DisC Diversity: Result Diversification based on Dissimilarity and Coverage*

Marina Drosou
Computer Science Department
University of Ioannina, Greece
mdrosou@cs.uoi.gr

Evaggelia Pitoura
Computer Science Department
University of Ioannina, Greece
pitoura@cs.uoi.gr

ABSTRACT

Recently, result diversification has attracted a lot of attention as a means to improve the quality of results retrieved by user queries. In this paper, we propose a new, intuitive definition of diversity called DisC diversity. A DisC diverse subset of a query result contains objects such that each object in the result is represented by a similar object in the diverse subset and the objects in the diverse subset are dissimilar to each other. We show that locating a minimum DisC diverse subset is an NP-hard problem and provide heuristics for its approximation. We also propose adapting DisC diverse subsets to a different degree of diversification. We call this operation zooming-in. We provide implementations of our algorithms based on the M-tree, a spatial index structure, and experimentally evaluate their performance.

1. INTRODUCTION

Result diversification has attracted considerable attention as a means of enhancing the quality of query results presented to users (e.g., [26, 32]). Consider, for example, a user who wants to buy a camera and submits a related query. A diverse result, i.e., a result containing various brands and models with different pixel counts and other technical characteristics is intuitively more informative than a homogeneous result containing only cameras with similar features.

There have been various definitions of diversity [10], based on (i) content (or similarity), i.e., objects that are dissimilar to each other (e.g., [32]), (ii) novelty, i.e., objects that contain new information when compared to what was previously presented (e.g., [9]) and (iii) semantic coverage, i.e., objects that belong to different categories or topics (e.g., [3]). Most previous approaches rely on assigning a diversity score to each object and then selecting either the k objects with the highest score for a given k (e.g., [4, 14]) or the objects with score larger than some threshold (e.g., [29]).

In this paper, we address diversity through a different perspective. Let $P$ be the set of objects in a query result. We consider two objects $p_1$ and $p_2$ in $P$ to be similar, if $\text{dist}(p_1, p_2) \leq r$ for some distance function $\text{dist}$ and real number $r$, where $r$ is a tuning parameter that we call radius. Given $P$, we select a representative subset $S \subseteq P$ to be presented to the user such that: (i) all objects in $P$ are similar with at least one object in $S$ and (ii) no two objects in $S$ are similar with each other. The first condition ensures that all objects in $P$ are represented, or covered, by at least one object in the selected subset. The second condition ensures that the selected objects of $P$ are dissimilar. We call the set $S$ a-Dissimilar and Covering subset or a-DisC diverse subset.

In contrary to previous approaches to diversification, we aim at computing subsets of objects that contain objects that are both dissimilar with each other and cover the whole result set. Furthermore, instead of specifying a required size $k$ of the diverse set or a threshold, our tuning parameter $r$ explicitly expresses the degree of diversification and determines the size of the diverse set. Increasing $r$ results in smaller, more diverse subsets, while decreasing $r$ results in larger, less diverse subsets. We call these operations, zooming-out and zooming-in respectively. One can also zoom-in or zoom-out locally to a specific object in the presented result.

As an example, consider searching for cities in Greece. Figure 1 shows the results of this query diversified based on geographical location for an initial radius $(a)$, after zooming-in $(b)$, zooming-out $(c)$ and local zooming-in a specific city $(d)$. As another example of local zooming in the case of categorical attributes, consider looking for cameras, where diversity refers to cameras with different features. Figure 2 depicts an initial most diverse result and the result of local zooming-in one individual camera in this result.

We formalize the problem of locating minimum DisC diverse subsets as an independent dominating set problem on graphs [17]. We provide a suite of heuristics for computing small DisC diverse subsets. We also consider the problem of adjusting the radius $r$. We explore the relation among DisC diverse subsets of different radii and provide algorithms for incrementally adapting a DisC diverse subset to a new radius. We provide theoretical upper bounds for the size of the diverse subsets produced by our algorithms for computing DisC diverse subsets as well as for their zooming counterparts. Since the crux of the efficiency of the proposed algorithms is locating neighbors, we take advantage of spatial data structures. In particular, we propose efficient algorithms based on the M-tree [30].

We compare the quality of our approach to other diversification methods both analytically and qualitatively. We also evaluate our various heuristics using both real and synthetic datasets. Our performance results show that the basic heuristic for computing dissimilar and covering subsets works faster than its greedy variation but produces larger sets. Relaxing the dissimilarity condition, although in theory could result in smaller sets, in our experiments does not...

*This paper has been granted the Reproducible Label by the VLDB 2013 reproducibility committee. More information concerning our evaluation system and experimental results can be found at http://www.dbxr.org.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Articles from this volume were invited to present their results at The 39th International Conference on Very Large Data Bases, August 26th - 30th 2013, Riva del Garda, Trento, Italy. Proceedings of the VLDB Endowment, Vol. 6, No. 1

Copyright 2012 VLDB Endowment 2150-8097/12/11... $10.00.
reduce the size of the result considerably. Our incremental algorithms for zooming in or out from \( r \) to a different radius \( r' \), when compared to computing a DisC diverse subset for \( r' \) from scratch, produce sets of similar sizes and closer to what the user intuitively expects, while imposing a smaller computational cost. Finally, we draw various conclusions for the M-tree implementation of these algorithms.

Most often diversification is modeled as a bi-criteria problem with the dual goal of maximizing both the diversity and the relevance of the selected results. In this paper, we focus solely on diversity. Since we “cover” the whole dataset, each user may “zoom-in” to the area of the results that seems most relevant to her individual needs. Of course, many other approaches to integrating relevance with DisC diversity are possible; we discuss some of them in Section 8.

In a nutshell, in this paper, we make the following contributions:

- we propose a new, intuitive definition of diversity, called DisC diversity and compare it with other models,
- we show that locating minimum DisC diverse subsets is an NP-hard problem and provide efficient heuristics along with approximation bounds,
- we introduce adaptive diversification through zooming-in and zooming-out and present algorithms for their incremental computation as well as corresponding theoretical bounds,
- we provide M-tree tailored algorithms and experimentally evaluate their performance.

The rest of the paper is structured as follows. Section 2 introduces DisC diversity and heuristics for computing small diverse subsets. Section 3 introduces adaptive diversification and Section 4 compares our approach with other diversification methods. In Section 5, we employ the M-tree for the efficient implementation of our algorithms, while in Section 6, we present experimental results. Finally, Section 7 presents related work and Section 8 concludes the paper.

2. DISC DIVERSITY

In this section, we first provide a formal definition of DisC diversity. We then show that locating a minimum DisC diverse subset of objects is an NP-hard problem and present heuristics for locating approximate solutions.

2.1 Definition of DisC Diversity

Let \( \mathcal{P} \) be a set of objects returned as the result of a user query. We want to select a representative subset \( S \) of these objects such that each object from \( \mathcal{P} \) is represented by a similar object in \( S \) and the objects selected to be included in \( S \) are dissimilar to each other.

We define similarity between two objects using a distance metric \( dist \). For a real number \( r, r \geq 0 \), we use \( N_r(p_i) \) to denote the set of neighbors (or neighborhood) of an object \( p_i \in \mathcal{P} \), i.e., the objects lying at distance at most \( r \) from \( p_i \):

\[
N_r(p_i) = \{ p_j | p_i \neq p_j \land dist(p_i, p_j) \leq r \}
\]

We use \( N_r^+(p_i) \) to denote the set \( N_r(p_i) \cup \{p_i\} \), i.e., the neighborhood of \( p_i \) including \( p_i \) itself. Objects in the neighborhood of \( p_i \) are considered similar to \( p_i \), while objects outside its neighborhood are considered dissimilar to \( p_i \). We define an \( r \)-DisC diverse subset as follows:

**Definition 1. (r-DisC Diverse Subset)** Let \( \mathcal{P} \) be a set of objects and \( r, r \geq 0 \), a real number. A subset \( S \subseteq \mathcal{P} \) is an \( r \)-DisCimilar-and-Covering diverse subset, or \( r \)-DisC diverse subset, of \( \mathcal{P} \), if the following two conditions hold:

(i) (coverage condition) \( \forall p_i \in \mathcal{P}, \exists p_j \in N_r^+(p_i) \) such that \( p_j \in S \) and (ii) (dissimilarity condition) \( \forall p_i, p_j \in S \) with \( p_i \neq p_j \), it holds that \( dist(p_i, p_j) > r \).

