Υποστήριξη Διαβάθμισης με Βάση Προτιμήσεις και Διαφορετικότητα σε Δίκτυο-Κεντρικά Συστήματα Διαχείρισης Δεδομένων

Η ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης

του Τμήματος Μηχανικών Η/Υ και Πληροφορικής Εξεταστική Επιτροπή

από την

Μαρίνα Δρόσου

ως μέρος των Υποχρεώσεων για τη λήψη του

ΔΙΔΑΚΤΟΡΙΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

Οκτώβριος 2013

Τριμελής Συμβουλευτική Επιτροπή

- Ευαγγελία Πιτουρά, Καθηγήτρια του Τμήματος Μηχανικών Η/Υ και Πληροφορικής του Πανεπιστημίου Ιωαννίνων (Επιβλέπουσα)
- Παναγιώτης Τριανταφύλλου, Καθηγητής του Τμήματος Επιστήμης Υπολογιστών του University of Glasgow (Ηνωμένο Βασίλειο)
- Πάνος Κ. Χρυσάνθης, Καθηγητής του Τμήματος Επιστήμης Υπολογιστών του University of Pittsburgh (ΗΠΑ)

Επταμελής Εξεταστική Επιτροπή

- Ευαγγελία Πιτουρά, Καθηγήτρια του Τμήματος Μηχανικών Η/Υ και Πληροφορικής του Πανεπιστημίου Ιωαννίνων (Επιβλέπουσα)
- Παναγιώτης Βασιλειάδης, Επίκουρος Καθηγητής του Τμήματος Μηχανικών Η/Υ και Πληροφορικής του Πανεπιστημίου Ιωαννίνων
- Αριστείδης Λύκας, Καθηγητής του Τμήματος Μηχανικών Η/Υ και Πληροφορικής του Πανεπιστημίου Ιωαννίνων
- Νικόλαος Μαμουλής, Καθηγητής του Τμήματος Επιστήμης Υπολογιστών του University of Hong Kong (Hong Kong)
- Παναγιώτης Τριανταφύλλου, Καθηγητής του Τμήματος Επιστήμης Υπολογιστών του University of Glasgow (Ηνωμένο Βασίλειο)
- Παναγιώτης Τσαπάρας, Επίκουρος Καθηγητής του Τμήματος Μηχανικών Η/Υ και Πληροφορικής του Πανεπιστημίου Ιωαννίνων
- Πάνος Κ. Χρυσάνθης, Καθηγητής του Τμήματος Επιστήμης Υπολογιστών του University of Pittsburgh (ΗΠΑ)

"All have their worth and each contributes to the worth of the others."
J.R.R. Tolkien, The Silmarillion

DEDICATION

To my family

Ευχαριστιες

Φτάνοντας στο τέλος αυτού του ταξιδιού, βρίσκω τον εαυτό μου να έχει αποκομίσει ένα πλήθος από οφέλη, πολύ περισσότερα από την αναμφισβήτητη κατάκτηση της γνώσης που συνοδεύει αυτού του είδους τα ερευνητικά ταξίδια. Η συμβολή του συνόλου των μεταπτυχιακών σπουδών μου στη διαμόρφωση του χαρακτήρα μου πιστεύω υπήρξε καθοριστική. Η έρευνα με δίδαξε επιμονή, υπομονή, αποφασιστικότητα και διορατικότητα. Επίσης, πέρα από το μαγευτικό ταξίδι στη γνώση, μου χάρισε απλόχερα και ένα πλήθος ταξιδιών, εμπειριών και γνωριμιών σχεδόν σε όλα τα σημεία του πλανήτη μας, κάτι για το οποίο αισθάνομαι ιδιαίτερα τυχερή και ευτυχής. Και τώρα, νιώθω την ανάγκη να ευχαριστήσω πολλούς συνεργάτες, αλλά και φίλους, για τη στήριξη και τη συμβολή τους στην ολοκλήρωση της διδακτορικής αυτής διατριβής.

Θα ήθελα, πρωτίστως, να ευχαριστήσω την Καθηγήτρια κ. Ευαγγελία Πιτουρά, επιβλέπουσα του συνόλου της ερευνητικής μου πορείας ως την ολοκλήρωση αυτής της διατριβής. Η πολυετής συνεργασία μας συνέβαλε καθοριστικά σε ένα πλήθος γνώσεων και δεξιοτήτων που νιώθω ότι κατέκτησα. Την ευχαριστώ τόσο για την ουσιαστική ερευνητική συνεργασία όσο και για το συνολικό ακαδημαϊκό ήθος το οποίο μου μετέδωσε.

Θα ήθελα, επίσης, να ευχαριστήσω τους συνεργάτες -αλλά κυρίως φίλους- στο Εργαστήριο Κατανεμημένης Διαχείρισης και Επεξεργασίας Δεδομένων του Τμήματος Μηχανικών Η/Υ και Πληροφορικής του Πανεπιστημίου Ιωαννίνων. Πρώτα από όλους, τον Δρ. Κώστα Στεφανίδη, με τον οποίο είχαμε μία άριστη συνεργασία, κυρίως τα πρώτα χρόνια της ερευνητικής μου δραστηριότητας, και ο οποίος με καθοδήγησε στα πρώτα βήματα της ερευνητικής μου πορείας. Επίσης, την κ. Ευτυχία Κωλέτσου, η οποία πάντα ενδυναμώνει την πίστη μου στους ανθρώπους και τις δυνατότητές τους, καθώς και τον κ. Δημήτρη Σουραβλιά.

Δε θα μπορούσα να παραλείψω τον Λέκτορα του University of Queensland της Αυστραλίας Δρ. Mohamed Sharaf για την ευκαιρία που μου πρόσφερε να μεταβώ και να εργαστώ ένα διάστημα στο εργαστήριό του. Η εμπειρία αυτή ήταν επικερδής σε ένα πλήθος διαφορετικών επιπέδων.

Θα ήθελα να αναφερθώ ξεχωριστά σε όλους τους ανθρώπους του Τμήματος Πληροφορικής (πλέον Μηχανικών Η/Υ και Πληροφορικής) του Πανεπιστημίου Ιωαννίνων οι οποίοι συνέβαλαν με τον έναν ή με τον άλλο τρόπο στην ολοκλήρωση αυτής της διατριβής.

Ιδιαίτερα τους φίλους και συμφοιτητές Μυρτώ, Γεωργία, Κατερίνα, Ευτυχία, Γιώργο και Πέτρο.

Ευχαριστώ επίσης τα μέλη της επιτροπής μου, κκ. Πάνο Χρυσάνθη, Peter Τριανταφύλλου, Παναγιώτη Βασιλειάδη, Παναγιώτη Τσαπάρα, Αριστείδη Λύκα και Νίκο Μαμουλή, για την τιμή που μου κάνουν να συμμετέχουν σε αυτή.

Ένα μεγάλο ευχαριστώ στους γονείς μου, Κώστα και Ντίνα, όχι μόνο για τη στήριξή τους καθόλη τη διάρκεια των σπουδών μου, αλλά και για τον τρόπο που με ανέθρεψαν, ώστε να πιστεύω στον εαυτό μου και να επιμένω μπροστά σε κάθε δυσκολία. Νιώθω τυχερή που γίνομαι καθημερινά αποδέκτης της υπέρμετρης αγάπης τους.

Τέλος, ένα τεράστιο ευχαριστώ στο σύντροφό μου Γρηγόρη, ο οποίος βρίσκεται δίπλα μου και με στηρίζει σε κάθε μου βήμα. Αυτή η διαδρομή δε θα ήταν στο ελάχιστο τόσο ενδιαφέρουσα εάν δεν ήταν κομμάτι της.

Ιωάννινα, Οκτώβριος 2013
Μαρίνα Δρόσου

CONTENTS

LIST OF FIGURES

List of Tables

Algorithm Index

ABSTRACT

Marina K. Drosou. PhD, Computer Science and Engineering Department, University of Ioannina, Greece. October, 2013. *Relevance and Diversity-based Ranking in Network-Centric Information Management Systems.* Thesis Supervisor: Evaggelia Pitoura.

With the explosion of the amount of information currently available online, locating valuable or important information can prove out to be an overwhelming task. This abundance of accessible information creates the need for developing methods towards selecting and presenting to users representative subsets. Various ranking techniques have been developed in the past, to allow users to quickly access what is most useful to them. Ranking of information is usually based on some notion of *relevance* of each specific piece of information, or *item*, to the user needs. Ranking based solely on relevance, however, may lead to enhancing the over-specialization problem, i.e., the retrieval of too homogeneous results for a user query. For this reason, retrieving *diverse* results, i.e., items that are different to each other, has recently attracted great attention as a means to complement relevance-based ranking and increase the quality of results retrieved by information systems.

Generally, the problem of diversification is defined as follows. Given a set $\mathcal{P}$ of items and a budget k, select a subset S of $\mathcal{P}$, with $|S| = k$, such that, S maximizes some diversification objective. Among the many different objectives proposed in the past for selecting diverse items, one can find approaches based on (i) *content* (or *similarity*), i.e., selecting items that are dissimilar to each other, (ii) *novelty*, i.e., selecting items that contain new information when compared to previously seen ones, and (iii) *coverage*, i.e., selecting items that belong to different categories. Selecting diverse items has been shown to be an NP-hard problem.

This PhD thesis concerns the development, implementation and evaluation of models, algorithms and techniques for the ranking of information being presented to users of network-centric information management systems. This ranking is based on the *importance* of each piece of information. We consider that importance is influenced by *both* relevance to user information needs *and* diversity. Relevance is important so that users are only presented with the most useful results according to their needs, while diversity ensures that the received results do not all contain similar information.

We focus on two different axes: (i) diversifying dynamic data and (ii) diversifying

data based on dissimilarity and coverage. In addition to this, we also develop a system prototype, called Poikilo (from the Greek "ποικίλο", meaning "diverse") for evaluating the results of various diversification models and algorithms.

Most previous research considers the *static* version of the problem, i.e., the available items out of which a diverse set is selected do not change over time. In this thesis, we focus on the *dynamic* diversification problem, where insertions and deletions of items are allowed and the diverse set needs to be refreshed to reflect such updates. We propose an index-based approach using a spatial-index structure, namely the Cover Tree. Cover trees are data structures originally proposed for approximate nearest-neighbor search. Motivated by popular proactive delivery paradigms, such as news alerts, RSS feeds and notification services in social networks, where users specify their interests and receive relevant notifications, we also consider the *continuous* version of the problem, where diversified sets are computed over streams of items. To avoid overwhelming users by forwarding to them all relevant items, we consider the case in which a representative diverse set is computed, instead, whose size can be configured by the users. We introduce a sliding window model along with *continuity* requirements. Such requirements ensure that the order in which the diverse items are delivered follows the order of their generation and that an item does not appear, disappear and then re-appear in the diverse set.

We introduce a suite of algorithms that exploit the cover tree to provide solutions with varying accuracy and complexity and also provide theoretical results that bound the accuracy of the solutions achieved with regards to the optimal solution. We extend our approach to select items that are both relevant and diverse. We consider two different approaches. The first one considers the relevance of the items when inserting them into the index, while the second one combines relevance with diversity when selecting items from the index. We perform an extensive experimental evaluation of the efficiency and effectiveness of our approach using both real and synthetic datasets.

We also address diversity through a different perspective. Let $\mathcal{P}$ be the set of available items and r a positive real number, which we call *radius*. Let also d be some distance metric. For p_i, p_j in $\mathcal{P}$, we consider that p_i is similar to p_j if and only if $d(p_i, p_j) \leq r$. We also say that p_j is *covered* by p_i. Given $\mathcal{P}$, we select a representative subset S, $S \subseteq \mathcal{P}$, to be presented to the user such that: (i) all items in $\mathcal{P}$ are covered by at least one item in S and (ii) no two items in S cover each other. The first condition ensures that all items in $\mathcal{P}$ are represented by at least one similar item in the selected subset. The second condition ensures that the selected items of $\mathcal{P}$ are dissimilar. We call S *r-Dissimilar and Covering* subset or *r-DisC* diverse subset.

In contrary to previous approaches to diversification, instead of specifying a required size k of the diverse set, our tuning parameter r explicitly expresses the degree of diversification and determines the size of the diverse subset. 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. To retrieve a concise representation of all items, we aim at selecting the DisC diverse subset containing the smallest number of items. We also define *weighted* DisC diverse subsets, where each item is associated with a weight indicating its relevance and *multiple radii* DisC diverse subsets, where each item is associated with a different radius. Multiple radii are used to allow different areas of the data to contribute more or less items to the selected diverse subset.

We formalize the problem of locating minimum and minimum weighted DisC diverse subsets as an *independent dominating* set problem on graphs. We provide efficient algorithms for locating approximate solutions as well as corresponding theoretical approximation bounds. We propose efficient implementations of our algorithms based on spatial index structures. In particular, we use the M-tree. We compare DisC diversity with other popular diversity models, both analytically and qualitatively, and provide an extensive experimental evaluation of various aspects of our approach.

Finally, we present a system prototype, called Poikilo, to assist users in locating, visualizing and comparing diverse items based on a suite of different diversification models and algorithms. We provide implementations of a wide variety of diversification approaches.

Περιληψη

Μαρίνα Δρόσου του Κωνσταντίνου και της Κωνσταντίνας. PhD, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Οκτώβριος, 2013. *Υποστήριξη Διαβάθμισης με Βάση Προτιμήσεις και Διαφορετικότητα σε Δίκτυο-Κεντρικά Συστήματα Διαχείρισης Δεδομένων.* Επιβλέπουσα: Ευαγγελία Πιτουρά.

Ο όγκος της πληροφορίας που γίνεται καθημερινά διαθέσιμος στους χρήστες διαδικτυακών συστημάτων είναι τεράστιος. Ο εντοπισμός χρήσιμης πληροφορίας μέσα σε αυτόν τον όγκο δεδομένων μπορεί να αποδειχθεί εξαιρετικά δύσκολος. Για τον λόγο αυτό, διάφορες τεχνικές διαβάθμισης πληροφορίας έχουν προταθεί κατά καιρούς, οι οποίες στοχεύουν στη διευκόλυνση των χρηστών κατά την αναζήτηση πληροφορίας. Η διαβάθμιση της πληροφορίας είναι συνήθως βασισμένη σε κάποια έννοια *συνάφειας* (relevance) ως προς το ερώτημα που έχει θέσει ο χρήστης. Ωστόσο, η διαβάθμιση με βάση αποκλειστικά τη συνάφεια μπορεί να ενισχύσει το πρόβλημα της υπερ-εξειδίκευσης, δηλαδή την ανάκτηση αποτελεσμάτων που είναι μεν σχετικά το καθένα με το ερώτημα του χρήστη αλλά είναι πολύ όμοια μεταξύ τους.

Η *ποικιλομορφία (diversity)* των δεδομένων έχει αναδειχθεί τα τελευταία χρόνια ως ένας τρόπος αντιμετώπισης του προβλήματος της υπερ-εξειδίκευσης. Πέραν αυτού, πολλές φορές, οι χρήστες θέτουν ερωτήματα με μία διάθεση εξερεύνησης, δηλαδή ενδιαφέρονται να ανακτήσουν αποτελέσματα τα οποία να καλύπτουν διαφορετικές οπτικές γωνίες του ερωτήματός τους. Φανταστείτε για παράδειγμα έναν χρήστη που ενδιαφέρεται για την αγορά ενός ακινήτου και θέτει ένα αντίστοιχο ερώτημα σε ένα πληροφοριακό σύστημα. Εν γένει, ένα σύνολο ποικίλων αποτελεσμάτων (π.χ. ακίνητα σε διαφορετικές συνοικίες, με διαφορετικό αριθμό υπνοδωματίων κτλ.) περιέχει πιο χρήσιμη πληροφορία από ένα σύνολο αποτελεσμάτων με παρόμοια χαρακτηριστικά. Η αύξηση της ποικιλομορφίας των αποτελεσμάτων δρα συμπληρωματικά με τη συνάφειά τους για τη βελτίωση της ποιότητας του αποτελέσματος που παρουσιάζεται στον χρήστη.

Γενικά, το πρόβλημα της επιλογής ποικιλόμορφων (diverse) αποτελεσμάτων ορίζεται ως εξής: Δοσμένου ενός συνόλου $\mathcal{P}$ αποτελεσμάτων, σκοπός είναι να βρούμε ένα υποσύνολο S του $\mathcal{P}$ τέτοιο ώστε να μεγιστοποιείται η ποικιλομορφία των επιλεγμένων αποτελεσμάτων, σύμφωνα με κάποιο κριτήριο ποικιλομορφίας.

