except in the first iteration, it performs I/Os in the order of 1,000. There are two reasons for this. SPP needs to retrieve a large number of objects (although less than LIN), and each retrieval may cost up to 5 I/Os due to the nearest neighbor search. To make matters worse, since SPP initiates multiple NN searches at different probing locations, it happens that the same object is retrieved at more than one NN search. This last reason explains why the number of I/Os increases at each iteration: the number of probing locations also increases.

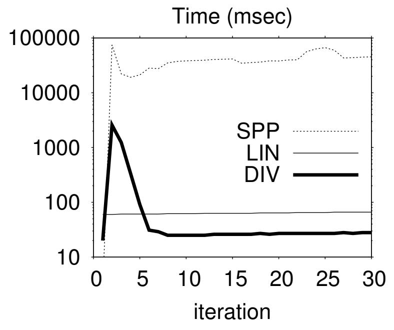
Figure 1d shows the processing time per iteration. LIN require constant time, around 60msec, per iteration. The same holds for DIV after the seventh iteration, as it requires around 25msec, per iteration. On the other hand SPP requires more than 10 seconds for each iteration after the first.



(a) Number of retrieved objects



(b) I/O cost



(c) Processing time

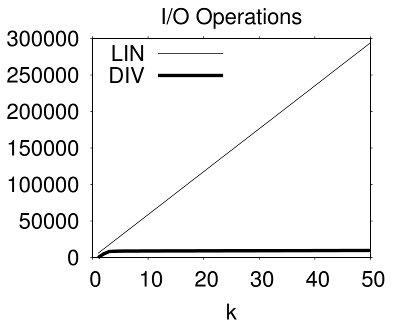
Fig. 2. NE, per iteration analysis, fixed $\mathcal{O}$

Per iteration analysis with fixed $\mathcal{O}$. We repeat the first experiment, but at this time we force DIV and LIN to start each iteration with the same set of objects as SPP. In other words, for fairness at each iteration, all algorithms solve an identical instance of the highest diversity gain problem. The lines for LIN and SPP are not affected in this experiment. The number of I/O operations and the per iteration processing time of DIV are largely unaffected by the enforcement of common $\mathcal{O}$ accross methods.

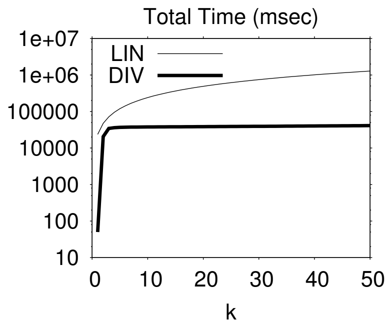
In the remainder of the experimental evaluation, we exclude the SPP algorithm due to its high processing cost.

Effect of k. In the next experiment, we vary the k value from 1 up to 50 in increments of one, while we set the number of objects fixed to $|\mathcal{U}| = 1\text{M}$. Figure 3 plots the results as a function of k. Note that in contrast to the previous experiments, the reported times are at the end of the algorithms execution and not per iteration.

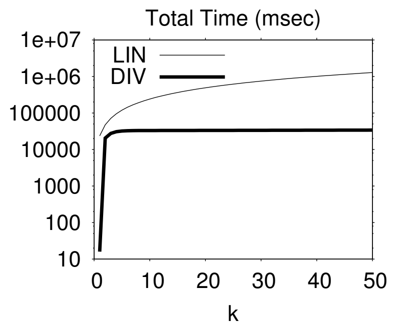
Figure 3a shows the I/O cost of LIN and DIV for the CLU dataset. Observe that the I/O cost of LIN grows linearly with k, since the cost per iteration is constant (see the discussion in the context of Figure 1). Note that the I/O cost of DIV for $k = 1$ is minimal, and that it slightly grows with k, so little that the



(a) CLU



(b) CLU



(c) UNI

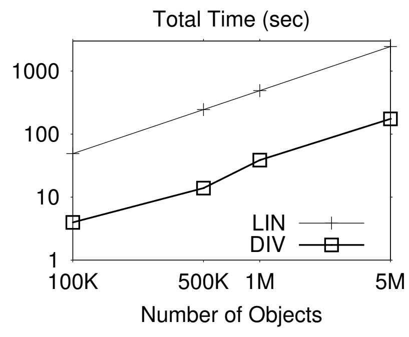
Fig. 3. Effect of k

trend is not apparent in the figure. This is in accordance to our per iteration analysis, where DIV's cost for retrieving additional objects is minor.

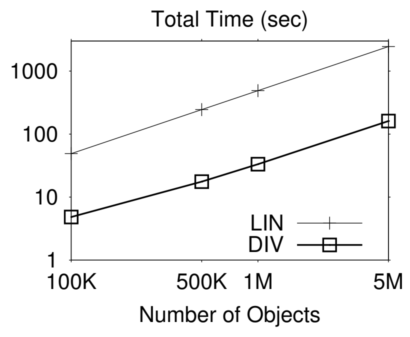
Figure 3b depicts the total processing time for the CLU dataset. Note that processing time is shown on a logarithmic axis. The trends that appear in the previous figure, also show here. DIV is already more than an order of magnitude faster than LIN for $k = 20$ and the gap increases with k. Figure 3c presents the effect of k on processing time for the UNI dataset. The trends are identical to those for the CLU dataset.

Varing $|\mathcal{U}|$. In this experiment, we vary the number of object in the universe $|\mathcal{U}|$ from 100K up to 10M, while we set $k = 20$. Figure 4 depicts the total processing time as a function of $|\mathcal{U}|$.

Figure 4a shows the processing time of DIV and LIN on the CLU dataset. As the number of objects increases, it becomes harder to identify the k object with the highest novelty. Hence the performance of both methods decreases with $|\mathcal{U}|$. Observe, however that the benefit of DIV over LIN remains constant at more than one order of magnitude. Similar results hold for the UNI dataset, as shown in Figure 4b.