The first condition ensures that all objects in \( \mathcal{P} \) are represented by at least one similar object in \( S \) and the second condition that the objects in \( S \) are dissimilar to each other. We call each object \( p_i \in S \) an \( r \)-DisC diverse object and \( r \) the radius of \( S \). When the value of \( r \) is clear from context, we simply refer to \( r \)-DisC diverse objects as diverse objects.

Given \( \mathcal{P} \), we would like to select the smallest number of diverse objects.

**Definition 2. (The Minimum r-DisC Diverse Subset Problem)** Given a set \( \mathcal{P} \) of objects and a radius \( r \), find an \( r \)-DisC diverse subset \( S^* \) of \( \mathcal{P} \), such that, for every \( r \)-DisC diverse subset \( S \) of \( \mathcal{P} \), it holds that \( |S^*| \leq |S| \).

In general, there may be more than one minimum \( r \)-DisC diverse subsets of \( \mathcal{P} \) (see Figure 3(a) for an example).

2.2 Graph Representation and NP-hardness

Let us consider the following graph representation of a set \( \mathcal{P} \) of objects. Let \( G_{r, \mathcal{P}} = (V, E) \) be an undirected graph such that there is a vertex \( v_i \in V \) for each object \( p_i \in \mathcal{P} \) and an edge \( (v_i, v_j) \in E \), if and only if, \( dist(p_i, p_j) \leq r \) for the corresponding objects \( p_i, p_j \). An example is shown in Figure 3(b).

Let us recall a couple of graph-related definitions. A dominating set \( D \) for a graph \( G \) is a subset of vertices of \( G \) such that every vertex of \( G \) not in \( D \) is joined to at least one vertex of \( D \) by some edge. An independent set \( I \) for a graph
NP-hard even for special kinds of graphs, such as for unit disk graphs [15]. The problem remains for a set \( I \) is equivalent to finding a Minimal Independent Dominating Set of the corresponding graph \( G_{P, r} \).

The Minimal Independent Dominating Set Problem has been proven to be NP-hard [15]. The problem remains NP-hard even for special kinds of graphs, such as for unit disk graphs [8]. Unit disk graphs are graphs whose vertices can be put in one to one correspondence with equisized circles in a plane such that two vertices are joined by an edge, if and only if, the corresponding circles intersect. \( G_{P, r} \) is a unit disk graph for Euclidean distances.

In the following, we use the terms dominance and coverage, as well as, independence and dissimilarity interchangeably. In particular, two objects \( p_i \) and \( p_j \) are independent, if \( \text{dist}(p_i, p_j) > r \). We also say that an object covers all objects in its neighborhood. We next present some useful properties that relate the coverage (i.e., dominance) and dissimilarity (i.e., independence) conditions. A maximal independent set is an independent set such that adding any other vertex to the set forces the set to contain an edge, that is, it is an independent set that is not a subset of any other independent set. It is known that:

**Lemma 1.** An independent set of a graph is maximal, if and only if, it is dominating.

From Lemma 1, we conclude that:

**Observation 2.** A minimum maximal independent set is also a minimum independent dominating set.

However, **Observation 3.** A minimum dominating set is not necessarily independent.

For example, in Figure 4, the minimum dominating set of the depicted objects is of size 2, while the minimum independent dominating set is of size 3.

### 2.3 Computing DisC Diverse Objects

We consider first a baseline algorithm for computing a DisC diverse subset \( S \) of \( P \). For presentation convenience, let us call black the objects of \( P \) that are in \( S \), grey the objects covered by \( S \) and white the objects that are neither black nor grey. Initially, \( S \) is empty and all objects are white. The algorithm proceeds as follows: until there are no more white objects, it selects an arbitrary white object \( p_i \), colors \( p_i \) black and colors all objects in \( N_r(p_i) \) grey. We call this algorithm Basic-DisC.

The produced set \( S \) is clearly an independent set, since once an object enters \( S \), all its neighbors become grey and thus are withdrawn from consideration. It is also a maximal independent set, since at the end there are only grey objects left, thus adding any of them to \( S \) would violate the independence of \( S \). From Lemma 1, the set \( S \) produced by Basic-DisC is an \( r \)-DisC diverse subset. \( S \) is not necessarily a minimum \( r \)-DisC diverse subset. However, its size is related to the size of any minimum \( r \)-DisC diverse subset \( S^* \) as follows:

**Theorem 1.** Let \( B \) be the maximum number of independent neighbors of any object in \( P \). Any \( r \)-DisC diverse subset \( S \) of \( P \) is at most \( B \) times larger than any minimum \( r \)-DisC diverse subset \( S^* \).

**Proof.** Since \( S \) is an independent set, any object in \( S^* \) can cover at most \( B \) objects in \( S \) and thus \( |S| \leq B|S^*| \).

The value of \( B \) depends on the distance metric used and also on the dimensionality \( d \) of the data space. For many distance metrics \( B \) is a constant. Next, we show how \( B \) is bounded for specific combinations of the distance metric \( \text{dist} \) and data dimensionality \( d \).

**Lemma 2.** If \( \text{dist} \) is the Euclidean distance and \( d = 2 \), each object \( p_i \) in \( P \) has at most 5 neighbors that are independent from each other.

**Proof.** Let \( p_1, p_2 \) be two independent neighbors of \( p_i \). Then, it must hold that \( \angle p_1p_ip_2 \) is larger than \( \frac{\pi}{2} \). Otherwise, \( \text{dist}(p_1, p_2) \leq \max\{\text{dist}(p_1, p_1), \text{dist}(p_1, p_2)\} \leq r \) which contradicts the independence of \( p_1 \) and \( p_2 \). Therefore, \( p_i \) can have at most \( 2 \pi r^2 - 1 = 5 \) independent neighbors.

**Lemma 3.** If \( \text{dist} \) is the Manhattan distance and \( d = 2 \), each object \( p_i \) in \( P \) has at most 7 neighbors that are independent from each other.

**Proof.** The proof can be found in the Appendix.

Also, for \( d = 3 \) and the Euclidean distance, it can be shown that each object \( p_i \) in \( P \) has at most 24 neighbors that are independent from each other. This can be shown using packing techniques and properties of solid angles [11].

We now consider the following intuitive greedy variation of Basic-DisC, that we call Greedy-DisC. Instead of selecting white objects arbitrarily at each step, we select the white object with the largest number of white neighbors, that is, the white object that covers the largest number of uncovered objects. Greedy-DisC is shown in Algorithm 1, where \( N_{r}^{W}(p_i) \) is the set of the white neighbors of object \( p_i \).

While the size of the \( r \)-DisC diverse subset \( S \) produced by Greedy-DisC is expected to be smaller than that of the subset produced by Basic-DisC, the fact that we consider for inclusion in \( S \) only white, i.e., independent, objects may still not reduce the size of \( S \) as much as expected. From Observation 3, it is possible that an independent covering set is larger than a covering set that also includes dependent objects. For example, consider the nodes (or equivalently the corresponding objects) in Figure 4. Assume that object \( v_2 \) is inserted in \( S \) first, resulting in objects \( v_1, v_3 \) and \( v_6 \) becoming grey. Then, we need two more objects, namely, \( v_4 \) and \( v_5 \), for covering all objects. However, if we consider for inclusion grey objects as well, then \( v_5 \) can join \( S \), resulting in a smaller covering set.

**Figure 3:** (a) Minimum \( r \)-DisC diverse subsets for the depicted objects: \( \{p_1, p_4, p_7\}, \{p_2, p_4, p_7\}, \{p_3, p_5, p_6\} \) and (b) their graph representation.

**Figure 4:** (a) Minimum dominating set \( (v_2, v_5) \) and (b) a minimum independent dominating set \( (v_2, v_4, v_6) \) for the depicted graph.
Algorithm 1 Greedy-DisC

Input: A set of objects \( P \) and a radius \( r \).
Output: An \( r \)-DisC diverse subset \( S \) of \( P \).