Στόχος αυτής της διατριβής είναι η ανάπτυξη, υλοποίηση και αξιολόγηση μοντέλων, αλγορίθμων και τεχνικών για την υποστήριξη διαβάθμισης με βάση τόσο τη συνάφεια όσο

και την ποικιλομορφία των αποτελεσμάτων σε δίκτυο-κεντρικά συστήματα διαχείρισης πληροφορίας. Επικεντρώνουμε το ενδιαφέρον μας κυρίως πάνω σε δύο άξονες: (i) την ποικιλομορφία πληροφορίας που αλλάζει δυναμικά στο χρόνο και (ii) την ποικιλομορφία πληροφορίας με βάση την ανομοιότητα (dissimilarity) και την κάλυψη (coverage).

Παρόλο το ενδιαφέρον της ερευνητικής κοινότητας για την αύξηση της ποικιλομορφίας, η προηγούμενη έρευνα αφορά κυρίως τη στατική έκδοση του προβλήματος, όπου η διαθέσιμη πληροφορία μένει αμετάβλητη στο χρόνο. Στα πλαίσια αυτής της διατριβής, ασχολούμαστε με τη δυναμική έκδοση του προβλήματος, όπου έχουμε δυναμικές εισαγωγές και διαγραφές δεδομένων οι οποίες πρέπει να αντικατοπτρίζονται στα ποικιλόμορφα αποτελέσματα που παρουσιάζονται στον χρήστη.

Ακολουθούμε μία προσέγγιση βασισμένη σε χωρικές δομές ευρετηρίασης, και συγκεκριμένα, σε δέντρα επικάλυψης (cover trees). Επικεντρωνόμαστε σε ένα κλασσικό πρόβλημα ποικιλομορφίας (MaxMin), το οποίο, δοσμένης μιας μετρικής απόστασης, στοχεύει στη μεγιστοποίηση της ελάχιστης απόστασης μεταξύ δύο αποτελεσμάτων. Το πρόβλημα αυτό έχει αποδειχθεί πως ανήκει στην κλάση των NP-hard προβλημάτων. Προτείνουμε ένα σύνολο αλγορίθμων, διαφορετικής ακρίβειας και πολυπλοκότητας, οι οποίοι εκμεταλλεύονται τις χωρικές ιδιότητες των δέντρων επικάλυψης για την αποδοτική επίλυση του προβλήματος. Αποδεικνύουμε θεωρητικά αποτελέσματα που φράσσουν την ακρίβεια των λύσεών μας σε σχέση με τη βέλτιστη λύση. Τέλος, επεκτείνουμε τους αλγορίθμους μας ώστε να συνυπολογίζουν και τη συνάφεια των αποτελεσμάτων κατά την επιλογή τους.

Η προηγούμενη έρευνα θεωρεί κυρίως το μέγεθος k του ποικιλόμορφου συνόλου S ως παράμετρο του προβλήματος και στοχεύει στην επιλογή των k αποτελεσμάτων που μεγιστοποιούν κάποιο κριτήριο ποικιλομορφίας. Στα πλαίσια αυτής της διατριβής, εξετάζουμε την ποικιλομορφία από μία νέα οπτική γωνία. Έστω $\mathcal{P}$ ένα σύνολο αποτελεσμάτων και r ένας θετικός πραγματικός αριθμός, τον οποίο καλούμε ακτίνα. Έστω ακόμα μία μετρική απόστασης d. Θεωρούμε πως δύο αποτελέσματα p_1, p_2 του $\mathcal{P}$ είναι *παρόμοια* αν και μόνο αν $d(p1, p2) \leq r$. Σε αυτήν την περίπτωση, λέμε επίσης πως το p_1 *καλύπτει* το p_2.

Δοσμένου του $\mathcal{P}$, θέλουμε να εντοπίσουμε ένα αντιπροσωπευτικό, ποικίλο υποσύνολο S του $\mathcal{P}$, τέτοιο ώστε (i) όλα τα αποτελέσματα του $\mathcal{P}$ να καλύπτονται από τουλάχιστον ένα αποτέλεσμα του S και (ii) όλα τα αποτελέσματα του S να είναι ανόμοια μεταξύ τους. Η πρώτη συνθήκη εξασφαλίζει ότι όλα τα αποτελέσματα εκπροσωπούνται από κάποιο παρόμοιο αποτέλεσμα στο επιλεγμένο υποσύνολο, ενώ η δεύτερη συνθήκη εξασφαλίζει ότι δεν υπάρχουν παρόμοια αποτελέσματα μεταξύ των επιλεγμένων. Καλούμε το υποσύνολο S r-Dissimilar-and-Covering (r-DisC) υποσύνολο και στοχεύουμε στην ανάκτηση ενός ελάχιστου r-DisC υποσυνόλου.

Σε αντίθεση με τις προηγούμενες προσεγγίσεις, αντί να ορίζουμε εξαρχής ως είσοδο στο πρόβλημά μας το μέγεθος k του υποσυνόλου S, η παράμετρός μας r προσδιορίζει τον απαιτούμενο βαθμό ποικιλομορφίας και εμμέσως καθορίζει το μέγεθος του S. Μεγάλη ακτίνα οδηγεί σε μικρότερα αλλά περισσότερο ποικίλα υποσύνολα, ενώ μικρή ακτίνα το αντίθετο.

Ασχολούμαστε επίσης με την περίπτωση στην οποία σε κάθε αποτέλεσμα αντιστοιχεί μία διαφορετική ακτίνα -μία τέτοια προσέγγιση είναι χρήσιμη όταν θέλουμε να δώσουμε διαφορετική έμφαση σε διαφορετικά αποτελέσματα- καθώς και με την περίπτωση στην οποία κάθε αποτέλεσμα φέρει έναν βαθμό συνάφειας.

Μοντελοποιούμε το πρόβλημα εντοπισμού DisC υποσυνόλων για μία κοινή ή για πολλαπλές ακτίνες ως ένα πρόβλημα εντοπισμού συνόλων ανεξαρτησίας και κάλυψης (independent dominating set) σε μη-κατευθυνόμενους και κατευθυνόμενους γράφους αντίστοιχα. Δείχνουμε ότι το πρόβλημα ανήκει στην κλάση των NP-hard προβλημάτων και προτείνουμε ένα σύνολο ευρετικών αλγορίθμων για την επίλυσή του. Παρέχουμε φράγματα για το μέγεθος των λύσεών μας σε σχέση με την βέλτιστη λύση.

Ορίζουμε τις πράξεις zooming-out και zooming-in για την προσαρμογή ενός συνόλου S σε κάποια μεγαλύτερη ή μικρότερη από την αρχική του ακτίνα αντίστοιχα. Μελετούμε τη σχέση μεταξύ του αρχικού και του τελικού υποσυνόλου. Παρέχουμε φράγματα για τη σχέση μεγέθους τους και προτείνουμε αλγορίθμους για την αυξητική μετατροπή του S.

Παρέχουμε αποδοτικές υλοποιήσεις των αλγορίθμων μας βασισμένες σε χωρικές δομές ευρετηρίασης, και συγκεκριμένα σε Μ-δέντρα (M-trees), εκμεταλλευόμενοι την ταχύτητα αυτών των δομών για τη γρήγορη ανάκτηση παρόμοιων αποτελεσμάτων.

Τέλος, αναπτύξαμε ένα ολοκληρωμένο σύστημα, το οποίο καλούμε «ΠΟΙΚΙΛΟ», για την αξιολόγηση και την οπτικοποίηση των αποτελεσμάτων που λαμβάνουμε από ένα πλήθος αλγορίθμων αύξησης της ποικιλομορφίας που έχουν κατά καιρούς προταθεί στη σχετική βιβλιογραφία.

Chapter 1

Introduction

Nowadays, a great volume of information becomes available to users every day from a number of on-line sources. Locating valuable or important information can prove out to be an overwhelming task, due to the great amount of accessible data. For this reason, various *ranking* techniques have been developed, to allow users to quickly access what is most useful to them. Ranking of information is usually based on some notion of *relevance* of each specific piece of information to the user needs. Ranking based solely on relevance, however, may lead to enhancing the *over-specialization problem*, i.e., the retrieval of too homogeneous results, or *items*, for a user query. Beside this, many user searches are of an exploratory nature, in the sense that users are interested in retrieving pieces of information that cover many aspects of their information needs. *Diversification* has recently attracted great attention as a means to complement relevance-based ranking and provide information systems with the means to retrieve more satisfying results (e.g., [88]). The aim of diversification is to retrieve results that are different to each other.

This PhD thesis concerns the development, implementation and evaluation of models, algorithms and techniques for the ranking of information being presented to users of network-centric information management systems. This ranking is based on the *importance* of each piece of information. We consider that importance is influenced by both relevance to user information needs and diversity. Relevance is important so that users are only presented with the most useful results according to their needs, while diversity ensures that the received results do not all contain similar information.

1.1 Overview

Many different approaches have been proposed in the past for selecting diverse items (e.g., [48, 57, 109, 113]). Most of these can be classified in three different categories [42], namely in terms of:

1. *content* (or *similarity*), i.e., selecting items that are dissimilar to each other (e.g., [114, 107]),
2. *novelty*, i.e., selecting items that contain new information when compared to previously seen ones (e.g., [35, 115]), and
3. *coverage*, i.e., selecting items that belong to different categories (e.g., [11, 104]).

Content-based definitions interpret diversity as an instance of the *p-dispersion problem*, whose objective is to select p out of n given items, so that the minimum distance between any pair of selected items is maximized [48]. The p-dispersion problem has been studied in the field of operations research for locating facilities that should be dispersed; such as franchises belonging to a chain or nuclear power plants. Formally, let $\mathcal{P} = \{p_1, \ldots, p_n\}$ be a set of n items. Given a distance metric $d : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^+$ indicating the dissimilarity of two items in $\mathcal{P}$, assume that the *diversity* of a set S, $S \subseteq \mathcal{P}$, is measured by a function $f : 2^{|\mathcal{P}|} \times d \rightarrow \mathbb{R}^+$. For a positive integer k, $k \leq n$, the content-based diversification problem is the problem of selecting a subset S^* of $\mathcal{P}$ such that:

$$S^* = \underset{\substack{S \subseteq \mathcal{P} \\ |S| = k}}{\operatorname{argmax}} f(S, d). \tag{1.1}$$

The choice of f affects the selection of items, even for a specific distance metric d. Two widely used functions are the *minimum* distance among the selected items and the *sum* of the distances of the selected items, formally defined as:

$$f_{\text{MIN}}(S, d) = \min_{\substack{p_i, p_j \in S \\ p_i \neq p_j}} d(p_i, p_j) \tag{1.2}$$

and

$$f_{\text{SUM}}(S, d) = \sum_{p_i, p_j \in S} d(p_i, p_j) \tag{1.3}$$

The corresponding problems are called MAXMIN and MAXSUM diversification problems. Intuitively, MAXMIN aims at discouraging the selection of nearby items, while MAXSUM at increasing the average pairwise distance among all items.

Novelty is a notion closely related to that of diversity, in the sense that items which are diverse from all items seen in the past are likely to contain novel information, i.e., information not seen before. A distinction between novelty and diversity is made in [35], where novelty is viewed as the need to avoid redundancy, whereas diversity is viewed as the need to resolve ambiguity.

Finally, some works view diversity in a different way, that of selecting items that cover many different interpretations of the user's information need. For example, [11] considers typical web search and, given a taxonomy of independent information categories, aims at retrieving items that cover many interpretations of the user query, especially interpretations that are considered important.

Given a set of items $\mathcal{P}$, locating an optimal diverse subset of $\mathcal{P}$ is an NP-hard problem for all these definitions (e.g., [47]). For this reason, various heuristics have been proposed for locating approximate solutions. Most of the heuristics in the related literature can be classified into two main groups, namely (i) *greedy* (e.g., [119]) and (ii) *interchange* (e.g., [113]) algorithms.

Greedy algorithms are the ones most commonly used since they are intuitive and fast. Such algorithms generally make use of two sets to locate a diverse subset of size k: the set $\mathcal{P}$ of the available n items and the set S which contains the selected, i.e., diverse, items. Items are iteratively moved from $\mathcal{P}$ to S and vice versa until $|S| = k$ and $|\mathcal{P}| = n - k$. In most approaches, S is initialized with some item, e.g., the most relevant one, and then items are moved one-by-one from $\mathcal{P}$ to S until k of them have been selected. Many greedy algorithms have been proved to provide $1/2$-approximations of the optimal solution.

Interchange algorithms have also been widely used in the literature for solving the diversification problem. Such algorithms are generally initialized with a random solution S and then iteratively attempt to improve that solution by interchanging an item in the solution with another item that is not in the solution.

Another line of research aims at selecting diverse results similarly to top-k results by employing some sort of threshold algorithm, often attempting to incorporate weights to this threshold (e.g., [87, 22]). This approach is more common in novelty-based definitions of diversity in information retrieval (e.g., [35, 115]). There is a crucial difference between the two problems, however, in that the diversity of a single item cannot be computed independently from that of other items as in the top-k case, since all diversity measures require comparing the item with any previously selected ones.

1.2 Thesis Contribution

In this thesis, we mainly focus on two different axes: (i) diversifying dynamic data and (ii) diversifying data based on dissimilarity and coverage. We also developed a system prototype, called POIKILO (from the Greek "ποικίλο", meaning "diverse") for evaluating the results of various diversification models and algorithms. We next present our contribution in each field.

Diverse Set Selection over Dynamic Data. Despite the considerable interest in diversification, most previous research considers the *static* version of the problem, i.e., the available items out of which a diverse set is selected do not change over time. Here, we focus on the *dynamic* diversification problem, where insertions and deletions of items are allowed and the diverse set needs to be refreshed to reflect such updates. In the past, we addressed the dynamic problem [41] using a sliding-window model. Our approach is based on keeping any non-expired diverse items in every new window and using them to initialize a greedy algorithm. Besides reducing the computational cost, this approach often performs better than executing the greedy algorithm from scratch for each window, since an already diversified subset of items is used as a seed. The dynamic problem was also addressed in [84], using an interchange heuristic instead. Here, we propose an index-based approach.

Our solution is based on *Cover Trees*. Cover trees are data structures originally proposed for approximate nearest-neighbor search [16]. Motivated by popular proactive delivery paradigms, such as news alerts, RSS feeds and notification services in social networks, where users specify their interests and receive relevant notifications, we also consider the *continuous* version of the problem, where diversified sets are computed over streams of items. To avoid overwhelming users by forwarding to them all relevant items, we consider the case in which a representative diverse set is computed, instead, whose size can be configured by the users. We introduce a sliding window model along with *continuity* requirements. Such requirements ensure that the order in which the diverse items are delivered follows the order of their generation and that an item does not appear, disappear and then re-appear in the diverse set.

We focus on the MaxMin diversity problem. We propose a suite of algorithms that exploit the cover tree to provide solutions with varying accuracy and complexity. We provide theoretical results that bound the accuracy of the solutions achieved with regards to the optimal solution.

In a nutshell, this thesis makes the following contributions:

- we propose indexing based on cover trees to address the dynamic diversification problem along with continuity requirements appropriate for a streaming scenario,
- we present a suite of methods with varying complexity that exploit the cover tree for the MaxMin problem and provide bounds for the achieved diversity with regards to the optimal solution,
- we extend the cover tree and our algorithms for selecting items that are both relevant and diverse, and
- we experimentally evaluate the efficiency and effectiveness of our approach using both real and synthetic datasets.

DisC Diversity: Result diversification based on Dissimilarity and Coverage. We also address diversity through a different perspective. Let $\mathcal{P}$ be the set of items in a query result and r a positive real number, which we call *radius*. Let also d be some distance function. For p_i, p_j in $\mathcal{P}$, we consider that p_i is similar to p_j if and only if

$d(p_i, p_j) \leq r$. We also say that p_j is *covered* by p_i. Given $\mathcal{P}$, we select a representative subset S, $S \subseteq \mathcal{P}$, to be presented to the user such that: (i) all items in $\mathcal{P}$ are covered by at least one item in S and (ii) no two items in S cover each other. The first condition ensures that all items in $\mathcal{P}$ are represented by at least one similar item in the selected subset. The second condition ensures that the selected items of $\mathcal{P}$ are dissimilar. We call the set S *r-Dissimilar and Covering* subset or *r-DisC* diverse subset.

In contrary to previous approaches to diversification, we aim at computing subsets of items that contain items 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.

We aim at retrieving a concise representation of all results. For this reason, among all DisC diverse subsets that answer the user query, we aim at selecting the one containing the smallest number of items. In case items are also associated with weights, we also take them into consideration when selecting our diverse items. We also consider extending the definition of DisC diverse subsets to allow each item to be associated with a different radius. We do this to allow different areas of the data to contribute more or less items to the selected diverse subset.

We formalize the problem of locating minimum and minimum weighted DisC diverse subsets as an independent dominating set problem on graphs [58]. In the case of a single radius, items can be represented via an undirected graph. When multiple radii are employed, a directed graph is used instead. We show that locating minimum DisC diverse subsets is an NP-hard problem and provide a suite of algorithms for locating approximate solutions. We also consider the problem of adjusting the radius r, or *zooming*. 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 similar items, we take advantage of spatial data structures. In particular, we propose efficient implementations based on the M-tree [33].