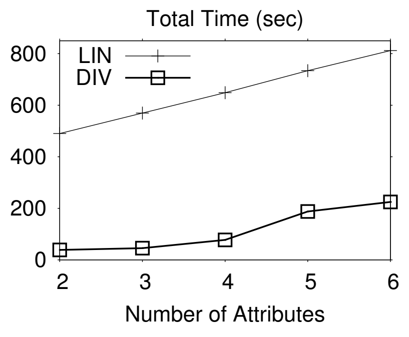


(a) CLU

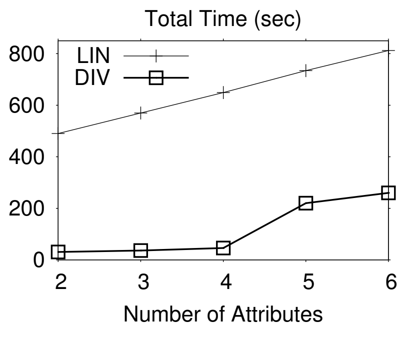


(b) UNI

Fig. 4. Effect of $|\mathcal{U}|$



(a) CLU



(b) UNI

Fig. 5. Effect of $|\mathcal{A}|$

Varing $|\mathcal{A}|$. Finally, we increase the number of attributes $|\mathcal{A}|$, i.e., dimensionality, from 2 up to 6. Figure 5 depicts the total processing time as a function of $|\mathcal{A}|$. Figure 5a shows that the performance of both methods deteriorates as the dimensionality of the CLU dataset increases. Note that the relative benefit of DIV against LIN takes its highest value at 4 dimensions. The results are similar for the UNI dataset, as shown in Figure 5b.

6 Conclusions

The diversification problem is to retrieve a set of objects such that their relevance, measured by distance, to a given query and their diversity, measured by pairwise distance, is maximized. Since it is a computationally hard problem, greedy approaches are typically used. This work introduces the highest diversity gain problem, which is integral in any greedy solution of the diversification problem. We show that for many diversification problems it is possible to define a novelty function that assigns score to objects, so that the most novel object optimally solves the highest diversity gain problem. Based on the study of the novelty function, we proposes an index-based algorithm that is I/O optimal. Experiments have shown that our approach is at least an order of magnitude faster than a recent greedy diversification method, and a simple linear scan.

References

1. Agrawal, R., Gollapudi, S., Halverson, A., Ieong, S.: Diversifying search results. In: WSDM. pp. 5–14 (2009)
2. Angel, A., Koudas, N.: Efficient diversity-aware search. In: SIGMOD. pp. 781–792 (2011)
3. Carbonell, J.G., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: SIGIR. pp. 335–336 (1998)
4. Demidova, E., Fankhauser, P., Zhou, X., Nejdl, W.: *DivQ*: diversification for keyword search over structured databases. In: SIGIR. pp. 331–338 (2010)
5. Dou, Z., Hu, S., Chen, K., Song, R., Wen, J.R.: Multi-dimensional search result diversification. In: WSDM. pp. 475–484 (2011)
6. Drosou, M., Pitoura, E.: Search result diversification. SIGMOD Record 39(1), 41–47 (2010)
7. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. In: PODS (2001)
8. Fraternali, P., Martinenghi, D., Tagliasacchi, M.: Top-k bounded diversification. In: SIGMOD. pp. 421–432 (2012)
9. Golenberg, K., Kimelfeld, B., Sagiv, Y.: Keyword proximity search in complex data graphs. In: SIGMOD. pp. 927–940 (2008)
10. Gollapudi, S., Sharma, A.: An axiomatic approach for result diversification. In: WWW. pp. 381–390 (2009)
11. Hassin, R., Rubinstein, S., Tamir, A.: Approximation algorithms for maximum dispersion. Operation Research Letters 21(3), 133–137 (1997)
12. Ilyas, I.F., Beskales, G., Soliman, M.A.: A survey of top-k query processing techniques in relational database systems. ACM Computing Surveys 40(4) (2008)
13. Jain, A., Sarda, P., Haritsa, J.R.: Providing diversity in k-nearest neighbor query results. In: PAKDD. pp. 404–413 (2004)
14. van Kreveld, M.J., Reinbacher, I., Arampatzis, A., van Zwol, R.: Multi-dimensional scattered ranking methods for geographic information retrieval. GeoInformatica 9(1), 61–84 (2005)
15. Mei, Q., Guo, J., Radev, D.R.: Divrank: the interplay of prestige and diversity in information networks. In: KDD. pp. 1009–1018 (2010)
16. Qin, L., Yu, J.X., Chang, L.: Diversifying top-k results. VLDB 5(11), 1124–1135 (2012)
17. Ravi, S., Rosenkrantz, D., Tayi, G.: Heuristic and special case algorithms for dispersion problems. Operations Research 42(2), 299–310 (1994)
18. Sacharidis, D., Deligiannakis, A.: Spatial cohesion queries. In: SIGSPATIAL (2015)
19. Vee, E., Srivastava, U., Shanmugasundaram, J., Bhat, P., Amer-Yahia, S.: Efficient computation of diverse query results. In: ICDE. pp. 228–236 (2008)
20. Vieira, M.R., Razente, H.L., Barioni, M.C.N., Hadjieleftheriou, M., Srivastava, D., Jr., C.T., Tsotras, V.J.: On query result diversification. In: ICDE. pp. 1163–1174 (2011)
21. Yu, C., Lakshmanan, L.V.S., Amer-Yahia, S.: It takes variety to make a world: diversification in recommender systems. In: EDBT. pp. 368–378 (2009)