1: \( S \leftarrow \emptyset \)
2: for all \( p_i \in P \) do
3:   color \( p_i \) white
4: end for
5: while there exist white objects do
6:   select the white object \( p_i \) with the largest \( |N^W_i(p_i)| \)
7:   \( S = S \cup \{p_i\} \)
8:   color \( p_i \) black
9: for all \( p_i \in N^w_r(p_i) \) do
10:   color \( p_i \) grey
11: end for
12: end while
13: return \( S \)

Motivated by this observation, we also define \( r \)-C diverse subsets that satisfy only the coverage condition of Definition 1 and modify Greedy-DisC accordingly to compute \( r \)-C diverse sets. The only change required is that in line 6 of Algorithm 1, we select both white and grey objects. This allows us to select at each step the object that covers the largest possible number of uncovered objects, even if this object is grey. We call this variation Greedy-C. In the case of Greedy-C, we prove the following bound for the size of the produced \( r \)-C diverse subset \( S \):

**Theorem 2.** Let \( \Delta \) be the maximum number of neighbors of any object in \( P \). The \( r \)-C diverse subset produced by Greedy-C is at most in \( \Delta \) times larger than the minimum \( r \)-DisC diverse subset \( S' \).

**Proof.** The proof can be found in the Appendix. □

3. ADAPTIVE DIVERSIFICATION

The radius \( r \) determines the desired degree of diversification. A large radius corresponds to fewer and less similar to each other representative objects, whereas a small radius results in more and less dissimilar representative objects. On one extreme, a radius equal to the largest distance between any two objects results in a single object being selected and on the other extreme, a zero radius results in all objects of \( P \) being selected. We consider an interactive mode of operation where, after being presented with an initial set of results for some \( r \), a user can see either more or less results by correspondingly decreasing or increasing \( r \).

Specifically, given a set of objects \( P \) and an \( r \)-DisC diverse subset \( S' \) of \( P \), we want to compute an \( r' \)-DisC diverse subset \( S'' \) of \( P \). There are two cases: (i) \( r' < r \) and (ii) \( r' > r \) which we call zooming-in and zooming-out respectively. These operations are global in the sense that the radius \( r \) is modified similarly for all objects in \( P \). We may also modify the radius for the specific area of the data set. Consider, for example, a user that receives an \( r \)-DisC diverse subset \( S' \) of the results and finds some specific object \( p_i \in S' \) more or less interesting. Then, the user can zoom-in or zoom-out by specifying a radius \( r' \), \( r' < r \) or \( r' > r \), respectively, centered on \( p_i \). We call these operations local zooming-in and local zooming-out respectively.

To study the size relationship between \( S' \) and \( S'' \), we define the set \( N'_{r_1,r_2}(p_i), r_2 \geq r_1, \) as the set of objects at distance at most \( r_2 \) from \( p_i \) which are at distance at least \( r_1 \) from each other, i.e., objects in \( N_{r_2}(p_i) \setminus N_{r_1}(p_i) \) that are independent from each other considering the radius \( r_1 \). The following lemma bounds the size of \( N'_{r_1,r_2}(p_i) \) for specific distance metrics and dimensionality.

**Lemma 4.** Let \( r_1, r_2 \) be two radii with \( r_2 \geq r_1 \). Then, for \( d \leq 2 \):

\[
|N'_{r_1,r_2}(p_i)| \leq 9 \log_\beta(\frac{r_2}{r_1}), \quad \text{where } \beta = 1 + \sqrt{5}
\]

\[
|N'_{r_1,r_2}(p_i)| \leq 4 \sum_{i=1}^{\gamma} (2i + 1), \quad \text{where } \gamma = \left[ \frac{r_2 - r_1}{r_1} \right]
\]

**Proof.** The proof can be found in the Appendix. □

Since we want to support an incremental mode of operation, the set \( S'' \) should be as close as possible to the already seen set \( S' \). Ideally, \( S'' \supseteq S' \), for \( r' < r \) and \( S'' \subseteq S' \), for \( r' > r \). We would also like the size of \( S'' \) to be as close as possible to the size of the minimum \( r'-\text{DisC} \) diverse subset.

If we consider only the coverage condition, that is, only \( r \)-C diverse subsets, then an \( r \)-C diverse subset of \( P \) is also an \( r'-\text{C} \) diverse subset of \( P \), for any \( r' \geq r \). This holds because \( N_r(p_i) \subseteq N_{r'}(p_i) \) for any \( r' \geq r \). However, a similar property does not hold for the dissimilarity condition. In general, a maximal independent diverse subset \( S'' \) of \( P \) for \( r \) is not necessarily a maximal independent diverse subset of \( P \), neither for \( r' > r \) nor for \( r' < r \). To see this, note that for \( r' > r \), \( S'' \) may not be independent, whereas for \( r' < r \), \( S'' \) may no longer be maximal. Thus, from Lemma 1, we reach the following conclusion:

**Observation 4.** In general, there is no monotonic property among the \( r \)-DisC diverse and the \( r' \)-DisC diverse subsets of a set of objects \( P \), for \( r \neq r' \).

For zooming-in, i.e., for \( r' < r \), we can construct \( r'-\text{DisC} \) diverse sets that are supersets of \( S'' \) by adding objects to \( S'' \) to make it maximal. For zooming-out, i.e., for \( r' > r \), in general, there may be no subset of \( S'' \) that is \( r'-\text{DisC} \) diverse. Take for example the objects of Figure 5(b) with \( S'' = \{p_1,p_2,p_3\} \). No subset of \( S'' \) is an \( r' \)-DisC diverse subset for this set of objects. Next, we detail the zooming-in and zooming-out operations.

3.1 Incremental Zooming-in

Let us first consider the case of zooming with a smaller radius, i.e., \( r' < r \). Here, we aim at producing a small independent covering solution \( S'' \), such that, \( S'' \supseteq S' \). For this reason, we keep the objects of \( S'' \) in the new \( r'-\text{DisC} \) diverse subset \( S'' \) and proceed as follows.

Consider an object of \( S' \), for example \( p_1 \) in Figure 5(a). Objects at distance at most \( r' \) from \( p_1 \) are still covered by \( p_1 \) and cannot enter \( S'' \). Objects at distance greater than \( r' \) and at most \( r \) may be uncovered and join \( S'' \). Each of these objects can enter \( S'' \) as long as it is not covered by some other object of \( S'' \) that lays outside the former neighborhood of \( p_1 \). For example, in Figure 5(a), \( p_4 \) and \( p_5 \) may enter \( S'' \).
greater than $N$ there are at most $S$ diverse subset, we proceed in two passes. In the first pass, can be either arbitrary (Greedy-Zoom-In and (ii) covers the largest number of uncovered objects is selectedceed in a greedy way, where at each turn the object that becomes larger, becomes removed from the solution, since they are no longer dissimilar to $p_1$. Thus, the size differenc between $S'$ and $S''$ is expected to be smaller than this theoretical upper bound.

### 3.2 Incremental Zooming-out

Next, we consider zooming with a larger radius, i.e., $r' > r$. In this case, the user is interested in seeing less and more dissimilar objects, ideally a subset of the already seen results for $r$, that is, $S' \subseteq S'$. However, in this case, as discussed, in contrast to zooming-in, it may not be possible to construct a diverse subset $S''$ that is a subset of $S'$. Thus, we focus on the following sets of objects: (i) $S' \setminus S''$, and (ii) $S'' \setminus S'$. The first set consists of the objects that belong to the previous diverse subset but are removed from the new one, while the second set consists of the new objects added to $S''$. To illustrate, let us consider the objects of Figure 5(b) and that $p_1, p_2, p_3 \in S'$. Since the radius becomes larger, $p_1$ now covers all objects at distance at most $r'$ from it. This may include a number of objects that also belonged to $S'$, such as $p_2$. These objects have to be removed from the solution, since they are no longer dissimilar to $p_1$. However, removing such an object, say $p_2$ in our example, can potentially leave uncovered a number of objects that were previously covered by $p_2$ (the shaded area of Figure 5(b)). In our example, requiring $p_2$ to remain in $S''$ means than $p_2$ should be now added to $S''$.