In a nutshell, this thesis makes the following contributions:

- we introduce a new, intuitive definition of diversity, called DisC diversity, based on using a radius r rather than a size limit k to select diverse items,

- we extend DisC diversity to the weighted and multiple radii cases and introduce a graph-based view of the problem in both cases,

- we introduce incremental diversification through zooming-in and zooming-out,

- we show that locating DisC diverse subsets is an NP-hard problem and provide efficient algorithms for their computation as well as corresponding theoretical approximation bounds,
- we provide efficient M-tree tailored implementations of our algorithms and experimentally evaluate their performance, and
- we compare DisC diversity with other popular diversity models, both analytically and qualitatively.

Poikilo: A System for Evaluating the Results of Diversification Models and Algorithms. Finally, during the elaboration of this thesis, we also developed a system prototype, called Poikilo, which is a system designed to assist users in locating, visualizing and comparing diverse results based on a suite of different diversification models and algorithms. We provide implementations of a wide variety of diversification approaches for retrieving diverse results.

Users of Poikilo can submit queries to a number of different datasets and see a visualization of a diversified subset of their query result. Users can choose among a wide selection of diversification algorithms and specify various configuration parameters. Furthermore, they can zoom-in and zoom-out of this initial diverse subset and navigate between consequent windows in the case of streaming data.

1.3 Thesis Layout

The rest of this thesis is structured as follows. In Chapter 2, we overview related work in the field of diversification. In Chapter 3, we introduce indexing based on cover trees to address the dynamic diversification problem. In Chapter 4, we introduce a novel definition of diversity, called DisC diversity, as well as, a suite of algorithms for locating concise DisC diverse subsets. Chapter 5 presents a system prototype for visualizing diverse results. Finally, Chapter 6 summarizes this thesis and overviews directions for future work.

CHAPTER 2

RELATED WORK ON SEARCH RESULT DIVERSIFICATION

TODAY, most user searches are of an exploratory nature, in the sense that users are interested in retrieving pieces of information that cover many aspects of their information needs. Therefore, recently, *result diversification* has attracted considerable attention as a means of counteracting the over-specialization problem, i.e., the retrieval of too homogeneous results in recommender systems and web search, thus enhancing user satisfaction (e.g., [119, 107]). Consider, for example, a user who wants to buy an apartment and submits a related web search query. A diverse result, i.e. a result containing various apartments in different neighborhoods with different number of bedrooms and other characteristics is intuitively more informative than a result that contains a homogeneous result containing only apartments with similar features.

Diversification is also useful in counter-weighting the effects of personalization. Personalization aims at tailoring results to meet the preferences of each specific individual (e.g., [73, 98]). However, this may lead to overly limiting the search results. Diversification can complement preferences and provide personalization systems with the means to retrieve more satisfying results (as in [88]).

In this chapter, we survey the various approaches taken in the area of result diversification. We classify the ways that diverse items are generally defined in the related literature in three different categories, namely in terms of:

1. *content* (or *similarity*), i.e., items that are dissimilar to each other (e.g., [114, 107]),

2. *novelty*, i.e., items that contain new information when compared to previously seen ones (e.g., [35, 115]), and

3. *coverage*, i.e., items that belong to different categories (e.g., [11]).

We present various diversification algorithms of the related literature and classify them into two main groups, namely (i) *greedy* (e.g., [119, 57]) and (ii) *interchange* (e.g., [113, 81]) algorithms. We also present other criteria often used along with diversity and show various measures used for evaluating the performance of diversification systems.

The rest of this chapter is structured as follows. In Section 2.1, we classify various definitions of the diversification problem, while in Section 2.2, we see how diversity is combined with other ranking criteria. In Section 2.3, we review proposed algorithms for efficiently retrieving diverse results and, in Section 2.4, we show measures used for evaluating the diversity of selected items. Finally, Section 2.5 summarizes this chapter.

2.1 Diversity Definitions

Generally, the problem of selecting diverse items can be expressed as follows:

Definition 2.1 (Diversification Problem)**.** Given a set[1] $\mathcal{P}$ of n available items and a restriction k on the number of wanted results, select a subset S^* of k items out of the n available ones, such that, the diversity among the items of S^* is maximized.

In this section, we present various specific definitions of the result diversification problem that can be found in the research literature. We classify these definitions based on the way that diverse items are defined, i.e., (i) content, (ii) novelty and (iii) coverage. Note that, this classification is sometimes fuzzy, since these factors are related to each other and, therefore, a definition can affect more than one of them.

2.1.1 Content-based definitions

Content-based definitions interpret diversity as an instance of the *p-dispersion problem*. The objective of the p-dispersion problem is to select p out of n given items, so that the minimum distance between any pair of selected items is maximized [48]. The p-dispersion problem has been studied in the field of operations research for locating facilities that should be dispersed; such as franchises belonging to a chain or nuclear power plants. Formally, the p-dispersion problem is defined as follows:

[1]In some works, the term "set" is used loosely to denote a set with *bag semantics* or a *multiset*, where the same item may appear more than once in the set.

Given a set $\mathcal{P}$ of items, $\mathcal{P} = \{p_1, \ldots, p_n\}$, a distance metric d among items and an integer k, locate a subset S^* of $\mathcal{P}$, such that:

$$S^* = \underset{\substack{S \subseteq \mathcal{P} \\ |S|=k}}{\operatorname{argmax}} f(S, d) \tag{2.1}$$

where

$$f(S, d) = \min_{\substack{p_i, p_j \in \mathcal{P} \\ p_i \neq p_j}} d(p_i, p_j) \tag{2.2}$$

Similar content-based definitions of diversity have been proposed in the context of web search and recommender systems. Often, however, the objective function that is maximized is the *average distance* of any two items, instead of the minimum one, that is:

$$f(S, d) = \sum_{p_i, p_j \in \mathcal{P}} d(p_i, p_j) \tag{2.3}$$

These two common diversification objectives are referred to as the MaxMin and MaxSum diversification problems, respectively. Locating a diverse set for both problems has been shown to be an NP-hard problem (e.g., [47]). Recently, a thorough complexity analysis of content-based diversification was presented in [38], where three interesting problems are studied. The first problem is the existence of a set S with $f(S, d) > \alpha$, for some value α, while the second problem is finding the number of sets with diversity larger than α. Finally, given a set S, the third problem concerns deciding how many other sets exist with diversity larger than that of S. MaxMin and MaxSum were both shown to be NP-complete, #NP-complete and coNP-complete for the three problems respectively.

Content-based definitions are among the most popular ones in the related literature. For example, in [119], the diversity of a set of recommendations in a typical recommender system is defined based on their *intra-list similarity*, which is the application of Equation 2.3 along with a user-defined distance metric. In [113], the distance between recommendations is measured based on their *explanations*. Given a set of items $\mathcal{P}$ and a set of users $\mathcal{U}$, the explanation of an item $p_i \in \mathcal{P}$ recommended to a user $u_i \in \mathcal{U}$ can be defined in a content-based approach as:

$$expl(u_i, p_i) = \{p_j \in \mathcal{P} | sim(p_i, p_j) > 0 \wedge p_j \in items(u_i)\} \tag{2.4}$$

where $sim(p_i, p_j)$ is the similarity of p_i, p_j and $items(u_i)$ is the set of all items rated in the past by user u_i. A non content-based collaborative filtering approach is also considered, in which:

$$expl(u_i, p_i) = \{u_j \in \mathcal{U} | sim'(u_i, u_j) > 0 \wedge p_i \in items(u_j)\} \tag{2.5}$$

where sim' is a similarity metric between two users. The similarity sim between two items p_i and p_j can be defined based on the Jaccard similarity coefficient, the cosine similarity or any other similarity measure. The diversity of a set of items $S \subseteq \mathcal{P}$ is defined as the average distance of all pairs of items (as in Equation 2.3). A similar Jaccard-based similarity measure is also used in [57]. In that case, each item is described by a sketch produced by a number of hash functions. Another alternative distance metric used in that work is a taxonomy-based categorical distance when this can be applied (e.g., in the case of documents).

A content-based definition of diversity has also been applied in the context of publish/subscribe systems [46, 41]. Here, given a period or a window of matching events and an integer k, only the k most diverse of them (based on Equation 2.3) are delivered to interested subscribers.

Another definition that can be classified in this category is the one used in [107] in the context of database systems. Given a database relation $\mathcal{R} = (A_1, \ldots, A_m)$, a *diversity ordering* of $\mathcal{R}$, denoted $\prec_{\mathcal{R}}$, is a total ordering of its attributes based on their importance, say $A_1 \prec_{\mathcal{R}} \ldots \prec_{\mathcal{R}} A_m$. Also, a *prefix with respect to* $\prec_{\mathcal{R}}$, denoted ρ, is defined as a sequence of attribute values in the order given by $\prec_{\mathcal{R}}$, moving from higher to lower priority. Let ρ be a prefix of length ℓ and t, t' be two tuples of $\mathcal{R}$ that share ρ. The similarity between t and t' is defined as:

$$sim_\rho(t, t') = \begin{cases} 1 & \text{if } t.A_{\ell+1} = t'.A_{\ell+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

Now, given an integer k, a subset S of $\mathcal{R}$ with cardinality k is defined to be diverse with respect to ρ if all tuples in S share the prefix ρ and the sum of their pair-wise similarities, as defined above, is minimized. S is also said to be diverse with respect to $\mathcal{R}$ if it is diverse with respect to every possible prefix for $\mathcal{R}$. This approach is also followed in [76], where it is extended to the case of dynamic diversity orderings, i.e., the diversity ordering of the attributes is not static but is rather specified by the user along with the query. The proposed method is based on computing *core covers*, i.e., sets of tuples of size smaller than k whose tuples can cover all the tuples of a diverse set of size k. A different problem for structured data is considered in [81], that of selecting a limited number of features that can maximally differentiate the available items.

On a related issue, content-based definitions of diversity have been widely used to extend the k*-nearest neighbor* problem, so that, given an item p_i, the k spatially closest results that are sufficiently different to each other are retrieved (e.g., [59, 92, 62, 10, 9]). Similar approaches are followed; in [59], for example, the distance between two items is based on the Gower coefficient, i.e., a weighted average of the respective attribute differences of the items. Assuming δ_z to be equal to the difference between the z^{th} attributes of two items p_i, p_j then:

$$d(p_i, p_j) = \sum_z w_z \delta_z \tag{2.7}$$

where w_z is a weight corresponding to the z^{th} dimension of the items. Two items are considered diverse if their distance is greater than a given threshold and a set S is considered diverse if all pairs of items in it are diverse.

Finally, content-based diversification has also been employed to enhance *skyline* search. Let p_i, p_j be two items in $\mathcal{P}$. We say that p_i *dominates* p_j if it is better than or equal to p_j in all dimensions and strictly better than p_j in at least one dimension. An item belongs to the skyline of a set if there does not exist any other item that dominates it. Skylines can often be large in size. Selecting k representative skyline items is considered in [101], where representative items are selected so that the distance between a non-selected skyline item from its nearest selected item is minimized, and [105, 21], where the dominance relationships among items are exploited to select a diverse subset of the skyline items.

2.1.2 Novelty-based definitions

Novelty is a notion closely related to that of diversity, in the sense that items which are diverse from all items seen in the past are likely to contain novel information, i.e., information not seen before.

A distinction between novelty and diversity in the context of information retrieval (IR) systems is made in [35], where novelty is viewed as the need to avoid redundancy, whereas diversity is viewed as the need to resolve ambiguity. Each document p_i and query q are considered as a collection of *information nuggets* from the space of all possible nuggets $\mathcal{O} = \{o_1, \dots, o_m\}$. Given a binary random variable R_{p_i} that denotes whether a document p_i is considered relevant to a given query q, it holds that:

$$P(R_{p_i} = 1 | q, p_i) = P(\exists o_i, \text{ such that } o_i \in p_i \cap q) \tag{2.8}$$

Now, given an ordered list of documents $p_1, \dots, p_n$ retrieved by an IR system for q, the probability that the k^{th} document is both novel and diverse from the $k-1$ first ones is equal to the probability of that document containing a nugget that cannot be found in the previous $k-1$ documents. Given a list of $k-1$ preceding documents, the probability that a nugget $o_i \in \mathcal{O}$ is novel for a query q is:

$$P(o_i \in q | q, p_1, \dots, p_{k-1}) = P(o_i \in q) \prod_{j=1}^{k-1} P(o_i \notin p_j) \tag{2.9}$$

Assuming that all nuggets are independent and equally likely to be relevant for all queries, then:

$$P(R_{p_k} = 1 | q, p_1, \dots, p_k) = 1 - \prod_{i=1}^{m} (1 - \gamma \alpha J(p_k, o_i)(1-\alpha)^{r_{o_i,k-1}}) \tag{2.10}$$

where $J(p_k, o_i) = 1$ if some human judge has determined that p_k contains the nugget o_i (or zero otherwise), α is a constant in $(0, 1]$ reflecting the possibility of a judge error in positive assessment, $\gamma = P(o_i \in q)$ and $r_{o_i,k-1}$ is the number of documents ranked up to position $k-1$ that have been judged to contain o_i, i.e., $r_{o_i,k-1} = \sum_{j=1}^{k-1} J(p_j, o_i)$. This approach requires prior knowledge of the nuggets and also considerable amount of human effort for judging the relevance of documents in order to compute the related probabilities.

Another work based on novelty is [115], which aims at enhancing adaptive filtering systems with the capability of distinguishing novel and redundant items. Such systems should identify items that are similar to previously delivered ones, in the sense of having the same topic, but also dissimilar to them, in the sense of containing novel information. The redundancy R of each item p_i is measured with respect to its *similarity* to all previously delivered items, denoted $D(p_i)$, as follows:

$$R(p_i|D(p_i)) = \operatorname*{argmax}_{p_j \in D(p_i)} R(p_i|p_j) \tag{2.11}$$

where $R(p_i|p_j)$ is the redundancy (similarity) of p_i with respect to another item p_j. Three different ways for measuring $R(p_i|p_j)$ are considered, namely the *set difference*, the *geometric distance* and the *distributional distance*. The set difference is based on the number of new terms that appear in p_i:

$$R(p_i|p_j) = \left| set(p_i) \cap \overline{set(p_j)} \right| \tag{2.12}$$

In the above formula, given a term w and a document p_i, it holds that $w \in set(p_i)$, if and only if, $count(w, p_i) > h$, where h is a constant and $count(w, p_i) = \alpha_1 tf_{w,p_i} + \alpha_2 df_w + \alpha_3 rdf_w$. tf_{w,p_i} is the frequency of w in p_i, df_w is the number of all filtered documents that contain w, rdf_w is the number of delivered documents that contain w and $\alpha_1, \alpha_2, \alpha_3$ are constants with $\alpha_1 + \alpha_2 + \alpha_3 = 1$. The geometric distance is based on the cosine similarity between p_i and p_j. If we represent each document p_i as a vector $\mathbf{p_i} = (tf_{w_1,p_i}, tf_{w_2,p_i}, \ldots, tf_{w_m,p_i})^T$, where $w_1, w_2, \ldots, w_m$ are all the available terms, then:

$$\begin{aligned} R(p_i|p_j) &= \cos(\mathbf{p_i}, \mathbf{p_j}) \\ &= \frac{\mathbf{p_i}^T \mathbf{p_j}}{\|\mathbf{p_i}\| \, \|\mathbf{p_j}\|} \end{aligned} \tag{2.13}$$

Finally, the distributional distance is based on a probabilistic language model. Each document p_i is represented by a unigram word distribution θ_{p_i} and the distance among two documents is measured via the Kullback-Leibler (KL) formula:

$$\begin{aligned} R(p_i|p_j) &= -KL(\theta_{p_i}, \theta_{p_j}) \\ &= -\sum_{w_j} P(w_j|\theta_{p_i}) \log \frac{P(w_j|\theta_{p_i})}{P(w_j|\theta_{p_j})} \end{aligned} \tag{2.14}$$

A mixture-model approach is considered in order to find the language models for the θ distributions.

Novelty is also used in [96] in the context of publish/subscribe systems, to promote items matching subscriptions that have rarely been matched in the past.

Finally, using some notion of novelty in recommenders is also proposed in [75]. The authors argue that, since the available data are generally sparse and they frequently change, recommenders continuously have to make decisions based on incomplete and changing data, thus, increasing novelty is useful. A batch scenario is considered, where users receive lists of recommendations in intervals. Let L_1, L_2 be two subsequent lists of size k for some user. The objective of the proposed approach is to maximize $\frac{|L_2 \setminus L_1|}{k}$.

2.1.3 Coverage-based definitions

Some works view diversity in a different way, that of selecting items that cover many different interpretations of the user's information need. For example, [11] considers typical web search and, given a query q and a taxonomy C of independent information categories, aims at retrieving k documents that cover many interpretations of q, especially interpretations that are considered important. The result diversification problem in this context is formally defined as follows: Given a query q, a set of documents $\mathcal{P}$, a taxonomy C, a probability distribution $P(c|q)$ of each category $c \in C$ being relevant to q, the probability $V(p_i|q, c)$ of each document $p_i \in \mathcal{P}$ being relevant to each category c for q and an integer k, find a set of documents S^*, such that:

$$S^* = \underset{\substack{S \subseteq \mathcal{P} \\ |S|=k}}{\operatorname{argmax}} P(S|q) \tag{2.15}$$

where:

$$P(S|q) = \sum_c P(c|q)(1 - \prod_{p_i \in S}(1 - V(p_i|q, c))) \tag{2.16}$$

The probability of p_i *not* covering a relevant to the query q category c is equal to $(1 - V(p_i|q, c))$. Therefore, the above formula, in essence, maximizes the probability of each relevant category c being covered by at least one document in S. This method requires prior knowledge of the taxonomy and the learning of the probability distributions.

[80] also makes use of a cover-based definition of diversity to locate and highlight diverse concepts in documents. Given a set of *sentences* S, the *cover* of S is the union of all terms t appearing in any sentence p_i in S, that is:

$$cover(S) = \bigcup_{p_i \in S} \bigcup_{t \in p_i} \{t\} \tag{2.17}$$

Assuming a function $g(i)$ that measures the benefit we have by covering a term exactly i times, the *gain* of S is:

$$gain(S) = \sum_{i=0}^{|S|} \sum_{t \in \mathcal{T}_i} w(t) g(i) \tag{2.18}$$

where $\mathcal{T}_i$ is the set of terms appearing in exactly i sentences in S and $w(t)$ is a weight for the term t. Now, the result diversification problem is defined as follows: Given a document consisting of n sentences $\mathcal{P} = \{p_1, \ldots, p_n\}$ and an integer k, locate a set of sentences S^*, such that:

$$S^* = \underset{\substack{S \subseteq \mathcal{P} \\ |S| \leq k}}{\operatorname{argmax}} gain(S) \tag{2.19}$$

A coverage-based approach is also followed in [77] for a different problem, that of selecting a set of diversified queries to recommend to users. The proposed method makes use of a query log to cluster queries into *query concepts*. Given an input user query, a small set of diversified relevant concepts is first selected. To do this, a probabilistic model is exploited, which aims at maximizing the likelihood that the recommended concepts cover as many interpretations of the query as possible. Only one representative query from each cluster corresponding to each relevant concept is then returned to the user.

A different approach is followed in [37], where an item list is considered most diverse, with respect to some set of topics related to the query, when the number of items it provides on each topic is proportional to the topic's popularity. To select a highly diversified list of items, for each position in the ranked result list, the topic that best maintains the overall proportionality is determined. Then, the best item on this specific topic is added to the list.

On a related issue, [117] considers modifying the way of computing subtopic coverage in a taxonomy in the case of semi-structured data by also considering the structural similarity between subtopics, i.e., among the subtopics covered by a selected item, a subtopic that differs considerably from its parent is considered to be "less" covered than a subtopic that is structurally more similar to its parent. Structure is also considered in [60] along with content-based diversity, where items are organized in a tree and a tree edit-distance between items is also considered, along with content, during diverse item selection.

2.2 Combination of Diversity with other Criteria

Diversity is often used along with some other ranking criterion, most commonly that of *relevance* to the user's query. To the best of our knowledge, the first work in which the two measures were combined is [23], in which *marginal relevance*, i.e., a linear combination of relevance and diversity, is proposed as a criterion for ranking results retrieved by IR systems. An item has high marginal relevance if it is both relevant to the user query q and also exhibits minimal similarity to previously selected items. Formally, given the set of all retrieved items $\mathcal{P}$ and the set of already selected ones, denoted S, the item $p_i^* \in \mathcal{P}\backslash S$ that has the maximum marginal relevance to S is:

$$p_i^* = \underset{p_i \in \mathcal{P}\backslash S}{\operatorname{argmax}} \left[\lambda(w(p_i) - (1-\lambda) \max_{p_j \in S} d(p_i, p_j)) \right] \tag{2.20}$$

where $w(p_i)$ is the relevance of p_i to the query and λ, $\lambda \in [0,1]$, is a diversification tuning parameter. This approach has also been applied in [88] as a means to reformulate queries submitted in web search. The above formulation of the problem corresponds to the MaxSum diversification problem. The objective function that is maximized in this case is:

$$f(S, w, d, \lambda) = (|S| - 1) \sum_{p_i \in S} w(p_i) + 2\lambda \sum_{p_i, p_j \in S} d(p_i, p_j) \tag{2.21}$$

where $\lambda > 0$. Other common variations of combining relevance and diversity are MaxMin diversification, where:

$$f(S, w, d, \lambda) = \min_{p_i \in S} w(p_i) + \lambda \min_{p_i, p_j \in S} d(p_i, p_j) \tag{2.22}$$

and also a *mono-objective* formulation of the problem in which:

$$f(S, w, d, \lambda) = \sum_{p_i \in S} \left[w(p_i) + \frac{\lambda}{|\mathcal{P} - 1|} \sum_{p_j \in \mathcal{P}} d(p_i, p_j) \right] \tag{2.23}$$

A different mono-objective formulation based on coverage rather than content is introduced in [74] in the context of items organized in a graph. Let $N_\ell(S)$ be the set of items which are at most ℓ edges away from at least one item of S on the graph. Then:

$$f(S, w, \ell) = \sum_{p_i \in N_\ell(S)} w(p_i) \tag{2.24}$$

[118] also employes a mono-objective diversity function but, in addition, considers that items belong to some manifold space and exploits traditional manifold ranking algorithms to retrieve relevant and diverse items.

An interesting analysis of eight intuitive axioms that diversification systems should satisfy concerning the combination of relevance and diversity can be found in [57]. It is shown, however, that not all of them can be satisfied simultaneously.

The combination of diversity and relevance has also been studied in [114] as an optimization problem. Let once again $\mathcal{P} = \{p_1, \ldots, p_n\}$ be a set of items and D be an $n \times n$ distance matrix with the $(i, j)^{\text{th}}$ element being equal to $d(p_i, p_j)$. Let also $\mathbf{m}$ be an n-dimensional vector with the i^{th} element being equal to $w(p_i)$. Consider, finally, an integer k and a binary n-dimensional vector $\mathbf{y}$ with the i^{th} element being equal to 1, if and only if, p_i belongs to the k most highly relevant and diverse items. Now, given a diversification factor $\lambda \in [0, 1]$, we can define the problem of selecting k items that are both as relevant and diverse as possible as follows:

$$\begin{aligned} \mathbf{y}^* &= \operatorname*{argmax}_{\mathbf{y}} (1 - \lambda)\alpha \mathbf{y}^T D \mathbf{y} + \lambda \beta \mathbf{m}^T \mathbf{y} \\ \text{s.t.} \quad & \mathbf{1}^T \mathbf{y} = k \text{ and} \\ & y(i) \in \{0, 1\}, 1 \leq i \leq n \end{aligned} \tag{2.25}$$

where α and β are normalization parameters.

Diversity has also been combined with spatial distance, as a relevance characterization, when solving the k-nearest diverse neighbors problem, e.g., in [59].

There has been considerable work on training IR systems to retrieve relevant results exhibiting high diversity. For example, [93] proposes adapting the degree of diversification λ to each specific query. In particular, given a previously unseen query, the authors aim at predicting an effective trade-off between relevance and diversity based on a log of similar previous queries. First, a training set is constructed by retrieving diverse sets for all values of λ and keeping the one with the best diversity according to some measure. Then, a regression model is trained which is later used to decide a degree of diversification for unseen queries. Similarly, in [94], the authors aim at predicting the best IR method for different interpretations of the query. [31] also considers a combination of IR ranking metrics along with the diversity of retrieved results to train a probabilistic model for result retrieval.

Besides relevance, a couple of works exist that combine diversity with *sentiment* (e.g., [64, 13, 9]). Sentiment is a meaningful property of text items, e.g., documents or blog comments. It can be computed based on the content of the item, e.g., by using a dictionary of positive and negative words and counting the number of occurrences of these words in the text. After a sentiment (positive, negative or neutral) has been computed for each item, it can be used along with relevance and diversity when selecting items, as in [64, 9], or it can be used as an input parameter set by users to bias the selection of items, as in [13].

Another criterion that can be combined with diversity is *freshness*. For example, [99], which considers the problem of recommending unanswered questions of a knowledge base to potential answerers, combines relevance, diversity and freshness in an effort to present answerers with questions they may find more interesting to answer. Note that freshness and novelty are two different concepts, since a novel item, i.e., an item the user has not seen before, is not necessarily fresh, i.e., recently created.

Besides the more traditional definitions of diversity, recently, there has been some related work on *temporal* diversity, which concerns the creation time of items. A highly temporally diversified set is a set containing items that were created at different time from each other. Temporal diversity is useful in streaming environments. Consider, for example, a blog publishing entries on some specific topic. A post published around the same date as many other posts is intuitively less informative than a post which is published at a distant time. In this spirit, a hybrid content/temporal diversity approach is proposed in [67] to retrieve items that are both dissimilar and temporally distant to each other. A different form of temporal diversity is also considered in [116], where the selection of diverse results is affected not only by their (spatial) distances but also by the time of the computation. Such an approach may increase the likelihood of a purchase decision in on-line shops, since users often buy products at specific intervals, e.g., a user may buy a printer ink cartridge every six months. The proposed method aims at exploiting such known intervals to compute the utility surplus of each product at the time of the query. To do this, a probabilistic model is trained based on the intervals between subsequent purchases of products.

2.3 Algorithms

Given a set $\mathcal{P}$ of items, $\mathcal{P} = \{p_1, \dots, p_n\}$, a distance metric d among items and an integer k, the diversification problem is to locate a subset S^* of $\mathcal{P}$, such that the diversity among the selected items is maximized, where the diversity of a set of items is defined based on some specific definition of Section 2.1.

Generally, the diversification problem has been shown to be NP-hard (e.g., [47]). Thus, to solve large instances of the problem, we need to rely on heuristics. Many heuristic algorithms have been used in the research literature and have been employed for solving variations of the problem in more than one research fields. We can classify these algorithms into two main categories: (i) *greedy* and (ii) *interchange*. In following, we describe algorithms in each category and their applications.

2.3.1 Greedy Algorithms

Greedy algorithms are the ones most commonly used since they are intuitive and some of them are also relatively fast. Greedy algorithms generally make use of two sets: the set $\mathcal{P}$ of available items and the set S which contains the selected ones. Items are iteratively moved from $\mathcal{P}$ to S and vice versa until $|S| = k$ and $|\mathcal{P}| = n - k$. In most works, S is initialized with some item, e.g., the most relevant one, and then items are moved one-by-one from $\mathcal{P}$ to S until k of them have been selected. The item that is moved each time is the one that has the maximum *item-set distance* from S. The item-set distance, denoted $d(p_i, S)$, between an item p_i and a set of items S is defined based on its distance from the items in S, for example:

$$d(p_i, S) = \min_{p_j \in S} d(p_i, p_j) \tag{2.26}$$

or

$$d(p_i, S) = \sum_{p_j \in S} d(p_i, p_j) \tag{2.27}$$

Ties are generally broken arbitrarily.

This greedy approach is, for example, used in [119] in the context of recommender systems, where, given a set of recommendations $\mathcal{P} = \{p_1, \dots, p_n\}$ and their degrees of relevance $w(p_i)$, $1 \leq i \leq n$, to a user query, diverse recommendations are produced. S is initialized with the most relevant recommendation. Then, recommendations are added one-by-one to S as follows: For each recommendation p_i not yet added to S, its item-set distance from the recommendations already in S is computed. These "candidate" recommendations are then sorted in order of (i) relevance to the query and (ii) item-set distance to S. The rank of each recommendation is a linear combination of its positions in the two sorted lists. The recommendation with the minimum rank is added to S and the process is repeated until S has k recommendations. Note that the recommender system has to produce a larger number of recommendations (n) out of which the final k ones will be selected. The larger this number, the higher the possibility that more diverse recommendations will be located (at the cost of higher computation cost).

A greedy heuristic is also employed in [80] for locating diverse sentences in documents. At each round, the sentence which has the highest gain for S (Equation 2.19) is added to S. [11] also follows the greedy approach. In that case, an algorithm is proposed that, given the set of the top-k most relevant documents to a query, it re-orders them in a way, such that, the objective function of Equation 2.16 is maximized. [57] employs another greedy variation, first presented in [61] as a solution to the p-dispersion problem, in which, at each iteration, the *two* remaining items with the largest pair-wise distance are added to S. A greedy solution is also used in [113] for recommenders. However, in that case, threshold values are also used to determine when two recommendations are considered distant. [59] also uses a greedy algorithm for locating the k-nearest diverse neighbors to a given item.

A variation of such greedy algorithms is introduced in [109, 108], in which, instead of selecting the best candidate item at each step, an item is randomly chosen among the first couple of top-ranked items. Also, items are selected not only based on their item-set distance from S, but also from $\mathcal{P} \backslash S$, in an effort to select the most diverse items in early rounds. These two techniques were experimentally shown to increase the diversity of the produced diverse set in many cases.

A greedy heuristic is also employed in [78] for the selection of a representative, diverse subset of skyline items.

Also, an extension of greedy algorithms to the case of streaming data is introduced in [41], in which, every time a diverse set is requested, a greedy heuristic is used to fill in the places of any expired items in the previous diverse set. This approach often outperforms running the same greedy heuristic from scratch, since the algorithm is initialized with a seed of already highly diversified results.

A special case of greedy algorithms are neighborhood algorithms. These algorithms start with a solution S containing one random item and then iteratively add items to the solution. The items to be considered at each iteration are limited based on the notion of r-*neighborhood* of an item $p_i \in \mathcal{P}$, denoted $N(p_i, \mathcal{P}, r)$, defined as:

$$N(p_i, \mathcal{P}, r) = \{p_j \in \mathcal{P} : d(p_i, p_j) \leq r\} \tag{2.28}$$

In other words, all items that have a smaller than or equal to r distance to p_i belong to its r-neighborhood. At each iteration, only items outside the r-neighborhoods of all already selected items are considered. Out of these items, one is chosen to be added to the solution. This can be the fist located item outside those r-neighborhoods, the one that has the smallest sum of distances to the already selected items or the one that has the largest sum of distances to the already selected items [48]. Note that the selection of r plays an important role as it restricts the number of items that are considered at each iteration. In fact, given a value of r, a solution S with $|S| = k$ may not even exist.

2.3.2 Interchange Algorithms

Interchange algorithms have also been used in the literature for solving the diversification problem. Generally, these algorithms are initialized with a random solution S and then iteratively attempt to improve that solution by interchanging an item in the solution with another item that is not in the solution. At each round, possible interchanges are the first met one that improves the solution or the one that improves the solution the most.

An interchange heuristic that combines relevance and diversity is proposed in [113]. In this approach, S is initialized with the k most relevant items. At each iteration, the item of S that contributes the least to the diversity of the entire set, i.e., the one with the minimum item-set distance, is interchanged with the most relevant item in $\mathcal{P} \backslash S$. Interchanges stop when there are no more items in $\mathcal{P} \backslash S$ with higher relevance than a given threshold.

Another work that employs an interchange algorithm is [82], where, given a set of structured search results, the goal is to identify a subset of their features that are able to differentiate them. Starting with a random subset of features, at each iteration, one of these features is interchanged with a better candidate feature.

Finally, an interchange heuristic is employed [84] for streaming data. Upon the arrival of a new item p_i, all possible interchanges between p_i and the items in the current diverse set S are performed and p_i replaces an item in the solution, if this replacement increases diversity. A similar technique is also proposed in [46] in the context of publish/subscribe systems.

2.3.3 Other Algorithms

An algorithm for achieving diversity in database systems based on a tree index structure, i.e., the Dewey tree, is presented in [107]. Each tuple of a database relation is represented by a path in the tree. Higher levels of the tree represent more important attributes, according to the diversity ordering of the relation (see Section 2.1). Diverse tuples are retrieved by traversing this tree. A similar tree, called D-Index, is used in [76].

Motivated by the fact that the one-dimensional p-dispersion problem can be solved optimally in polynomial time, [48] considers a dimensionality-reduction heuristic that projects items in one dimension. However this approach does not result in good solutions in practice.

A hybrid greedy/interchange heuristic is used in [41] in the context of continuous data. In this case, a diverse subset S is located using a greedy approach and then its diversity is further improved by performing interchanges.

Another related approach is that of [79], where, given the set of a database query results, these results are grouped in k clusters and the corresponding k medoids are retrieved as a subset of k representative and diverse results.

Also, in [114], where the diversification problem is formulated as an optimization one, a solution is approximated via optimization techniques that include problem relaxation and quantization.