To produce an $r'$-DisC diverse subset based on an $r$-DisC diverse subset, we proceed in two passes. In the first pass, we examine all objects of $S'$ in some order and remove their diverse neighbors that are now covered by them. At the second pass, objects from any uncovered areas are added to $S'$. Again, we have an arbitrary and a greedy variation, denoted Basic-Zoom-Out and Greedy-Zoom-Out respectively. Algorithm 3 shows the greedy variation; the first pass (lines 4-11) considers $S' \setminus S''$, while the second pass (lines 12-15) considers $S'' \setminus S'$. Initially, we color all previously black objects red. All other objects are colored white. We consider three variations for the first pass of the greedy algorithm: selecting the red objects with (a) the largest number of red neighbors, (b) the smallest number of red neighbors and (c) the largest number of white neighbors. Variations (a) and (c) aim at minimizing the objects to be added in the second pass, that is, $S'' \setminus S'$, while variation (b) aims at maximizing $S' \cap S''$. Algorithm 3 depicts variation (a), where $N_i^{r'}(p_i)$ denotes the set of red neighbors of object $p_i$.

**Lemma 5.** For the set $S'$ generated by the Basic-Zoom-In and Greedy-Zoom-In algorithms, it holds that:

(i) $S' \subseteq S''$ and

(ii) $|S'| \leq N_i^{r'}(p_i)|S''|

**Proof.** Condition (i) trivially holds from step 1 of the algorithm. Condition (ii) holds since for each object in $S'$ there are at most $N_i^{r'}(p_i)$ independent objects at distance greater than $r'$ from each other that can enter $S''$. 

In practice, objects selected to enter $S''$, such as $p_2$ and $p_3$ in Figure 5(a), are likely to cover other objects left uncovered by the same or similar objects in $S'$. Therefore, the size difference between $S'$ and $S''$ is expected to be smaller than this theoretical upper bound.

### 4. COMPARISON WITH OTHER MODELS

The most widely used diversification models are MaxMin and MaxSum that aim at selecting a subset $S$ of $P$ so as
the minimum or the average pairwise distance of the selected objects is maximized (e.g., [16, 27, 6]). More formally, an optimal MAXMIN (resp., MAXSUM) subset of $P$ is a subset with the maximum $f_{\text{MIN}} = \min_{p_i, p_j \in S} \text{dist}(p_i, p_j)$ (resp., $f_{\text{SUM}} = \sum_{p_i, p_j \in S} \text{dist}(p_i, p_j)$) over all subsets of the same size.

Let us first compare analytically the quality of an $r$-DisC solution to the optimal values of these metrics.

**Lemma 7.** Let $P$ be a set of objects, $S$ be an $r$-DisC diverse subset of $P$ and $\lambda$ be the $f_{\text{MIN}}$ distance between objects of $S$. Let $S^*$ be an optimal MAXMIN subset of $P$ for $k = |S|$ and $\lambda^*$ be the $f_{\text{MIN}}$ distance of $S^*$. Then, $\lambda^* \leq 3 \lambda$.

**Proof.** Each object in $S^*$ is covered by (at least) one object in $S$. There are two cases, either (i) all objects $p_1, p_2 \in S^*$, $p_1 \neq p_2$, are covered by different objects is $S$, or (ii) there are at least two objects in $S^*$, $p_1, p_2, p_1 \neq p_2$ that are both covered by the same object $p$ in $S$. Case (i): Let $p_1$ and $p_2$ be two objects in $S$ such that $\text{dist}(p_1, p_2) = \lambda$ and $p_1$ and $p_2$ respectively be the objects in $S^*$ that each covers. Then, by applying the triangle inequality twice, we get: $\text{dist}(p_1, p_2) \leq \text{dist}(p_1, p_1) + \text{dist}(p_1, p_2) \leq \text{dist}(p_1, p_1) + \text{dist}(p_1, p_2) + \text{dist}(p_2, p_2)$. By coverage, we get: $\text{dist}(p_1, p_2) \leq r + \lambda \leq 3 \lambda$, thus $\lambda^* \leq 3 \lambda$. Case (b): Let $p_1^* p_2^*$ be two objects in $S^*$ that are covered by the same object $p$ in $S$. Then, by coverage and the triangle inequality, we get $\text{dist}(p_1^*, p_2^*) \leq \text{dist}(p_1, p) + \text{dist}(p_2, p) \leq 2r$, thus $\lambda^* \leq 2 \lambda$. □

**Lemma 8.** Let $P$ be a set of objects, $S$ be an $r$-DisC diverse subset of $P$ and $\sigma$ be the $f_{\text{SUM}}$ distance between objects of $S$. Let $S^*$ be an optimal MAXSUM subset of $P$ for $k = |S|$ and $\sigma^*$ be the $f_{\text{SUM}}$ distance of $S^*$. Then, $\sigma^* \leq 3 \sigma$.

**Proof.** We consider the same two cases for the objects in $S$ covering the objects in $S^*$ as in the proof of Lemma 7. Case (i): Let $p_1^*$ and $p_2^*$ be two objects in $S^*$ and $p_1$ and $p_2$ be the objects in $S$ that cover them respectively. Then, by applying the triangle inequality twice, we get: $\text{dist}(p_1^*, p_2^*) \leq \text{dist}(p_1^*, p_1) + \text{dist}(p_1, p_2) \leq \text{dist}(p_1^*, p_1) + \text{dist}(p_1, p_2) + \text{dist}(p_2, p_2)$. By coverage, we get: $\text{dist}(p_1^*, p_2^*) \leq 2r + \text{dist}(p_1, p_2) \ (1)$. Case (ii): Let $p_1^*$ and $p_2^*$ be two objects in $S^*$ that are covered by the same object $p$ in $S$. Then, by coverage and the triangle inequality, we get: $\text{dist}(p_1^*, p_2^*) \leq \text{dist}(p_1^*, p) + \text{dist}(p_2, p) \leq 2r \ (2)$. From (1) and (2), we get: $\sum_{p_1, p_2 \in S, p_1 \neq p_2} \text{dist}(p_1^*, p_2^*) \leq \sum_{p_1, p_2 \in S, p_1 \neq p_2} 2r + \text{dist}(p_1^*, p_2^*)$. From independence, $\forall p_1, p_2 \in S$, $p_1 \neq p_2$, $\text{dist}(p_1, p_2) \geq r$. Thus, $\sigma^* \leq 3 \sigma$. □

Next, we present qualitative results of applying MAXMIN and MAXSUM to a 2-dimensional “Clustered” dataset (Figure 6). To implement MAXMIN and MAXSUM, we used greedy heuristics which have been shown to achieve good solutions [10]. In addition to MAXMIN and MAXSUM, we also show results for $r$-C diversity (i.e., covering but not necessarily independent subsets for the given $r$) and $k$-medoids, a widespread clustering algorithm that seeks to minimize $\frac{1}{|P|} \sum_{p_i \in P} \text{dist}(p_i, c(p_i))$, where $c(p_i)$ is the closest object of $p_i$ in the selected subset, since the located medoids can be viewed as a representative subset of the dataset. To allow for a comparison, we first run Greedy-DisC for a given $r$ and then use as $k$ the size of the produced diverse subset. In this example, $k = 15$ for $r = 0.7$.

MAXSUM diversification and $k$-medoids fail to cover all areas of the dataset; MAXSUM tends to focus on the outskirts of the dataset, whereas $k$-medoids clustering reports only central points, ignoring outliers. MAXMIN performs better in this aspect. However, since MAXMIN seeks to retrieve objects that are as far apart as possible, it fails to retrieve objects from dense areas; see, for example, the center areas of the clusters in Figure 6. DisC gives priority to such areas and, thus, such areas are better represented in the solution. Note also that MAXSUM and $k$-medoids may select near duplicates, as opposed to DisC and MAXMIN. We also experimented with variations of MAXSUM proposed in [27] but the results did not differ substantially from the ones in Figure 6(b). For $r$-C diversity, the resulting selected set is one object smaller, however, the selected objects are less widely spread than in DisC. Finally, note that, we are not interested in retrieving as representatives subsets that follow the same distribution as the input dataset, as in the case of sampling, since such subsets will tend to ignore outliers.

Instead, we want to cover the whole dataset and provide a complete view of all its objects, including distant ones.

**5. IMPLEMENTATION**

Since a central operation in computing DisC diverse subsets is locating neighbors, we introduce algorithms that exploit a spatial index structure, namely, the M-tree [30]. An M-tree is a balanced index tree that can handle large volumes of dynamic data of any dimensionality in general metric spaces. In particular, an M-tree partitions space around some of the indexed objects, called pivots, by forming a bounding ball region of some covering radius around them. Let $c$ be the maximum node capacity of the tree. Internal nodes have at most $c$ entries, each containing a pivot object $p_v$, the covering radius $r_v$ around $p_v$, the distance of $p_v$ from its parent pivot and a pointer to the subtree $t_v$. All objects in the subtree $t_v$ rooted at $p_v$ are within distance at most equal to the covering radius $r_v$ from $p_v$. Leaf nodes have entries containing the indexed objects and their distance from their parent pivot.