Concerning the computation of diverse skyline items, [105] follows an approach based on hashing items and computing distances based on hash-signatures.

Finally, a recent line of research focuses on viewing diversification as a top-k problem (e.g., [15, 51, 113]) and using threshold algorithms (e.g., [50, 49]) for selecting diverse items, aiming at pruning a portion of the candidate items. Threshold-based techniques have also be employed in [114], where variations of the optimization problem of Equation 2.25 are considered (e.g., maximize the diversity of the selected items given a relevance threshold and, the dual, maximize the relevance of the selected items given a minimum required diversity). Placing a threshold on diversity, however, may be hard, since it requires an estimation of the achievable diversity.

2.4 Evaluation Measures

The diversity of a set S of selected items can be evaluated by the value of the objective function f based on which the diversity problem is defined, e.g. Equation 2.2 or Equation 2.3. This approach is used in most of the related work (e.g., [119, 113, 46, 80, 114]). The computed value can be normalized by the corresponding value for the set S^*, i.e., the optimal solution to the diversification problem. This, however, is not always feasible due to the high cost of computing the optimal solution.

In the field of IR systems, there has been an effort to adapt traditional IR evaluation measures so as to become diversity-aware. A key difference of these approaches is that the retrieved results are usually viewed as an *ordered list* instead of a set. These adapted measures are usually applied along with novelty-based or coverage-based diversity definitions.

For example, [35] proposes evaluating retrieved results through a weighted *Normalized Discounted Cumulative Gain* measure (denoted α-NDCG), a measure often used in the context of IR systems that measures the gain of an item being at a specific position of the list, given the items that precede it. Given an ordered list of items, the k^{th} element of the list's gain vector, denoted $\mathbf{G}$, is computed based on Equation 2.10 as:

$$\mathbf{G}[k] = \sum_{i=1}^{m} J(p_k, o_i)(1-\alpha)^{r_{o_i,k-1}} \tag{2.29}$$

and the corresponding cumulative gain vector, denoted $\mathbf{GC}$, is computed as:

$$\mathbf{CG}[k] = \sum_{j=1}^{k} \mathbf{G}[j] \tag{2.30}$$

Usually, the elements of the cumulative gain vector are weighted according to their position in the list, so the discounted cumulative gain vector, denoted $\mathbf{DGC}$, is computed as:

$$\mathbf{DCG}[k] = \sum_{j=1}^{k} \frac{\mathbf{G}[j]}{\log_2(1+j)} \tag{2.31}$$

The discounted cumulative gain vector computed for a list is finally normalized by the optimal discounted cumulative gain. However, the computation of this optimal gain is an NP-complete problem and, thus, in practice, its value is approximated via heuristics.

The adaptation of the NDCG measure is also considered in [11, 27], where NDCG is aggregated over all available categories that a document may cover (see Section 2.1). This variation is called *Intent-Aware Normalized Discounted Cumulative Gain* measure (denoted NDCG-IA). Its value for the k^{th} element of a list S of items retrieved for a query q is:

$$NDCG\text{-}IA(S,k) = \sum_{c} P(c|q) NDCG(S,k|c) \tag{2.32}$$

The same aggregation method can be applied to other IR measures as well, such as *Mean Reciprocal Rank* (MRR) and *Mean Average Precision* (MAP).

A redundancy-aware variation of the traditional *precision* and *recall* measures is considered in [115]:

$$Redundancy\text{-}Precision = \frac{R^-}{R^- + N^-} \tag{2.33}$$

and

$$Redundancy\text{-}Recall = \frac{R^-}{R^- + R^+} \tag{2.34}$$

where R^- is the set of non-delivered redundant items, N^- is the set of non-delivered non-redundant ones and R^+ is the set of delivered redundant ones.

Variations of these measures have been studied in [106, 25, 110], where the introduction of the user as an explicit random variable is considered, in an effort to provide a personalized diverse result.

Besides deriving appropriate measures, user studies are also important in evaluating the usefulness of diversification. In [119], two thousand volunteers from the BookCrossing[2] community were asked to rate recommendations produced by using diversification techniques. The results vary according to the method used to acquire the initial recommendations but, overall, users rated diversified recommendations higher than non-diversified ones in all cases, as long as diversity contributed up to 40% to the linear combination of the relevance and diversity measures. A higher contribution led to a lower overall rating by the users. An interesting finding is that, when diversified results were presented to users, the individual recommendations where generally rated lower but the overall rating of the recommendation list as a whole was higher.

[2]http://www.bookcrossing.com

2.5 Summary

In this chapter, we presented the various definitions of the result diversification problem proposed in the research literature and classified them into three main categories, namely content-based, novelty-based and coverage-based. These three factors are closely related and, therefore, most related work considers more than one of them. We also reviewed different approaches taken for the combination of diversity with other ranking criteria, most commonly that of relevance to the user's information need. We classified the algorithms used in the literature for locating diverse items into two main categories (greedy and interchange) and also discussed other used approaches. Finally, we reviewed a number of measures used for evaluating diversity.

Chapter 3

Diverse Set Selection over Dynamic Data

The abundance of information available online creates the need for developing methods towards selecting and presenting to users representative result sets. To this end, result diversification has attracted considerable attention as a means of increasing user satisfaction. Result diversification takes many forms including selecting items so that their content dissimilarity, novelty or topic coverage is maximized [42].

Most previous approaches to computing diverse sets rely on *greedy* or *interchange* heuristics. Greedy heuristics (e.g., [119, 57]) build a diverse set incrementally, selecting one item at a time so that some diversity measure is maximized, whereas interchange heuristics (e.g., [113, 81]) start from a random initial set and try to improve it.

Despite the considerable interest in diversification, most previous research considers the *static* version of the problem, i.e., the available items out of which a diverse set is selected do not change over time. Here, we focus on the *dynamic* diversification problem, where insertions and deletions of items are allowed and the diverse set needs to be refreshed to reflect such updates. The dynamic problem was addressed in [41] using a greedy heuristic and in [84] using an interchange heuristic. Here, we propose an index-based approach.

Our solution is based on cover trees. Cover trees are data structures originally proposed for approximate nearest-neighbor search [16]. They were recently used to compute medoids [79] and priority medoids [18]. An index-based approach was also followed in [107] for the static version of the problem. The proposed index exploited a Dewey-encoding tree and can be used only with a specific diversity function on structured data. Our approach is more general and can be used with any diversity function.

Motivated by popular proactive delivery paradigms, such as news alerts, RSS feeds and notification services in social networks, where users specify their interests and receive relevant notifications, we also consider the *continuous* version of the problem, where diversified sets are computed over streams of items. To avoid overwhelming users by forwarding to them all relevant items, we consider the case in which a representative diverse set is computed, instead, whose size can be configured by the users. We introduce a sliding window model along with *continuity* requirements. Such requirements ensure that the order in which the diverse items are delivered follows the order of their generation and that an item does not appear, disappear and then re-appear in the diverse set.

We focus on the MaxMin diversity problem defined as the problem of selecting k out of a set of n items so that the minimum distance between any two of the selected items is maximized. The MaxMin problem is known to be NP-hard [47]. We propose a suite of algorithms that exploit the cover tree to provide solutions with varying accuracy and complexity. We provide theoretical results that bound the accuracy of the solutions achieved with regards to the optimal solution. The most efficient algorithm achieves a $^{b-1}/_{2b^2}$-approximation of the optimal solution, where b is the base of the cover tree, whereas the most expensive algorithm, a pruned implementation of a greedy heuristic, a $^1/_2$-approximation.

Our incremental algorithms produce results of quality comparable to that achieved by re-applying the greedy heuristic to re-compute a diverse set, while avoiding the cost of re-computation. Using cover trees also allows the efficient enforcement of the continuity requirements. Furthermore, multiple queries with different values of k can be supported.

In many cases, the items in the result set are associated with a relevance rank. We have extended our approach to support the computation of diverse subsets of ranked sets of items. We first show how to incorporate relevance in the diversity function used to build the cover tree. In addition, to allow for dynamically tuning the relative importance of relevance and diversity, we introduce an alternative solution based on weighted cover trees along with appropriate algorithms.

Finally, note that a recent line of research focuses on combining relevance and diversity by viewing diversification as a top-k problem ([15, 59, 51]). In such cases, threshold algorithms are used for selecting diverse items aiming at pruning a portion of the candidate items. Such approaches assume the existence of indices to provide sorted access to items, e.g., based on relevance or their distance from a given item. Here, instead, we aim at constructing indices that will guide the selection process.

In a nutshell, we make the following contributions:

- we address the dynamic diversification problem along with continuity requirements appropriate for a streaming scenario
- we present a suite of methods based on spatial indexing for the MaxMin problem and provide bounds for the achieved diversity with regards to the optimal solution, and
- we extend our methods to the case of selecting items that are both relevant and diverse.

The rest of this chapter is structured as follows. In Section 3.1, we present our diversification framework and define the Continuous k-Diversity Problem. Section 3.2 presents the cover tree index structure, while Section 3.3 introduces algorithms for computing diverse items. Section 3.4 considers combining diversity with relevance. Section 3.5 presents our experimental results. In Section 3.6, we present related work specific to index-based diversification, while in Section 3.7, a summary of the chapter.

3.1 The Diversification Model

Here, we focus on a general form of diversification based on content dissimilarity. Next, we first provide a formal definition of the diversity problem and then introduce its continuous variation.

3.1.1 The *k*-Diversity Problem

Let $\mathcal{P} = \{p_1, \dots, p_n\}$ be a set of n items. Given a distance metric $d : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^+$ indicating the dissimilarity of two items in $\mathcal{P}$, assume that the *diversity* of a set S, $S \subseteq \mathcal{P}$, is measured by a function $f : 2^{|\mathcal{P}|} \times d \rightarrow \mathbb{R}^+$. For a positive integer k, $k \leq n$, the k-Diversity Problem is the problem of selecting a subset S^* of $\mathcal{P}$ such that:

$$S^* = \operatorname*{argmax}_{\substack{S \subseteq \mathcal{P} \\ |S| = k}} f(S, d). \tag{3.1}$$

The choice of f affects the selection of items, even for a specific distance metric d. Two widely used functions are the *minimum* distance among the selected items and the *sum* of the distances of the selected items, formally defined as:

$$f_{\text{Min}}(S, d) = \min_{\substack{p_i, p_j \in S \\ p_i \neq p_j}} d(p_i, p_j) \tag{3.2}$$

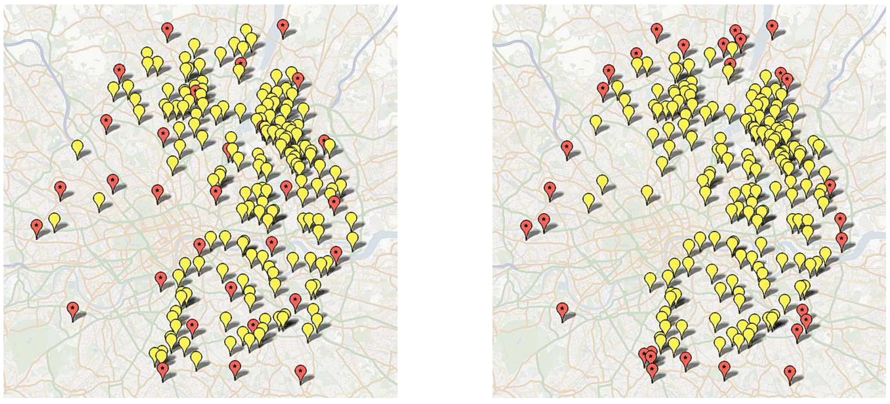

(a) MaxMin. (b) MaxSum.

Figure 3.1: MaxMin vs. MaxSum for $n = 200$ and $k = 30$. Diverse items are marked with a darker (red) color.

and

$$f_{\textsc{Sum}}(S, d) = \sum_{\substack{p_i, p_j \in S \\ p_i \neq p_j}} d(p_i, p_j). \tag{3.3}$$

The corresponding problems are called MaxMin and MaxSum. Intuitively, MaxMin aims at discouraging the selection of nearby items, while MaxSum at increasing the average pairwise distance among all items. An example is shown in Figure 3.1, which depicts the $k = 30$ most diverse, in terms of geographical distance, apartments for sale from a set of $n = 200$ available apartments in the London area retrieved from [6]. In general, MaxSum tends to select items in the outskirts of the set $\mathcal{P}$, whereas MaxMin selects items that are more representative of $\mathcal{P}$ in the sense that they provide a better coverage of it. In the rest of this chapter, we will focus on the MaxMin problem that exhibits this desired property.

The MaxMin Greedy Heuristic. The k-Diversity Problem is known to be NP-hard [47]. Various heuristics have been proposed, among which a natural greedy heuristic (Algorithm 3.1) has been shown experimentally to outperform the others in most cases ([41, 48]). The algorithm starts by selecting either a random item or the two items in $\mathcal{P}$ that are the furthest apart (line 1). Then, it continues by selecting the items that have the maximum distance from the items already selected, where the distance of an item p_i from a set of items S is defined as:

$$d(p_i, S) = \min_{p_j \in S} d(p_i, p_j) \tag{3.4}$$

It has been shown (e.g., in [89]) that the minimum distance of the set produced by the greedy heuristic is a $^1/_2$-approximation of the minimum distance of the optimal solution and that no polynomial algorithm can provide a better guarantee.

Algorithm 3.1 Greedy Heuristic.

Input: A set of items $\mathcal{P}$, an integer k.

Output: A set S with the k most diverse items of $\mathcal{P}$.

```
1: $\mathcal{P}^*, q^* \leftarrow \text{argmax}_{\mathcal{P},q \in \mathcal{P}, \mathcal{P} \neq q} d(\mathcal{P}, q)$
2: $S \leftarrow \{\mathcal{P}^*, q^*\}$
3: while $|S| < k$ do
4:     $\mathcal{P}^* \leftarrow \text{argmax}_{\mathcal{P} \in \mathcal{P}} d(\mathcal{P}, S)$
5:     $S \leftarrow S \cup \{\mathcal{P}^*\}$
6: return $S$
```

3.1.2 The Continuous *k*-Diversity Problem

We consider the case in which the set $\mathcal{P}$ changes over time and we want to refresh the computed k most diverse items to represent the updated set. In general, the insertion (or deletion) of even a single item may result in a completely different diverse set. The following simple example demonstrates this. Consider the set $\mathcal{P} = \{(4,4),(3,3),(5,6),(1,7)\}$ of points in the 2-dimensional Euclidean space and $k = 2$. The two most diverse items of $\mathcal{P}$ are $(4,4)$ and $(1,7)$. Assume that $(0,0)$ is added to $\mathcal{P}$. Now, the two most diverse items of $\mathcal{P}$ are $(0,0)$ and $(5,6)$.

In many applications, new items are generated in a continuous manner and, thus, the set $\mathcal{P}$ changes gradually over time. For example, consider a user continuously receiving a representative, i.e., most diverse, subset of the stream of available apartments in her area. We would like to offer her a continuous view of the most diverse items in this stream.

We adopt a sliding-window model where the k most diverse items are computed over sliding windows of length w in the input stream. The window length may be defined either in time units (e.g., "the k most diverse items in the last hour"), or in number of items (e.g., "the k most diverse items among the 100 most recent items"). Without loss of generality, we assume item-based windows. We allow windows to not only slide but also jump, i.e., move forward more than one item. For windows of length w and a jump step of length h, $h \leq w$, consequent windows share $w - h$ common items (Figure 3.2). Two consequent jumping windows may correspond, for example, to the items seen by a user in two consequent visits to her RSS reader application. Between these two visits, some items have ceased to be valid, new items have been generated, while a number of older items remain valid. Note that, for $h = 1$, jumping windows behave as regular sliding windows, while for $h = w$, consequent windows are disjoint which corresponds to periodic behavior with a period of length w.

Formally, let $\mathbb{P}$ be a stream of items. We denote the i^{th} jumping window of $\mathbb{P}$ as $\mathcal{P}_i$. The UNCONSTRAINED CONTINUOUS k-DIVERSITY PROBLEM is the problem of selecting a subset S_i^* of $\mathcal{P}_i$ for each $\mathcal{P}_i$, such that:

$$S_i^* = \underset{\substack{S_i \subseteq \mathcal{P}_i \\ |S_i| = k}}{\text{argmax}} f(S_i, d) \tag{3.5}$$

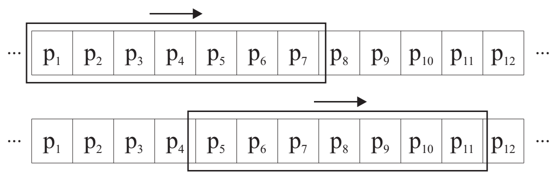


Figure 3.2: Two consequent windows with $w = 7$ and $h = 4$.

Constrained Continuous *k*-Diversity Problem. Since users may expect some continuity in the diverse sets they receive in consequent retrievals, we consider additional requirements on how the diverse sets change over time.

First, we want to avoid cases where diverse items that are still valid disappear. This may lead to confusing results, where an item appears in one diverse set, disappears in the next one and then appears again. Thus, an item selected as diverse remains in the diverse set until it expires, i.e., exits the current window. The diverse set is complemented with new items that are diverse with regards to those previously selected diverse items that are still valid. For instance, in the apartments example, the user sees new items that are diverse with regards to other previously seen apartments that are still available. We call this the *durability* requirement.

Second, we want the order in which items are chosen as diverse to follow the order of their appearance in the stream. This means that, once an item $\mathcal{P}$ is selected as diverse, we cannot later on select an item older than $\mathcal{P}$. We call this the *freshness* requirement. This is a desirable property in case of notification services, such as news alerts and RSS feeds, so as to ensure that the diverse items selected to be delivered to the users follow the chronological order of their publication. Raising this requirement may result in out-of-order delivery which may seem unnatural to users.

Based on the above observations, we now formally define the CONSTRAINED CONTINUOUS k-DIVERSITY PROBLEM.

Definition 3.1. (CONSTRAINED CONTINUOUS k-DIVERSITY PROBLEM). Let $\mathbb{P}$ be a stream of items, $\mathcal{P}_{i-1}$, $\mathcal{P}_i$, $i > 1$, be any two consequent windows and S^*_{i-1} be the diverse subset of $\mathcal{P}_{i-1}$. The CONSTRAINED CONTINUOUS k-DIVERSITY PROBLEM is the problem of selecting a subset S^*_i of $\mathcal{P}_i$, $\forall i$, such that:

$$S^*_i = \underset{\substack{S_i \subseteq \mathcal{P}_i \\ |S_i| = k}}{\operatorname{argmax}} f(S_i, d) \tag{3.6}$$

and the following two constraints are satisfied:

1. $p_j \in (S^*_{i-1} \cap \mathcal{P}_i) \Rightarrow p_j \in S^*_i$ (durability requirement),
2. Let p_l be the newest item in S^*_{i-1}. Then, $\nexists p_j \in \mathcal{P}_i \backslash S^*_{i-1}$ with $j < l$, such that, $p_j \in S^*_i$ (freshness requirement).

3.2 Index-based Diversification

To compute diverse sets in a dynamic setting, we rely on a tree structure, called *cover tree*, to index the items in $\mathcal{P}$. In this section, we provide a formal definition of the cover tree along with algorithms for constructing cover trees appropriate for the diversification problem.

3.2.1 The Cover Tree

A Cover Tree (CT) [16] for a set $\mathcal{P}$ is a leveled tree where each level is associated with an integer ℓ which increases as we move up the tree. Each node in the tree is associated with exactly one item $p \in \mathcal{P}$, while each item may be associated with multiple nodes, but with at most one at each level. In the following, when clear from context, we use $\mathcal{P}$ to refer to both the item $\mathcal{P}$ and the node associated with $\mathcal{P}$ at a specific level.

Let C_ℓ be the set of items at level ℓ and ℓ_{max} and ℓ_{min} be respectively the levels of the root and the leaves. A cover tree of *base* b, $b > 1$, is a tree that obeys the following invariants:

1. *Nesting*: For all levels ℓ, $\ell_{min} < \ell \leq \ell_{max}$, $C_\ell \subseteq C_{\ell-1}$, i.e., once an item $\mathcal{P}$ appears in the tree at some level, then there is a node associated with $\mathcal{P}$ at every lower level.

2. *Separation*: For all levels ℓ, $\ell_{min} \leq \ell \leq \ell_{max}$, and all distinct $p_i, p_j \in C_\ell$, it holds that, $d(p_i, p_j) > b^\ell$.

3. *Covering*: For all levels ℓ, $\ell_{min} \leq \ell < \ell_{max}$ and all $p_i \in C_\ell$, there exists a $p_j \in C_{\ell+1}$, such that, $d(p_i, p_j) \leq b^{\ell+1}$ and the node associated with p_j is the parent of the node associated with p_i.

An example is shown in Figures 3.3 and 3.4.

The CT was originally proposed with base $b = 2$. Here, we use a more general base b, $b > 1$. Generally, larger base values result in shorter and wider trees, since fewer nodes are able to "cover" the nodes beneath them. The value of b determines the granularity with which we move from one level to the next, i.e., how many more items become visible as we descend the tree.

Due to the CT invariants, if an item $\mathcal{P}$ appears for the first time at level ℓ of the tree, then $\mathcal{P}$ is a child of itself at all levels below ℓ. This observation provides us with a more space-efficient representation of the CT achieved by coalescing all nodes whose only child is a self child. We call this representation *explicit*. The explicit representation of a CT for a set $\mathcal{P}$ with n items requires $O(n)$ space [16]. Although we use an explicit representation in our experiments, for ease of presentation, we shall use the full implicit representation when describing the algorithms.

Next, we first present an algorithm for batch constructing a CT tailored for the MaxMin problem. Then, we consider an incremental construction of a CT appropriate for dynamic environments.

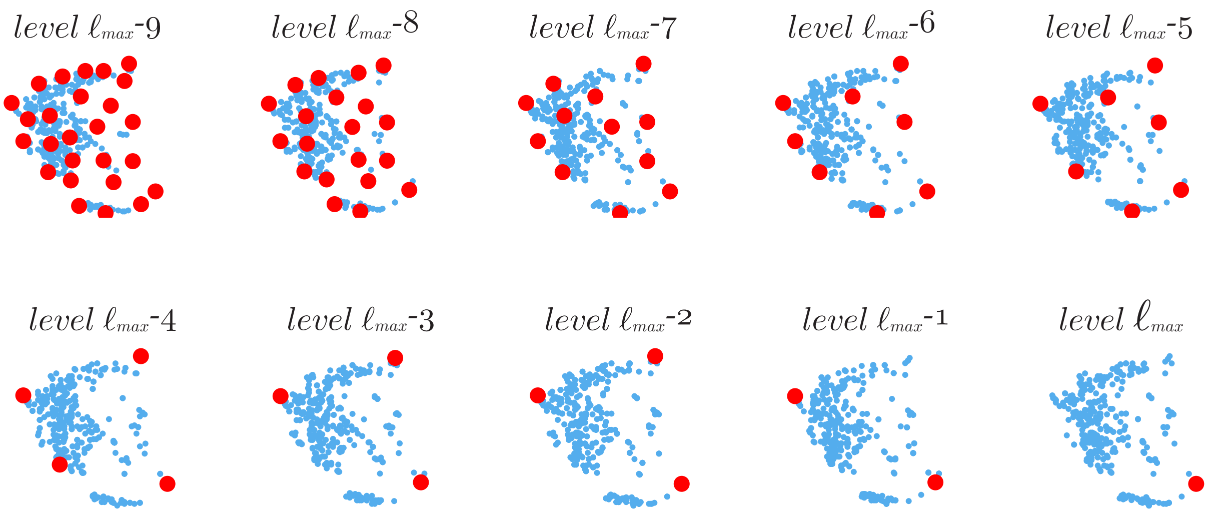


Figure 3.3: An example of the top 10 levels of a cover tree for a set of items in the 2-dimensional Euclidean space. Bold points represent the items (i.e., nodes) at each level, moving from lower levels to the root level, as we move from left to right.

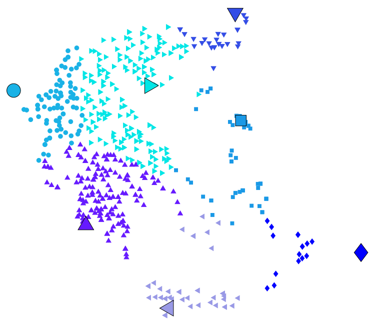


Figure 3.4: The $(\ell_{max} - 5)^{\text{th}}$ level of the cover tree of Figure 3.3. The items (i.e., nodes) of the level are drawn with large bold symbols, while the items of lower levels covered by these nodes are drawn with the same symbol (and color) as their ancestor.

3.2.2 Batch Construction

Given a set $\mathcal{P}$ of items, we build an appropriate CT for $\mathcal{P}$ using a bottom-up approach as depicted in Algorithm 3.2. First, we construct the lowest level that includes all items in $\mathcal{P}$ (lines 1-5). Then, given a level ℓ to build the next level $\ell+1$, we select items from level ℓ whose distance is larger than $b^{\ell+1}$ (so that the separation invariant is maintained), as long as such items exist (lines 6-17). To construct a CT whose items at each level are as far apart from each other as possible, we follow a greedy approach in selecting which items from C_ℓ to include in $C_{\ell+1}$. Specifically, we start by selecting the two items in C_ℓ that are the farthest apart from each other (line 9) and continue by selecting the item that has the largest minimum distance from the items already selected (lines 12-17).

The remaining items at C_ℓ are assigned a parent node from $C_{\ell+1}$ so that the covering invariant holds (lines 18-21). To reduce the overlap among the areas covered by sibling

Algorithm 3.2 Batch Cover Tree Construction.

Input: A set of items $\mathcal{P}$, a base b.

Output: A cover tree T of base b for $\mathcal{P}$.

```
1: $\ell \leftarrow \lfloor \log_b (\min_{p,q \in P} d(p,q)) \rfloor$
2: $T.C_\ell \leftarrow \emptyset$
3: for all $p \in P$ do
4:     $T.C_\ell \leftarrow T.C_\ell \cup \{p\}$
5: while $|T.C_\ell| > 1$ do
6:     $T.C_{\ell+1} \leftarrow \emptyset$
7:     $candidates \leftarrow T.C_\ell$
8:     $p^*, q^* \leftarrow \text{argmax}_{p,q \in candidates}\, d(p,q)$
9:     $T.C_{\ell+1} \leftarrow T.C_{\ell+1} \cup \{p^*, q^*\}$
10:    $candidates \leftarrow candidates \backslash \{p^*, q^*\}$
11:    while $candidates \neq \emptyset$ do
12:        $candidates \leftarrow candidates \backslash \{p : \exists q \in T.C_{\ell+1}$ with $d(p,q) \leq b^{\ell+1}\}$
13:        $p^* \leftarrow \text{argmax}_{p \in candidates}\, d(p, T.C_{\ell+1})$
14:        $T.C_{\ell+1} \leftarrow T.C_{\ell+1} \cup \{p^*\}$
15:        $candidates \leftarrow candidates \backslash \{p^*\}$
16:    for all $p \in T.C_\ell$ do
17:        $q^* \leftarrow \text{argmin}_{q \in T.C_{\ell+1}}\, d(p, T.C_{\ell+1})$
18:        make $q$ parent of $\mathcal{P}$
19:    $T.C_\ell \leftarrow T.C_{\ell+1}$
20:    $\ell \leftarrow \ell + 1$
21: return $T$
```

nodes, we assign each node to its closest candidate parent (line 19). We call this step *nearest parent* heuristic. Clearly, from the way the tree is constructed, $C_{\ell+1} \subseteq C_\ell$, thus the nesting invariant also holds. We call the tree constructed using this procedure, *Batch Cover Tree* (*BCT*).

We shall prove that the set of items C_ℓ at each level ℓ of the BCT correspond to the result of applying the MaxMin greedy heuristic (Algorithm 3.1) on $\mathcal{P}$, for $k = |C_\ell|$. Our proof uses the following observation. Let $S^{GR}(P,k)$ denote the result of applying the MaxMin greedy heuristic on $\mathcal{P}$ for k.

Observation 3.1. *For any $k > 2$, $S^{GR}(P, k+1) \supset S^{GR}(P,k)$.*

Theorem 3.1. *Let $\mathcal{P}$ be a set of items and T be a BCT for $\mathcal{P}$. For all levels ℓ, $\ell_{min} \leq \ell < \ell_{max}$, of T, it holds:*

$$C_\ell = S^{GR}(P, |C_\ell|)$$

Proof. We shall prove the theorem by induction on the level ℓ. The theorem holds trivially for ℓ equal to the lowest level of the tree, since this level includes all items in $\mathcal{P}$. Assume that it holds for level ℓ. We shall show that it also holds for level $\ell + 1$.

Consider the construction of level $\ell + 1$. From the induction step, it holds that, $C_\ell = S^{GR}(P, |C_\ell|)$. Let $\mathcal{P}$ be the *first* item in C_ℓ such that $\mathcal{P}$ is the best candidate, i.e., has the maximum minimum distance from the items already selected, but $\mathcal{P}$ cannot be moved to $C_{\ell+1}$ because $\mathcal{P}$ is covered by an item already selected to be in $C_{\ell+1}$. Let C', $C' \subset C_\ell$, be the set of items already selected to be included in $C_{\ell+1}$. This means that, it holds: $\min_{q' \in C'} d(p, q') \geq \min_{q' \in C'} d(p', q')$, for all $p' \in C_\ell \backslash C'$ (1) and, also, $\exists\ q \in C'$ such that $d(p, q) \leq b^{\ell+1}$ (2). From (1) and (2), we get that for all $p' \in C_\ell \backslash C'$, $\exists\ q \in C'$ such that $d(p', q) \leq b^{\ell+1}$, that is, all remaining items are already covered by items in C'.

Thus, $\mathcal{P}$ is the last item that is considered for inclusion in $C_{\ell+1}$, since all other remaining items in C_ℓ are already covered. Therefore, to construct $C_{\ell+1}$, the items from C_ℓ to be included in level $\ell+1$ are considered in the same order as in the greedy heuristic, until one item that violates the separation criterion (it is covered by the selected items) is encountered. When this happens the selection stops. By the induction step and Observation 3.1, this concludes the proof. ∎

Note that, we have made an implicit assumption that no ties occur when selecting items. In the absence of ties, both the greedy heuristic and the BCT construction algorithm select items deterministically. We can raise this assumption, by considering that if ties exist, these are resolved in a specific order that may vary depending on the nature of the items, for instance, by selecting the most recent among the items.

Regarding the complexity of Algorithm 3.2, computational steps are shared among levels. Each level $C_{\ell+1}$ is a subset of C_ℓ and, more specifically, it consists of the items of C_ℓ in the order in which they were inserted into C_ℓ up to the first item whose minimum distance from the already selected items of C_ℓ is smaller than $b^{\ell+1}$. Therefore, it suffices to perform these computational steps only once (at the lowest level) and just maintain the order in which each item was selected from the lowest level for inclusion in the next level. This gives us an $O(n^2)$ complexity.

As a final remark, another way to view the BCT is as caching the results of the greedy heuristic for all k and indexing them for efficient retrieval.

3.2.3 Dynamic Construction

In dynamic environments, it is not efficient to re-construct a BCT whenever an item is inserted or deleted. Thus, we construct a cover tree *incrementally* as new items arrive and old ones expire. We refer to such trees as *Incremental Cover Trees* (*ICTs*).

Incremental Insertion. To insert a new item $\mathcal{P}$ into a CT, we use the recursive insert procedure shown in Algorithm 3.3. It is based on the insertion algorithm in [16] and subsequent corrections in [71] that we have extended to work for any $b > 1$. `Insert` is called recursively starting from the root level, until a level is found at which $\mathcal{P}$ is separated from all other items (lines 2-4). Each time, `Insert` is called only with the nodes that cover $\mathcal{P}$ (line 5). When the first level such that $\mathcal{P}$ is separated from all other items is located, a node that covers $\mathcal{P}$ is selected as its parent (lines 8-9). To select a node as a parent for $\mathcal{P}$, we use a *nearest parent* heuristic (as in the batch construction)

Algorithm 3.3 Insert($\mathcal{P}$, $T.Q_\ell$, ℓ)

Input: An item $\mathcal{P}$, a set of nodes $T.Q_\ell$ of a cover tree T at level ℓ.

```
1: $C \leftarrow \{children(q) : q \in T.Q_\ell\}$
2: **if** $d(p, C) > b^\ell$ **then**
3:     **return** true
4: **else**
5:     $T.Q_{\ell-1} \leftarrow \{q \in C : d(p,q) \le \frac{b^\ell}{b-1}\}$
6:     $flag \leftarrow Insert(p, T.Q_{\ell-1}, \ell - 1)$
7:     **if** $flag$ **and** $d(p, T.Q_\ell) \le b^\ell$ **then**
8:         $q^* \leftarrow \text{argmin}_{q \in T.Q_\ell, d(p,q) \le b^\ell} d(p,q)$
9:         make $\mathcal{P}$ a child of $q^*$
10:        **return** false
11:    **else**
12:        **return** flag
```

Algorithm 3.4 Delete(p, $\{T.Q_\ell, T.Q_{\ell+1}, \ldots, T.Q_{\ell_{max}}\}, \ell$)

Input: An item $\mathcal{P}$, sets of nodes $\{T.Q_\ell, T.Q_{\ell+1}, \ldots, T.Q_{\ell_{max}}\}$ of a cover tree T, a level ℓ.

```
1: $C \leftarrow \{children(q) : q \in T.Q_\ell\}$
2: $T.Q_{\ell-1} \leftarrow \{q \in C : d(p,q) \le \frac{b^\ell}{b-1}\}$
3: $Delete(p, \{T.Q_{\ell-1}, T.Q_\ell, \ldots, T.Q_{\ell_{max}}\}, \ell - 1)$
4: **if** $d(p, C) = 0$ **then**
5:     delete $\mathcal{P}$ from level $\ell - 1$ and from $children(parent(p))$
6:     **for** $q \in children(p)$ in greedy order **do**
7:         $\ell' \leftarrow \ell - 1$
8:         **while** $d(q, T.Q_{\ell'}) > b^{\ell'}$ **do**
9:             add $q$ into level $\ell'$
10:            $T.Q_{\ell'} \leftarrow T.Q_{\ell'} \cup \{p\}$
11:            $\ell' \leftarrow \ell' + 1$
12:        $q^* \leftarrow \text{argmin}_{q' \in T.Q_{\ell'}} d(p', q)$
13:        make $q$ a child of $q^*$
```

and assign $\mathcal{P}$ to its closest candidate parent. The complexity of the algorithm depends on how many nodes of each level cover $\mathcal{P}$.