The construction of an M-tree is influenced by the splitting policy that determines how nodes are split when they exceed their maximum capacity $c$. Splitting policies indicate (i) which two of the $c + 1$ available pivots will be promoted to the parent node to index the two new nodes (promote policy) and (ii) how the rest of the pivots will be assigned to the two new nodes (partition policy). These policies affect the overlap among nodes. For computing diverse subsets:

(i) We link together all leaf nodes. This allows us to visit all objects in a single left-to-right traversal of the leaf

**Figure 6: Solutions by the various diversification methods for a clustered dataset. Selected objects are shown in bold. Circles denote the radius $r$ of the selected objects.**
nodes and exploit some degree of locality in covering the objects.

(ii) To compute the neighbors \( N_r(p_i) \) of an object \( p_i \) at radius \( r \), we perform a range query centered around \( p_i \) with distance \( r \), denoted \( Q(p_i, r) \).

(iii) We build trees using splitting policies that minimize overlap. In most cases, the policy that resulted in the lowest overlap was (a) promoting as new pivots the pivot \( p_i \) of the overflowed node and the object \( p_j \) with the maximum distance from \( p_i \) and (b) partitioning the objects by assigning each object to the node whose pivot has the closest distance with the object. We call this policy “MinOverlap”.

5.1 Computing Diverse Subsets

Basic-DisC. The Basic-DisC algorithm selects white objects in random order. In the M-tree implementation of Basic-DisC, we consider objects in the order they appear in the leaves of the M-tree, thus taking advantage of locality. Upon encountering a white object \( p_i \) in a leaf, the algorithm colors it black and executes a range query \( Q(p_i, r) \) to retrieve and color grey its neighbors. Since the neighbors of an indexed object are expected to reside in nearby leaf nodes, such range queries are in general efficient. We can visualize the progress of Basic-DisC as gradually coloring all objects in the leaf nodes from left-to-right until all objects become either grey or black.

Greedy-DisC. The Greedy-DisC algorithm selects at each iteration the white object with the largest white neighborhood (line 6 of Algorithm 1). To efficiently implement this selection, we maintain a sorted list, \( L \), of all white objects ordered by the size of their white neighborhoods. To initialize \( L \), note that, at first, for all objects \( p_i \), \( N^w_i(p_i) = N_i(p_i) \). We compute the neighborhood size of each object incrementally as we build the M-tree. We found that computing the size of neighborhoods while building the tree (instead of performing one range query per object after building the tree) reduces node accesses up to 45%.

Considering the maintenance of \( L \), each time an object \( p_i \) is selected and colored black, its neighbors at distance \( r \) are colored grey. Thus, we need to update the size of the white neighborhoods of all affected objects. We consider two variations. The first variation, termed Grey-Greedy-DisC, executes, for each newly colored grey neighbor \( p_j \) of \( p_i \), a range query \( Q(p_j, r) \) to locate its neighbors and reduce by one the size of their white neighborhood. The second variation, termed White-Greedy-DisC, executes one range query for all remaining white objects within distance less than or equal to \( 2r \) from \( p_i \). These are the only white objects whose white neighborhood may have changed. Since the cost of maintaining the exact size of the white neighborhoods may be large, we also consider “lazy” variations. Lazy-Grey-Greedy-DisC involves grey neighbors at some distance smaller than \( r \), while Lazy-White-Greedy-DisC involves white objects at some distance smaller than \( 2r \).

Pruning. We make the following observation that allows us to prune subtrees while executing range queries. Objects that are already grey do not need to be colored grey again when some other of their neighbors is colored black.

Pruning Rule: A leaf node that contains no white objects is colored grey. When all its children become grey, an internal node is colored grey. While executing range queries, any top-down search of the tree does not need to follow subtrees rooted at grey nodes.

As the algorithms progresses, more and more nodes become grey, and thus, the cost of range queries reduces over time. For example, we can visualize the progress of the Basic-DisC (Pruned) algorithm as gradually coloring all tree nodes grey in a post-order manner.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default value</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>M-tree node capacity</td>
<td>30</td>
<td>25 - 100</td>
</tr>
<tr>
<td>M-tree splitting policy</td>
<td>MinOverlap</td>
<td>various</td>
</tr>
<tr>
<td>Dataset cardinality</td>
<td>10000</td>
<td>579 - 50000</td>
</tr>
<tr>
<td>Dataset dimensionality</td>
<td>2</td>
<td>2 - 10</td>
</tr>
<tr>
<td>Dataset distribution</td>
<td>normal</td>
<td>uniform, normal</td>
</tr>
<tr>
<td>Distance metric</td>
<td>Euclidean</td>
<td>Euclidean, Hamming</td>
</tr>
</tbody>
</table>

Greedy-C. The Greedy-C algorithm considers at each iteration both grey and white objects. A sorted structure \( L \) has to be maintained as well, which now includes both white and grey objects and is substantially larger. Furthermore, the pruning rule is no longer useful, since grey objects and nodes need to be accessed again for updating the size of their white neighborhood.

5.2 Adapting the Radius

For zooming-in, given an \( r \)-DisC diverse subset \( S^r \) of \( P \), we would like to compute an \( r' \)-DisC diverse subset \( S^{r'} \) of \( P \), \( r' < r \), such that, \( S^r \supseteq S^{r'} \). A naive implementation would require as a first step locating the objects in \( N^r_{i_{max}}(p_i) \) (line 3 of Algorithm 2) by invoking two range queries for each \( p_i \) (with radius \( r \) and \( r' \) respectively). Then, a diverse subset of the objects in \( N^r_{i_{max}}(p_i) \) is computed either in a basic or in a greedy manner. However, during the construction of \( S^r \), objects in the corresponding M-tree have already been colored black or grey. We use this information based on the following rule.

Zooming Rule: Black objects of \( S^r \) maintain their color in \( S^{r'} \). Grey objects maintain their color as long as there exists a black object at distance at most \( r' \) from them.

Therefore, only grey nodes with no black neighbors at distance \( r' \) may turn black and enter \( S^{r'} \). To be able to apply this rule, we augment the leaf nodes of the M-tree with the distance of each indexed object to its closest black neighbor.

The Basic-Zoom-In algorithm requires one pass of the leaf nodes. Each time a grey object \( p_i \) is encountered, we check whether it is still covered, i.e., whether its distance from its closest black neighbor is smaller or equal to \( r' \). If not, \( p_i \) is colored black and a range query \( Q(p_i, r') \) is executed to locate and color grey the objects for which \( p_i \) is now their closest black neighbor. At the end of the pass, the black objects of the leaves form \( S^{r'} \). The Greedy-Zoom-In algorithm involves the maintenance of a sorted structure \( L \) of all white objects. To build this structure, the leaf nodes are traversed, grey objects that are now found to be uncovered are colored white and inserted into \( L \). Then, the white neighborhoods of objects in \( L \) are computed and \( L \) is sorted accordingly.

Zooming-out and local zooming algorithms are implemented similarly.

6. EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our algorithms using both synthetic and real datasets. Our synthetic datasets consist of multidimensional objects, where values at each dimension are in \([0, 1]\). Objects are either uniformly distributed in space (“Uniform”) or form (hyper) spherical clusters of different sizes (“Clustered”). We also employ two real datasets. The first one (“Cities”) is a collection of 2-dimensional points representing geographic information about 5922 cities and villages in Greece [2]. We normalized the values of this dataset in \([0, 1]\). The second real dataset (“Cameras”) consists of 7 characteristics for 579 digital cameras from [1], such as brand and storage type.
We use the Euclidean distance for the synthetic datasets and “Cities”, while for “Cameras”, whose attributes are categorical, we use $\text{dist}(p_i, p_j) = \sum \delta(p_i, p_j)$, where $\delta(p_i, p_j)$ is equal to 1, if $p_i$ and $p_j$ differ in the $i^{th}$ dimension and 0 otherwise, i.e., the Hamming distance. Note that the choice of an appropriate distance metric is an important but orthogonal to our approach issue.

Table 1 summarizes the values of the input parameters used in our experiments and Table 2 summarizes the algorithms employed.

Solution Size. We first compare our various algorithms in terms of the size of the computed diverse subset (Table 3). We consider Basic-DisC, Greedy-DisC (note that, both Grey- Greedy-DisC and White-Greedy-DisC produce the same solution) and Greedy-C. We also tested “lazy” variations of the greedy algorithm, namely Lazy-Grey-Greedy-DisC with distance $r/2$ and Lazy-White-Greedy-DisC with distance $3r/2$. Grey-Greedy-DisC locates a smaller DisC diverse subset than Basic-DisC in all cases. The lazy variations also perform better than Basic-DisC and comparable with Grey-Greedy-DisC. Lazy-White-Greedy-DisC seems to approximate better the actual size of the white neighborhoods than Grey-Greedy-DisC and in most cases produces smaller subsets. Greedy-C produces subsets with size similar with those produced by Greedy-Greedy-DisC. This means that raising the independence assumption does not lead to substantially smaller diverse subsets in our datasets. Finally, note that the diverse subsets produced by all algorithms for the “Clustered” dataset are smaller than for the “Uniform” one, since objects are generally more similar to each other.

Computational Cost. Next, we evaluate the computational cost of our algorithms, measured in terms of tree node accesses. Figure 7 reports this cost, as well as, the cost savings when the pruning rule of Section 5 is employed for Basic-DisC and Greedy-DisC (as previously detailed, this pruning cannot be applied to Greedy-C). Greedy-DisC has higher cost than Basic-DisC. The additional computational cost becomes more significant as the radius increases. The reason for this is that Greedy-DisC performs significantly more range queries. As the radius increases, objects have more neighbors and, thus, more M-tree nodes need to be accessed in order to retrieve them, color them and update the size of the neighborhoods of their neighbors. On the contrary, the cost of Basic-DisC is reduced when the radius increases, since it does not need to update the size of any neighborhood. For larger radii, more objects are colored grey by each selected (black) object and, therefore, less range queries are performed. Both algorithms benefit from pruning (up to 50% for small radii). We also experimented with employing bottom-up rather than top-down range queries. At most cases, the benefit in node accesses was less than 5%.

Figure 8 compares Grey-Greedy-DisC with White-Greedy-DisC and their corresponding “lazy” variations. We see that White-Greedy-DisC performs better than Grey-Greedy-DisC for the clustered dataset as $r$ increases. This is because in this case, grey objects share many common white neighbors which are accessed multiple times by Grey-Greedy-DisC for updating their white neighborhood size and only once by White-Greedy-DisC. The lazy variations reduce the computational cost further.

In the rest of this section, unless otherwise noted, we use the (Grey-)Greedy-DisC (Pruned) heuristic.

Impact of Dataset Cardinality and Dimensionality. For this experiment, we employ the “Clustered” dataset and vary its cardinality from 5000 to 15000 objects and its dimensionality from 2 to 10 dimensions. Figure 9 shows the corresponding solution size and computational cost as computed by the Greedy-DisC heuristic. We observe that the solution size is more sensitive to changes in cardinality when the radius is small. The reason for this is that for large radii, a selected object covers a large area in space. Therefore, even when the cardinality increases and there are many available objects to choose from, these objects are quickly covered by the selected ones. In Figure 9(b), the increase in the computational cost is due to the increase of range queries required to maintain correct information about the size of the white neighborhoods. Increasing the dimensionality of the dataset causes more objects to be selected as diverse as shown in Figure 9(c). This is due to the “curse of dimensionality” effect, since space becomes sparser at higher dimensions. The computational cost may however vary as dimensionality increases, since it is influenced by the cost of computing the neighborhood size of the objects that are colored grey.

### Table 2: Algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic-DisC</td>
<td>B-DisC</td>
<td>Selects objects in order of appearance in the leaf level of the M-tree.</td>
</tr>
<tr>
<td>Grey-Greedy-DisC</td>
<td>G-DisC</td>
<td>Selects each iteration the white object $p$ with the largest white neighborhood.</td>
</tr>
<tr>
<td>Greedy-C</td>
<td>G-C</td>
<td>Selects at each iteration the non-black object $p$ with the largest white neighborhood.</td>
</tr>
<tr>
<td>Lazy-Grey-Greedy-DisC</td>
<td>L-Gr-G-DisC</td>
<td>One range query per grey node at distance $r$ from $p$.</td>
</tr>
<tr>
<td>Lazy-White-Greedy-DisC</td>
<td>L-Wh-G-DisC</td>
<td>One range query per white node at distance $2r$ from $p$.</td>
</tr>
</tbody>
</table>

### Table 3: Solution size.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-DisC</td>
<td>Selects objects in order of appearance in the leaf level of the M-tree.</td>
</tr>
<tr>
<td>G-DisC</td>
<td>Selects each iteration the white object $p$ with the largest white neighborhood.</td>
</tr>
<tr>
<td>L-Gr-G-DisC</td>
<td>One range query per grey node at distance $r$ from $p$.</td>
</tr>
<tr>
<td>L-Wh-G-DisC</td>
<td>One range query per white node at distance $2r$ from $p$.</td>
</tr>
<tr>
<td>G-C</td>
<td>Selects at each iteration the non-black object $p$ with the largest white neighborhood.</td>
</tr>
</tbody>
</table>

### Table 4: Solution size.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-DisC</td>
<td>Selects objects in order of appearance in the leaf level of the M-tree.</td>
</tr>
<tr>
<td>G-DisC</td>
<td>Selects each iteration the white object $p$ with the largest white neighborhood.</td>
</tr>
<tr>
<td>L-Gr-G-DisC</td>
<td>One range query per grey node at distance $r$ from $p$.</td>
</tr>
<tr>
<td>L-Wh-G-DisC</td>
<td>One range query per white node at distance $2r$ from $p$.</td>
</tr>
<tr>
<td>G-C</td>
<td>Selects at each iteration the non-black object $p$ with the largest white neighborhood.</td>
</tr>
</tbody>
</table>

We use the Euclidean distance for the synthetic datasets and “Cities”, while for “Cameras”, whose attributes are categorical, we use $\text{dist}(p_i, p_j) = \sum \delta(p_i, p_j)$, where $\delta(p_i, p_j)$ is equal to 1, if $p_i$ and $p_j$ differ in the $i^{th}$ dimension and 0 otherwise, i.e., the Hamming distance. Note that the choice of an appropriate distance metric is an important but orthogonal to our approach issue.

Table 1 summarizes the values of the input parameters used in our experiments and Table 2 summarizes the algorithms employed.

Solution Size. We first compare our various algorithms in terms of the size of the computed diverse subset (Table 3). We consider Basic-DisC, Greedy-DisC (note that, both Grey- Greedy-DisC and White-Greedy-DisC produce the same solution) and Greedy-C. We also tested ”lazy” variations of the greedy algorithm, namely Lazy-Grey-Greedy-DisC with distance $r/2$ and Lazy-White-Greedy-DisC with distance $3r/2$. Grey-Greedy-DisC locates a smaller DisC diverse subset than Basic-DisC in all cases. The lazy variations also perform better than Basic-DisC and comparable with Grey-Greedy-DisC. Lazy-White-Greedy-DisC seems to approximate better the actual size of the white neighborhoods than Grey-Greedy-DisC and in most cases produces smaller subsets. Greedy-C produces subsets with size similar with those produced by Grey-Greedy-DisC. This means that raising the independence assumption does not lead to substantially smaller diverse subsets in our datasets. Finally, note that the diverse subsets produced by all algorithms for the “Clustered” dataset are smaller than for the “Uniform” one, since objects are generally more similar to each other.

Computational Cost. Next, we evaluate the computational cost of our algorithms, measured in terms of tree node accesses. Figure 7 reports this cost, as well as, the cost savings when the pruning rule of Section 5 is employed for Basic-DisC and Greedy-DisC (as previously detailed, this pruning cannot be applied to Greedy-C). Greedy-DisC has higher cost than Basic-DisC. The additional computational cost becomes more significant as the radius increases. The reason for this is that Greedy-DisC performs significantly more range queries. As the radius increases, objects have more neighbors and, thus, more M-tree nodes need to be accessed in order to retrieve them, color them and update the size of the neighborhoods of their neighbors. On the contrary, the cost of Basic-DisC is reduced when the radius increases, since it does not need to update the size of any neighborhood. For larger radii, more objects are colored grey by each selected (black) object and, therefore, less range queries are performed. Both algorithms benefit from pruning (up to 50% for small radii). We also experimented with employing bottom-up rather than top-down range queries. At most cases, the benefit in node accesses was less than 5%.