Next, we prove the correctness of the insertion algorithm.

Theorem 3.2. *Let T be a cover tree for a set $\mathcal{P}$ and $\mathcal{P}$ be an item, $\mathcal{P} \notin \mathcal{P}$. If $\mathcal{P}$ can be inserted at an existing level of T, then calling `Insert` (Algorithm 3.3) with input $\mathcal{P}$ and the root level $C_{\ell_{max}}$ of T returns a cover tree for $\mathcal{P} \cup \{p\}$.*

Proof. Since $\mathcal{P}$ can be inserted at an existing level, there is always a (sufficiently low) level of the tree where the condition of line 2 holds for the first time. Let $\ell - 1$ be this level. Since $\ell - 1$ is the highest level where this condition holds, it must hold that $d(p, T.Q_\ell) \le b^\ell$. Therefore, the second condition of line 7 holds and we can always find a parent for the new node, thus maintaining the covering invariant. Whenever a new node is inserted at some level, it is also inserted at all lower levels as a child of itself, thus the nesting invariant is maintained. It remains to prove the separation invariant. We shall prove it for level $\ell - 1$. The proof proceeds similarly for lower levels. Consider some other item q in $C_{\ell-1}$. If $q \in C$, then $d(p,q) > b^{\ell-1}$. If not, then there is a higher level $\ell' > \ell$ where some ancestor of q, say q' was eliminated by line 5, i.e., $d(p,q') > \frac{b^{\ell'+1}}{b-1}$. Using the triangle inequality, we have that $d(p,q) \geq d(p,q') - d(q,q') \geq d(p,q') - \sum_{j=\ell}^{\ell'} b^j = d(p,q') - \frac{b^{\ell'+1}-b^{\ell}}{b-1} > \frac{b^{\ell'+1}}{b-1} + \frac{b^{\ell}-b^{\ell'+1}}{b-1} = \frac{b^{\ell}}{b-1} > \frac{(b-1)b^{\ell-1}}{b-1} = b^{\ell-1}$. ∎

For clarity of presentation, we have made the assumption that the new item $\mathcal{P}$ can be inserted at an existing level of the tree. In the general case, when the new item $\mathcal{P}$ must be inserted at a level lower than ℓ_{min}, we keep copying nodes of $C_{\ell_{min}}$ to a new level $C_{\ell_{min}-1}$, until $\mathcal{P}$ is separated from all other items in the new level. Similarly, when $\mathcal{P}$ has a distance from the root node larger than $b^{\ell_{max}}$, we promote both the root node and $\mathcal{P}$ to a new higher level $\ell_{max} + 1$ and repeat this process until one of the two nodes can cover the other. Note that, since the explicit representation of the tree is stored, duplication of levels is only virtual and is performed very efficiently.

Incremental Deletion. Similar to insertion, to delete an item, starting from the root, `Delete` (Algorithm 3.4) is called until the item $\mathcal{P}$ to be deleted is located, keeping note of the candidate nodes at each level that may have $\mathcal{P}$ as a descendant. When $\mathcal{P}$ is located, it is deleted from the tree. In addition, all of its children are reassigned to some other candidate parent.

Algorithm 3.4 includes two heuristics for improving the quality of the resulting CT. One is the usual *nearest parent* heuristic shown in line 13: we assign each child of $\mathcal{P}$ to the closest among its candidate parents. The other heuristic refers to the order in which the children of $\mathcal{P}$ are examined in line 6. We examine them in a greedy manner starting from the one farthest apart from the items at level ℓ' and continue to process them in decreasing order of their distance to the items currently in ℓ'.

Theorem 3.3. *Let T be a cover tree for a set $\mathcal{P}$ and $\mathcal{P}$ be an item, $\mathcal{P} \in \mathcal{P}$. If $\mathcal{P} \notin C_{\ell_{max}}$ of T, calling `Delete` (Algorithm 3.4) with input $\mathcal{P}$ and the root level $C_{\ell_{max}}$ of T returns a cover tree for $\mathcal{P} \setminus \{p\}$.*

Proof. The item $\mathcal{P}$ is deleted from all levels that include it, thus the nesting invariant is maintained. For each child q of $\mathcal{P}$, we move up the tree, until a parent for q is located, inserting q in all intermediate levels ℓ' to ensure that the nesting invariant is not violated. Such a parent is guaranteed to be found (at least at the level of the root). Adding q under its new parent does not violate the separation invariant in any of the intermediate levels since $d(q,q') > b^{\ell'}$, for all q' in $T.Q_{\ell'}$. The covering constraint also holds for the parent of q. ∎

For ease of presentation, we assumed that $\mathcal{P} \notin C_{\ell_{max}}$. Otherwise, we need to select a new root. Note that, it is possible that none of the children of the old root covers all of its siblings. In this case, we promote those siblings that continue to be separated from each other in a new (higher) level $\ell_{max} + 1$ and continue to do so until we end up with a level having a single node.

3.3 Diverse Set Computation

In this section, we present algorithms that use the cover tree to solve the k-diversity problem. The Level algorithms exploit the separation property, i.e., the higher the tree level, the farthest apart its nodes. We also present an efficient implementation of the Greedy Heuristic (Algorithm 3.1) that exploits the covering property to prune the search space.

3.3.1 The Level Family of Algorithms

We consider first the intuitive algorithm of selecting k items from the highest possible level of a cover tree, that is, from level ℓ_k, such that, $|C_{\ell_k+1}| < k$ and $|C_{\ell_k}| \geq k$ (depicted in Algorithm 3.5). Locating this level can be implemented efficiently, e.g., by using a hash table to store the size of each level. After locating ℓ_k, the complexity of the algorithm is $O(k)$, since a random subset of C_{ℓ_k} is selected.

Algorithm 3.5 Level-Basic Algorithm.

Input: A cover tree T, an integer k.
Output: A set S with k diverse items in T.

```
1: $\ell_k \leftarrow \ell_{max}$
2: while $|T.C_{\ell_k}| < k$ do
3:     $\ell_k \leftarrow \ell_k - 1$
4: $S \leftarrow$ any subset of size $k$ of $T.C_{\ell_k}$
5: return $S$
```

The following theorem characterizes the solution attained by the *Level-Basic* algorithm with respect to the optimal solution.

Theorem 3.4 (Approximation Bound). *Let $\mathcal{P}$ be a set of items, $d^{OPT}(\mathcal{P}, k)$ be the minimum distance of the optimal diverse set for the* MaxMin *problem for $k \geq 2$ and $d^{CT}(\mathcal{P}, k)$ be the minimum distance of the diverse set computed by the Level-Basic algorithm. Then:*

$$d^{CT}(\mathcal{P}, k) \geq \alpha \; d^{OPT}(\mathcal{P}, k), \text{ where } \alpha = \tfrac{b-1}{2b^2}.$$

Proof. Let $S^{OPT}(\mathcal{P}, k)$ be an optimal set of k diverse items. To prove Theorem 3.4, we shall bound the level where the least common ancestor (LCA) of any pair of items p_1, $p_2 \in S^{OPT}(\mathcal{P}, k)$ appears in the cover tree. Assume that the LCA of any two items p_1, p_2 in the optimal solution appears for the first time at level m. That is, m is the lowest (furthest from the root) level that such an LCA appears.

Let us now compute a bound on m. Assume that the LCA of any two items p_1, p_2 $\in S^{OPT}(\mathcal{P}, k)$ appears at level m. Let $\mathcal{P}$ be this ancestor. From the triangle inequality, $d(p_1, p) + d(p_2, p) \geq d(p_1, p_2)$. Since p_1, $p_2 \in S^{OPT}(\mathcal{P}, k)$, it holds that, $d(p_1, p_2) \geq d^{OPT}(\mathcal{P}, k)$. Thus:

$$d(p_1, p) + d(p_2, p) \geq d^{OPT}(\mathcal{P}, k). \quad (1)$$

From the covering invariant of the cover tree, it holds that, $d(p_1, p) \leq \sum_{j=-\infty}^{m} b^j \leq \frac{b^{m+1}}{b-1}$. Similarly, $d(p_2, p) \leq \frac{b^{m+1}}{b-1}$. Substituting in (1), we get that $2\ \frac{b^{m+1}}{b-1} \geq d^{OPT}(\mathcal{P}, k)$. Solving for m, we have $m \geq \log_b \left(\frac{b-1}{2} d^{OPT}(\mathcal{P}, k)\right) - 1$.

Since m is the first level that the LCA of any two items in the optimal solution appears, from the covering property, it holds that at level $m - 1$, there are at least k items, i.e., the distinct ancestors of the k items in the optimal solution. Thus, there are at least k items at level

$$m - 1 = \log_b \left(\frac{b-1}{2} d^{OPT}(\mathcal{P}, k)\right) - 2. \quad (2)$$

This means that the cover tree algorithm will select items from this or a higher level. From the separation invariant of the cover tree, we have $d^{CT}(\mathcal{P}, k) \geq b^{m-1}$. Using (2), we get that $d^{CT}(\mathcal{P}, k) \geq b^{\log_b\left(\frac{b-1}{2} d^{OPT}(\mathcal{P},k)\right)-2} \Rightarrow d^{CT}(\mathcal{P}, k) \geq \frac{b-1}{2} d^{OPT}(\mathcal{P}, k)\ b^{-2}$, which proves the theorem. ∎

We also consider algorithms that, instead of selecting any k items from level ℓ_k, select these items greedily. The first algorithm, called *Level-Greedy*, performs a greedy selection among all items at level ℓ_k. This requires $k|C_{\ell_k}|$ distance computations. The second algorithm, called *Level-Inherit*, (Algorithm 3.6), initializes the solution with all items in C_{ℓ_k+1} and selects the remaining $k - |C_{\ell_k+1}|$ items from C_{ℓ_k} in a greedy manner. Thus, it requires $(k - |C_{\ell_k+1}|)|C_{\ell_k}|$ distance computations.

Algorithm 3.6 Level-Inherit Algorithm.

Input: A cover tree T, an integer k.

Output: A set S with k diverse items in T.

```
1: $\ell_k \leftarrow \ell_{max}$
2: **while** $|T.C_{\ell_k}| < k$ **do**
3:     $\ell_k \leftarrow \ell_k - 1$
4: $S \leftarrow T.C_{\ell_k+1}$
5: $candidates \leftarrow T.C_{\ell_k} \backslash T.C_{\ell_k+1}$
6: **while** $|S| < k$ **do**
7:     $p^* \leftarrow \text{argmax}_{p \in candidates}\ d(p, S)$
8:     $S \leftarrow S \cup \{p^*\}$
9:     $candidates \leftarrow candidates \backslash \{p^*\}$
10: **return** $S$
```

Clearly, the bound of Theorem 3.4 holds for the solution of Level-Greedy. It also holds for the solution of Level-Inherit, since due to nesting, an item that appears at some level of the tree also appears at all levels below it, thus, all items selected by Level-Inherit belong to C_{ℓ_k}. In general, these two algorithms are expected to produce more diverse sets than the estimated general bound. In particular, for Batch Cover Trees (BCTs), we can prove a better approximation. Specifically, it follows from Theorem 3.1, that the application of Level-Greedy and Level-Inherit algorithms on a BCT produces the same solution with the greedy heuristic.

Corollary 3.1. *Let $\mathcal{P}$ be a set of items, $k \geq 2$, $d^{GR}(P,k)$ be the minimum distance of the diverse set computed by the greedy heuristic and $d^{BCT}(\mathcal{P},k)$ be the minimum distance of the diverse set computed by Level-Basic or Level-Inherit when applied on a BCT for $\mathcal{P}$. It holds that $d^{BCT}(\mathcal{P},k) = d^{GR}(P,k) \geq {}^1/_2\ d^{OPT}(\mathcal{P},k)$.*

3.3.2 Greedy Heuristic using Cover Trees

Next, we present algorithms that use the cover tree to prune the search space of the greedy heuristic.

The algorithms proceed as follows. We initialize the diverse set S by selecting either the root or the two furthest apart leaves of the tree. This corresponds to initializing the greedy heuristic with either a random or the two most distant items. Then, we proceed in rounds. At each round, we select one item by descending the tree seeking for the item $\mathcal{P}$ with the maximum distance, $d(p,S)$, from the current set S. Specifically, at each of the $k-1$ (or $k-2$) rounds, we start descending the tree from the highest level C_ℓ that contains items that are not already in S. We locate the item $\mathcal{P}$ of C_ℓ with the largest $d(p,S)$ and use it to prune its siblings. Then, we consider as candidates the children of all non-pruned nodes of C_ℓ and repeat the process for $C_{\ell-1}$. In the end, the best candidate from the leaf level is added to S and we proceed to the next round. This process is shown in Algorithm 3.7.

Pruning is based on the following observation. Suppose that at some point we consider for inclusion in S an item $\mathcal{P}$ in C_ℓ. Let $d(p,S)$ be the distance of $\mathcal{P}$ from S and q be any sibling of $\mathcal{P}$. Then, the best candidate in the subtree of q is at distance at most:

$$\sum_{j=\ell_{min}+1}^{\ell} b^j = \frac{b^{\ell+1} - b^{\ell_{min}+1}}{b-1} \tag{3.7}$$

from q. Therefore, we can safely prune nodes according to the following CT pruning rule:

CT Pruning Rule: Let $\mathcal{P}$ and q be two nodes at level ℓ in a CT. If $d(p,S) \geq d(q,S) + \frac{b^{\ell+1}-b^{\ell_{min}+1}}{b-1}$, we can prune the subtree rooted at q.

The CT pruning rule is pessimistic, in the sense that it assumes that each node may have a child located as far as possible from it. A more efficient pruning rule can be used at the trade-off of maintaining some extra information. Specifically, at each node $\mathcal{P}$ in the tree, we maintain the distance of $\mathcal{P}$ from the node in its subtree that is the furthest

Algorithm 3.7 Greedy-CT Algorithm.

Input: A cover tree T, an integer k.

Output: A set S with the k most diverse items in T.

```
1: $S \leftarrow \{T.root\}$
2: **while** $|S| < k$ **do**
3:     $Q \leftarrow children(T.root)$
4:     **while** $Q \neq \emptyset$ **do**
5:         $p^* \leftarrow \arg\max_{p \in Q} d(p, S)$
6:         $Q' \leftarrow children(p^*)$
7:         **for all** $\mathcal{P} \in Q \backslash \{p^*\}$ **do**
8:             **if** $q$ is not pruned by the pruning rule **then**
9:                 $Q' \leftarrow Q' \cup children(q)$
10:         $Q \leftarrow Q'$
11:     $S \leftarrow S \cup \{p^*\}$
12: **return** $S$
```

apart from $\mathcal{P}$. We call this distance the *distance weight* of $\mathcal{P}$ denoted by $w_d(p)$. We call the tree that is annotated with such weights a *Weighted Cover Tree* (*WCT*). Then, we can use Algorithm 3.7 along with the following pruning rule:

WCT Pruning Rule: Let $\mathcal{P}$ and q be two nodes at level ℓ in a WCT. If $d(p, S) \geq d(q, S) + w_d(q)$, we can prune the subtree rooted at q.

3.3.3 Other Issues

Constrained Continuous *k*-Diversity. The two requirements of constrained continuous k-diversity (Definition 3.1) can be easily enforced using cover trees. For the durability requirement, items that are selected as diverse are marked as such and remain part of the diverse set, until they expire. Let z be the number of such items. In this case, our algorithms just select $k - z$ additional items from the tree. For the freshness requirement, non-diverse items that are older than the newest item in the current diverse set are marked as "invalid" in the CT and are not considered further as candidates for inclusion.