Figure 8 compares Grey-Greedy-DisC with White-Greedy-DisC and their corresponding “lazy” variations. We see that White-Greedy-DisC performs better than Grey-Greedy-DisC for the clustered dataset as $r$ increases. This is because in this case, grey objects share many common white neighbors which are accessed multiple times by Grey-Greedy-DisC for updating their white neighborhood size and only once by White-Greedy-DisC. The lazy variations reduce the computational cost further.

In the rest of this section, unless otherwise noted, we use the (Grey-)Greedy-DisC (Pruned) heuristic.

Impact of Dataset Cardinality and Dimensionality. For this experiment, we employ the “Clustered” dataset and vary its cardinality from 5000 to 15000 objects and its dimensionality from 2 to 10 dimensions. Figure 9 shows the corresponding solution size and computational cost as computed by the Greedy-DisC heuristic. We observe that the solution size is more sensitive to changes in cardinality when the radius is small. The reason for this is that for large radii, a selected object covers a large area in space. Therefore, even when the cardinality increases and there are many available objects to choose from, these objects are quickly covered by the selected ones. In Figure 9(b), the increase in the computational cost is due to the increase of range queries required to maintain correct information about the size of the white neighborhoods. Increasing the dimensionality of the dataset causes more objects to be selected as diverse as shown in Figure 9(c). This is due to the “curse of dimensionality” effect, since space becomes sparser at higher dimensions. The computational cost may however vary as dimensionality increases, since it is influenced by the cost of computing the neighborhood size of the objects that are colored grey.
to the node with the closest pivot) and, finally, selecting objects to each new node (instead of assigning each object each other resulted in increased fat-factor. Even higher fat-factor is required by employing the “MinOverlap” policy. Selecting as for four different policies. The lowest fat-factor was ac-

the number of these objects, \( h \) the height of the tree and \( m \) the number of nodes in the tree. Ideally, the tree would require accessing one node per level for each point query which yields a fat-factor of zero. The worst tree would visit all nodes for every point query and its fat-factor would be equal to one.

We created various M-trees using different splitting poli-
cies which result in different fat-factors. We present results for four different policies. The lowest fat-factor was ac-
quired by employing the “MinOverlap” policy. Selecting as new pivots the two objects with the greatest distance from each other resulted in increased fat-factor. Even higher fat-
factors were observed when assigning an equal number of objects to each new node (instead of assigning each object to the node with the closest pivot) and, finally, selecting the new pivots randomly produced trees with the highest fat-factor among all policies.

Figure 10 reports our results for our uniform and clustered 2-dimensional datasets with cardinality equal to 10000. For the uniform dataset, we see that a high fat-factor leads to more node accesses being performed for the same solution. This is not the case for the clustered dataset, where objects are gathered in dense areas and thus increasing the fat-factor does not have the same impact as in the uniform case, due to pruning and locality. As the radius of the computed subset becomes very large, the solution size becomes very small, since a single object covers almost the entire dataset, this is why all lines of Figure 10 begin to converge for \( r > 0.5 \).

We also experimented with varying the capacity of the nodes of the M-tree. Trees with smaller capacity require more node accesses since more nodes need to be recovered to locate the same objects; when doubling the node capacity, the computational cost was reduced by almost 45%.

Zooming. In the following, we evaluate the performance of our zooming algorithms. We begin with the zooming-in heuristics. To do this, we first generate solutions with Greedy-DisC for a specific radius \( r \) and then adapt these solutions for radius \( r' \). We use Greedy-DisC because it gives the smallest sized solutions. We compare the results to the solutions generated from scratch by Greedy-DisC for the new radius. The comparison is made in terms of solution size, computational cost and also the relation of the pro-
duced solutions as measured by the Jaccard distance. Given

Impact of M-tree Characteristics. Next, we evaluate how the characteristics of the employed M-trees affect the computational cost of computed DisC diverse subsets. Note that, different tree characteristics do not have an impact on which objects are selected as diverse.

Different degree of overlap among the nodes of an M-tree may affect its efficiency for executing range queries. To quantify such overlap, we employ the fat-factor [25] of a tree \( T \) defined as:

\[
f(T) = \frac{Z - nh}{n} \cdot \frac{1}{m - h}
\]

where \( Z \) denotes the total number of node accesses required to answer point queries for all objects stored in the tree, \( n \) the number of these objects, \( h \) the height of the tree and \( m \) the number of nodes in the tree. Ideally, the tree would require accessing one node per level for each point query which yields a fat-factor of zero. The worst tree would visit all nodes for every point query and its fat-factor would be equal to one.

We created various M-trees using different splitting poli-
cies which result in different fat-factors. We present results for four different policies. The lowest fat-factor was ac-
quired by employing the “MinOverlap” policy. Selecting as new pivots the two objects with the greatest distance from each other resulted in increased fat-factor. Even higher fat-
factors were observed when assigning an equal number of objects to each new node (instead of assigning each object to the node with the closest pivot) and, finally, selecting
two sets $S_1, S_2$, their Jaccard distance is defined as:

$$\text{Jaccard}(S_1, S_2) = 1 - \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

Figure 11 and Figure 12 report the corresponding results for different radii. Due to space limitations, we report results for the “Clustered” and “Cities” datasets. Similar results are obtained for the other datasets as well. Each solution reported for the zooming-in algorithms is adapted from the Greedy-DisC solution for the immediately larger radius and, thus, the x-axis is reversed for clarity; e.g., the zooming solutions for $r = 0.02$ in Figure 11(a) and Figure 12(a) are adapted from the Greedy-DisC solution for $r = 0.03$.

We observe that the zooming-in heuristics provide similar solution sizes with Greedy-DisC in most cases, while their computational cost is smaller, even for Greedy-Zoom-In. More importantly, the Jaccard distance of the adapted solutions for $r'$ to the Greedy-DisC solution for $r$ is much smaller than the corresponding distance of the Greedy-DisC solution for $r'$ (Figure 13). This means that computing a new solution for $r'$ from scratch changes most of the objects returned to the user, while a solution computed by a zooming-in heuristic maintains many common objects in the new solution. Therefore, the new diverse subset is intuitively closer to what the user expects to receive.

Figure 14 and Figure 15 show corresponding results for the zooming-out heuristics. The Greedy-Zoom-Out(c) heuristic achieves the smallest adapted DisC diverse subsets. However, its computational cost is very high and generally exceeds the cost of computing a new solution from scratch. Greedy-Zoom-Out(a) also achieves similar solution sizes with Greedy-Zoom-Out(c), while its computational cost is much lower. The non-greedy heuristic has the lowest computational cost. Again, all the Jaccard distances of the zooming-out heuristics to the previously computed solution are smaller than that of Greedy-DisC (Figure 16), which indicates that a solution computed from scratch has only a few objects in common from the initial DisC diverse set.

7. RELATED WORK

Other Diversity Definitions: Diversity has recently attracted a lot of attention as a means of enhancing user satisfaction [27, 4, 16, 6]. Diverse results have been defined in

Figure 10: Varying M-tree fat-factor.

Figure 11: Solution size for zooming-in.

Figure 12: Node accesses for zooming-in.

Figure 13: Jaccard distance for zooming-in.

Figure 14: Solution size for zooming-out.

Figure 15: Node accesses for zooming-out.

Figure 16: Jaccard distance for zooming-out.
Various ways [10], namely in terms of content (or similarity), novelty and semantic coverage. Similarity definitions (e.g., [31]) interpret diversity as an instance of the p-dispersion problem [13] whose objective is to choose p out of n given points, so that the minimum distance between any pair of chosen points is maximized. Our approach differs in that the size of the diverse subset is not an input parameter. Most current novelty and semantic coverage approaches to diversification (e.g., [9, 3, 29, 14]) rely on associating a diversity score with each object in the result and then either selecting the top-k highest ranked objects or those objects whose score is above some threshold. Such diversity scores are hard to interpret, since they do not depend solely on the object. Instead, the score of each object is relative to which objects precede it in the rank. Our approach is fundamentally different in that we treat the result as a whole and select DisC diverse subsets of it that fully cover it.

Another related work is that of [18] that extends nearest neighbor search to select k neighbors that are not only spatially close to the query object but also differ on a set of predefined attributes above a specific threshold. Our work is different since our goal is not to locate the nearest and most diverse neighbors of a single object but rather to locate an independent and covering subset of the whole dataset.

On a related issue, selecting k representative skyline objects is considered in [22]. Representative objects are selected so that the distance between a non-selected skyline object from its nearest selected object is minimized. Finally, another related method for selecting representative results, besides diversity-based ones, is k-medoids, since medoids can be viewed as representative objects (e.g., [19]). However, medoids may not cover all the available space. Medoids were extended in [5] to include some sense of relevance (priority medoids).

The problem of diversifying continuous data has been recently considered in [12, 21, 20] using a number of variations of the MAXMIN and MAXSUM diversification models.

Results from Graph Theory: The properties of independent and dominating (or covering) subsets have been extensively studied. A number of different variations exist. Among these, the Minimum Independent Dominating Set Problem (which is equivalent to the r-DisC diverse problem) has been shown to have some of the strongest negative approximation results: in the general case, it cannot be approximated in polynomial time with a factor of n^{1−ǫ} for 0 < ǫ < 1 unless P = NP [17]. However, some approximation results have been found for special graph cases, such as bounded degree graphs [7]. In our work, rather than providing polynomial approximation bounds for DisC diversity, we focus on the efficient computation of non-minimum but small DisC diverse subsets. There is a substantial amount of related work in the field of wireless networks research, since a Minimum Connected Dominating Set of wireless nodes can be used as a backbone for the entire network [24]. Allowing the dominating set to be connected has an impact on the complexity of the problem and allows different algorithms to be designed.

8. SUMMARY AND FUTURE WORK

In this paper, we proposed a novel, intuitive definition of diversity as the problem of selecting a minimum representative subset S of a result P, such that each object in P is represented by a similar object in S and that the objects included in S are not similar to each other. Similarity is modeled by a radius r around each object. We call such subsets r-DisC diverse subsets of P. We introduced adaptive diversification through decreasing r, termed zooming-in, and increasing r, called zooming-out. Since locating minimum r-DisC diverse subsets is an NP-hard problem, we introduced heuristics for computing approximate solutions, including incremental ones for zooming, and provided corresponding theoretical bounds. We also presented an efficient implementation based on spatial indexing.

There are many directions for future work. We are currently looking into different ways of integrating relevance with DisC diversity. The first approach is by a "weighted" variation of the DisC subset problem, where each object has an associated weight based on its relevance. Now the goal is to select a DisC subset having the maximum sum of weights. The other approach is to allow multiple radii, so that relevant objects get a smaller radius than the radius of less relevant ones. Other potential future directions include implementations using different data structures and designing algorithms for the online version of the problem.

Acknowledgments

M. Drosou was supported by the ESF and Greek national funds through the NSRF - Research Funding Program: “Heracles II”. E. Pitoura was supported by the project “InterSocial” financed by the European Territorial Cooperation Operational Program “Greece - Italy” 2007-2013, co-funded by the ERDF and national funds of Greece and Italy.

9. REFERENCES

at its center and let \(N^W(u)\) be the number of white nodes in it. If a node in the star is labeled grey by Greedy-C, these nodes are charged some cost. By the greedy condition of the algorithm, this cost can be at most \(1/|N^W(u)|\) per newly covered node. Otherwise, the algorithm would rather have chosen \(u\) for the dominating set because \(u\) would cover at least \(|N^W(u)|\) nodes. In the worst case, no two nodes in the star of \(u\) are covered at the same iteration. In this case, the first node that is labeled grey is charged at most \(1/\delta(u)+1\), the second node is charged at most \(1/\delta(u)\) and so on, where \(\delta(u)\) is the degree of \(u\). Therefore, the total cost of a star is at most:

\[
\frac{1}{\delta(u)} + \frac{1}{\delta(u)^2} + \ldots + \frac{1}{\delta(u)^{\alpha-1}} = H(\delta(u)^{\alpha-1}) \leq H(\Delta+1) \approx \ln \Delta
\]

where \(H(i)\) is the \(i\)th harmonic number. Since a minimum dominating set is equal or smaller than a minimum independent dominating set, the theorem holds.

**Proof of Lemma 3:** For the proof, we use a technique for partitioning the annulus between \(r_1\) and \(r_2\) similar to the one in [23] and [28]. Let \(r_1\) be the radius of an object \(p\) (Figure 17(b)) and \(\alpha\) a real number with \(0 < \alpha < \frac{\pi}{2}\). We draw circles around the object \(p\) with radii \((2\cos\alpha)r\), \((2\cos\alpha)^{r+1}, (2\cos\alpha)^{r+2}, \ldots, (2\cos\alpha)^{\gamma-1}, (2\cos\alpha)^\gamma\), such that \((2\cos\alpha)^{\gamma} \leq r_1\) and \((2\cos\alpha)^{r+1} > r_2\) and \((2\cos\alpha)^{\gamma-1} < r_2\) and \((2\cos\alpha)^\gamma \geq r_2\). It holds that \(x_p = \ln r_1 / \ln (2\cos\alpha)\) and \(y_p = \left[ \frac{\ln r_2}{\ln (2\cos\alpha)} \right].\) In this way, the area around \(p\) is partitioned into \(x_p - y_p\) annuli plus the \(r_1\)-disk around \(p\).

Consider an annulus \(A\). Let \(r_1\) and \(r_2\) be two neighbors of \(p\) in \(A\) with \(dist(p_1, p_2) > r_1\). Then, it must hold that \(|p_1| |p_2| > \alpha\). To see this, we draw two segments from \(p\) crossing the inner and outer circles of \(A\) at \(a, b, c, d\) such that \(p_1\) resides in \(pb\) and \(\angle bpd = \alpha\), as shown in the figure. Due to the construction of the circles, it holds that \(\angle bpd = \alpha\) and, therefore, it holds that \(|bp| = |pd| = \alpha\). Therefore, for any object \(p_3\) in the area \(abc\) of \(A\), it holds that \(|p_3| > |p_2|\) which means that all objects in that area are neighbors of \(p_1\), i.e., at distance less or equal to \(r_1\). For this reason, \(p_2\) must reside outside this area which means that \(|p_1| |p_2| > \alpha\). Based on this, we see that there exist at most \(\frac{\pi}{4} - 1\) independent (for \(r_1\)) nodes in \(A\).

The same holds for all annuli. Therefore, we have at most \(\gamma\) \(\left( 2\cos\alpha - 1 \right)\) independent nodes in the annuli. For \(0 < \alpha < \frac{\pi}{4}\), this has a minimum when \(\alpha\) is close to \(\frac{\pi}{5}\) and that minimum value is \(9 \left\lceil \ln (\gamma r_2 / r_1) \right\rceil = 9 \left\lceil \log_{10} (r_2 / r_1) \right\rceil\), where \(\beta = 1 + \frac{\sqrt{5}}{2}\).

**Proof of Lemma 4(ii):** Let \(r_1\) be the radius of an object \(p\). We draw Manhattan circles around the object \(p\) with radii \(r_1, 2r_1, \ldots\) until the radius \(r_2\) is reached. In this way, the area around \(p\) is partitioned into \(\gamma = \left( \frac{r_2 - r_1}{r_1} \right)\) Manhattan annuli plus the \(r_1\)-Manhattan-disk around \(p\).

Consider an annulus \(A\). The objects shown in Figure 17(c) cover the whole annulus and their Manhattan pairwise distances are all greater or equal to \(r_1\). Assume that the annulus spans among distance \(ir_1\) and \((i+1)r_1\) from \(p\), where \(i\) is an integer with \(i > 1\). Then, \(|ab| = \sqrt{2(r_1 + r_1/2)^2}\). Also, for two objects \(p_1, p_2\) it holds that \(|p_1| |p_2| = \sqrt{2(r_1/2)^2}\). Therefore, at one quadrant of the annulus there are \(|ab| = 2i + 1\) independent neighbors which means that there are \(4(2i + 1)\) independent neighbors in \(A\). Therefore, there are in total \(\sum_{i=1}^{\gamma} 4(2i + 1)\) independent neighbors (for \(r_1\)) neighbors of \(p\).