Adjusting *k*. The CT can be used to provide results for multiple queries with different k. Thus, each user can individually tune the amount of diverse items she wishes to receive. Furthermore, the CT supports a "zooming" type of functionality. Assume that a user selects a specific value for k. After receiving the k most diverse items, she can request a larger number of closer to each other items by choosing a larger k ("zoom-in"), or a smaller number of farther apart items by choosing a smaller k ("zoom-out"). We can exploit the nesting invariant to achieve continuity in the following sense. Let S be the set of the k most diverse items and let ℓ be the highest level of the CT at which all items of S appear. For $k' > k$, we would like the set S' with the k' most diverse items

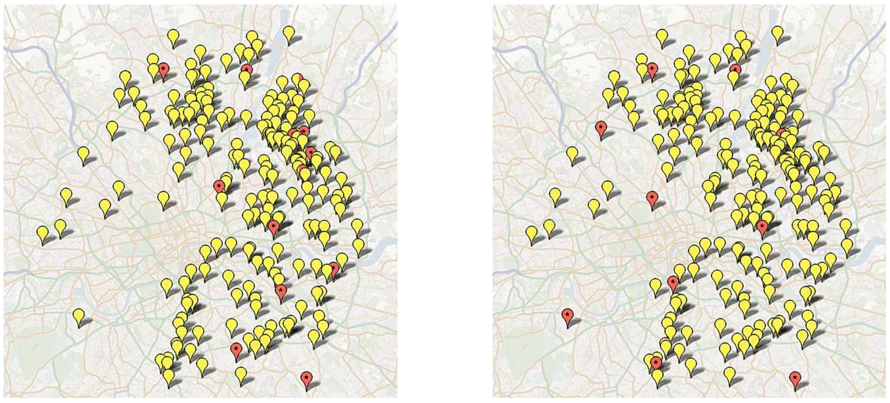

(a) Relevance. (b) Relevance and Diversity.

Figure 3.5: Selecting $k = 10$ out of $n = 200$ apartments in London based (a) solely on relevance (i.e., price) and (b) incorporating diversity (i.e., geographical distance). Selected items are marked with a darker (red) color.

to be such that $S' \supseteq S$. To achieve this, we select items from level ℓ or lower, since the items in S appear at all levels $m \leq \ell$. Analogously, for $k' < k$, to construct the set S' with the k' most diverse items such that $S' \subseteq S$, we may select those items of S that appear at levels higher than ℓ.

3.4 Diversity and Relevance

In many cases, the items in the result of a query are ranked, most often based on their relevance to the user query. In this case, diversification also addresses the over-specialization problem, i.e., retrieving results that are very similar to each other. An example is shown in Figure 3.5 using our apartments dataset, where relevance is defined based on price, i.e., the cheaper the apartment the more relevant, and diversity is based on geographical location. Using only relevance, a user is presented with apartments mostly from east London, while with diversity, some relatively cheap apartments from other regions in London are also selected.

The MaxMin *k*-Diversity problem with relevance. In general, the relevance score of an item is application dependent. Without loss of generality, we assume a relevance function $r : P \to \mathbb{R}^+$ that assigns a relevance score to each item, where a higher value indicates that the item is more relevant to a particular query or user. A natural bi-criteria objective seeks to maximize both the relevance and the diversity of the selected subset. In particular, the MaxMin k-Diversity With Relevance Problem for a positive integer k, $k \leq n$, is the problem of selecting a subset S^* of $\mathcal{P}$ such that:

$$S^* = \underset{\substack{S \subseteq P \\ |S|=k}}{\operatorname{argmax}} f_r(S, d, r) \tag{3.8}$$

with

$$f_r(S,d,r) = \min_{p_i \in S} r(p_i) + \lambda \min_{\substack{p_i,p_j \in S \\ p_i \neq p_j}} d(p_i,p_j) \tag{3.9}$$

where $\lambda > 0$ is a parameter that tunes the importance of diversification.

A combined relevance-diversity (d_r) approach. It was shown in [57] that the MaxMin k-Diversity With Relevance Problem is equivalent with the MaxMin k-Diversity Problem if we replace the distance function d with the function d_r:

$$d_r(\lambda, p_i, p_j) = \frac{1}{2}(r(p_i) + r(p_j)) + \lambda d(p_i, p_j). \tag{3.10}$$

If we define $d_r(\lambda, p_i, p_j) = 0$, for $p_i = p_j$, it is to easy to see that d_r is a metric, if d is a metric.

To incorporate relevance, we can now build the CTs using distance d_r instead of d. It is straightforward to see that all algorithms and related bounds advanced for the diversity-only case directly apply to the combined relevance-diversity case.

Supporting a varying λ. A drawback of the combined approach is that we need to maintain a different CT for each different value of λ. We would like to be able to adjust λ dynamically without having to reconstruct the trees. To this end, we consider building CTs based solely on distance d and enhancing our algorithms for selecting diverse sets so as to incorporate relevance in the selection.

Let C_{ℓ_k} be the highest level with at least k nodes. The enhanced Level-Basic algorithm selects the k most relevant items of C_{ℓ_k}, while the Level-Greedy algorithm performs a greedy selection among the corresponding items using the combined distance d_r, instead of d.

We also introduce a new level algorithm, called *Level-Hybrid*, whose goal is to allow nodes with large relevance scores that appear in low levels of the CT to enter the diverse set. *Level-Hybrid* uses an extended CT. In this extended CT, for each internal node $\mathcal{P}$, we maintain a pointer to the node that has the largest relevance score among all nodes in the subtree rooted at $\mathcal{P}$. Let $best(p)$ be this node. Level-Hybrid (Algorithm 3.8) performs a greedy selection among the k nodes from level C_{ℓ_k} whose descendants have the best relevance scores and these k descendants. Level-Hybrid performs $k \cdot 2k = 2k^2$ distance computations.

In the CT implementation of the greedy heuristic, subtrees are pruned based on both diversity and relevance. To this end, we maintain at each internal node $\mathcal{P}$, the largest relevance value, $w_r(p)$, called *relevance weight*, of any node in the subtree of $\mathcal{P}$. The best possible pruning is achieved, if we also use the distance weight. Using both weights, we have the following pruning rule.

WCT Pruning Rule With d_r: Let $\mathcal{P}$ and q be two nodes at level ℓ in a WCT. If $d_r(p,S) \geq d_r(q,S) + \frac{1}{2}(r(q) + w_r(q)) + \lambda w_d(q)$, we can prune the subtree rooted at q.

Clearly, we could maintain only the relevance weight, in which case the distance is bounded using the CT pruning rule.

Algorithm 3.8 Level-Hybrid Algorithm.

Input: A cover tree T, an integer k, a real number λ.

Output: A set S with the k most diverse items in T.

```
1: $\ell_k \leftarrow \ell_{max}$
2: while $|T.C_{\ell_k}| < k$ do
3:     $\ell_k \leftarrow \ell_k - 1$
4: $C \leftarrow$ the $k$ nodes $\mathcal{P}$ in $T.C_{\ell_k}$ with the most relevant $best(p)$
5: $candidates \leftarrow \emptyset$
6: for all $p \in C$ do
7:     $candidates \leftarrow candidates \cup \{p, best(p)\}$
8: $p^* \leftarrow \text{argmax}_{p \in candidates}\, r(p)$
9: $S \leftarrow S \cup \{p^*\}$
10: $candidates \leftarrow candidates \backslash \{p^*\}$
11: while $|S| < k$ do
12:     $p^* \leftarrow \text{argmax}_{p \in candidates}\, dr(\lambda, p, S)$
13:     $S \leftarrow S \cup \{p^*\}$
14:     $candidates \leftarrow candidates \backslash \{p^*\}$
15: return $S$
```

Maximal Marginal Relevance (MMR) Another popular approach for combining relevance and diversity is Maximal Marginal Relevance (MMR) (e.g., [23, 51]). MMR constructs a relevant and diverse subset S in a greedy fashion, by starting with either a random or the most relevant item and adding at each round the item p_i with the maximum contribution, i.e., the item p_i with the maximum quantity:

$$mr(\lambda, p_i, S) = \lambda r(p_i) + (1 - \lambda) \min_{p_j \in S} d(p_i, p_j) \tag{3.11}$$

where $\lambda \in [0, 1]$ is a parameter that tunes the relative importance of each of the two factors.

All the presented algorithms are directly applicable to MMR by using mr instead of d_r. For example, we now have the following pruning rule.

WCT Pruning Rule With MMR: Let $\mathcal{P}$ and q be two nodes at level ℓ in a WCT. If $d_r(p, S) \geq \lambda w_r(q) + (1 - \lambda)\ (d(q, S) + w_d(q))$, we can prune the subtree rooted at q.

3.5 Experimental Evaluation

In this section, we experimentally evaluate the performance of cover trees for dynamically computing diverse sets.

Datasets. We use a variety of datasets, both real and synthetic. Our *synthetic datasets* consist of two-dimensional points in the Euclidean space, where each dimension takes values in [0, 1]. Items are either uniformly distributed or form clusters of different

Table 3.1: Characteristics of the datasets.

Dataset	**Cardinality**	**Dimensions**	**Distance**	**Relevance scores**
Uniform	10,000	2	Euclidean	Uniform/Clustered
Clustered	10,000	2	Euclidean	Uniform/Clustered
Cities	5,922	2	Euclidean	Clustered
Nestoria	1,000	8	Haversine	Price-based
Faces	300	256	Cosine	Uniform
Flickr	1,000	-	Jaccard	Uniform

Table 3.2: Input parameters.

Parameter	**Range**		**Default**	
	Synthetic	**Real**	**Synthetic**	**Real**
Base (b)	1.2-2.2		1.6	
Diversification factor (λ)	0.0-1.0		0.2	
Dataset size (n)	1-10,000	300-5,922	4,000	–
Size of diverse set (k)	100-300	10-100	150	50
Window size (w)	1,000	100	(no window)	
Window jump step (h)	100-900	10-90	(no window)	

sizes. We assign relevance scores to items either uniformly or in a "clustered" manner around specific target items, so that items that are closer to the target items get larger relevance scores than items further away. Clustered assignment is used to model the common case where we get high relevance scores around specific items that correspond to different interpretations of the query. Thus, we get four combinations: (i) uniform spatial distribution with uniform relevance scores ("*Uniform-Uniform*"), (ii) uniform spatial distribution with clustered relevance scores centered around uniformly distributed target items ("*Uniform-Clustered*"), (iii) clustered spatial distribution with uniform relevance scores ("*Clustered-Uniform*") and (iv) clustered spatial distribution with clustered relevance scores around the centers of the spatial clusters ("*Clustered-Clustered*").

We also employ *four real datasets*. "*Nestoria*" consists of information about 1,000 apartments for sale in the London area retrieved from [6]. We relate relevance with price and consider cheaper apartments as more relevant, while similarity is measured based on geographic proximity (Haversine distance). "*Cities*" is a collection of geographical points representing 5,922 cities and villages in Greece [5]. We assign relevance scores in a clustered manner to model the fact that some specific areas may be more interesting than others. "*Faces*" consists of 256 features extracted from each of 300 human face images with the eigenfaces method [2] and uniformly distributed relevance scores. Finally, for "*Flickr*", we used data from [4] which consists of tags assigned by users to photographs uploaded to the Flickr photo service [3]. Table 3.1 summarizes our datasets, while Table 3.2 our parameters.

Our datasets capture result sets with different data characteristics. Concerning spatial distribution, for example, "*Uniform-Uniform*" contains items that cover all the available space, while "*Cities*" (due to the geography of Greece, which includes a large number of islands) provides us with both dense and sparse areas of items (Figure 3.3). "*Faces*" contains many distinct small dense areas, while "*Flickr*" is generally a very sparse dataset.

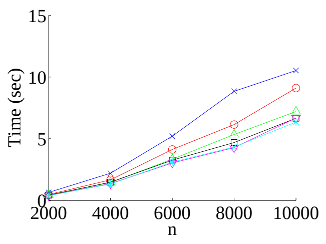
Setup. All methods are implemented in Java using JDK 1.6. Our experiments were executed on an Intel Core i3-2100 3.1GHz PC with 3GB of RAM.

3.5.1 Building and Maintaining Cover Trees

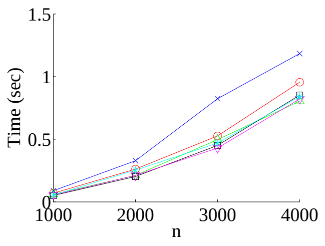
First, we evaluate the cost of building cover trees. Figure 3.6 shows the real-time cost of building an ICT by incrementally inserting items. This cost depends on b, since smaller values of b lead to new items being inserted in lower tree levels, thus increasing the cost of individual insertions. The cost also depends on the distance metric used, since some distance computations are more expensive. For example, inserting 1,000 items of the "*Flickr*" dataset, using the Jaccard distance, takes up to 5 seconds, while inserting the same number of items takes less than 0.1 seconds for our Euclidean datasets. The results are similar for the omitted datasets.

The cost of building a BCT can be divided into (i) the cost of selecting items from the leaf level to build the first non-leaf level and (ii) the cost of assigning nodes to suitable parents. Table 3.3 shows these costs for the uniform dataset. The cost of step (i) is the same as the cost of executing the greedy heuristic for $k = n$ and is independent of b or the dataset distribution. The cost of step (i) dominates that of step (ii) and this is why the total building cost for BCTs does not differ significantly with b or with spatial distribution. Building BCTs is orders of magnitude more expensive than building the corresponding ICTs for the same datasets.

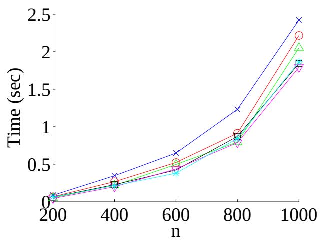
Figure 3.7 depicts the size of ICTs and BCTs for different values of b ($n = 4,000$ for our synthetic datasets). The x-axis corresponds to the tree level, starting from 0 which denotes the root level, while the y-axis corresponds to the width (i.e., number of nodes) of the corresponding level. Smaller values of b lead to taller and narrower trees. Further, although ICTs are constructed incrementally, the resulting trees have almost identical structure with the corresponding BCTs. In general, the height of the tree depends on the minimum and maximum pairwise distances in the dataset, while the width of the levels depends on the spatial distribution of the data. Therefore, for example, levels get narrower faster as we move up the tree for "*Cities*" rather than for '*Uniform*", even though their height is roughly the same, since "*Cities*" is a more



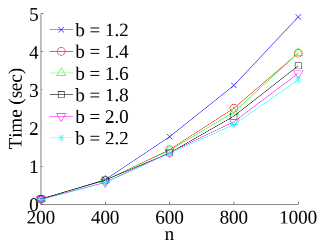
(a) Uniform.



(b) Cities.

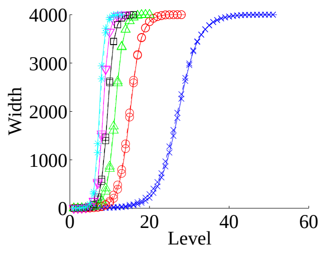


(c) Nestoria.

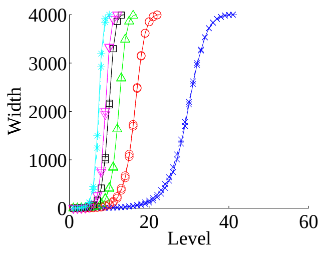


(d) Flickr.

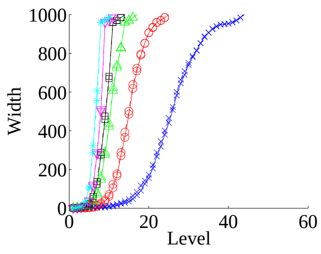
Figure 3.6: ICT building cost. The y-axis corresponds to the total time to incrementally insert all n items in the tree.



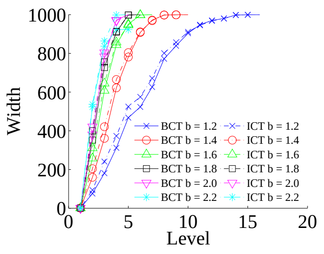
(a) Uniform.



(b) Cities.



(c) Nestoria.



(d) Flickr.

Figure 3.7: Tree sizes of the constructed ICTs and BCTs (full implicit representation). The drawn lines in each figure correspond to smaller b values as we move from left to right.

clustered dataset. Similarly, a wide tree is constructed for "*Flickr*", due to sparsity.

In terms of maintenance, a single insertion in an ICT costs 1 msec for trees up to 5,000 and up to 1.3 msec for trees with 10,000 items. The cost of deletions is higher because, after a node is removed, its children have to be re-assigned to new parents. For all datasets and b values, a single deletion requires less than 3 msec for trees up to 5,000 and less than 7 msec for trees with 10,000 items. We also measured the cost of maintaining weights in the case of WCTs which may require some extra bookkeeping to update weights. For all datasets and b values, 4-6 additional nodes where accessed per insertion on average. The effect on execution time is negligible.

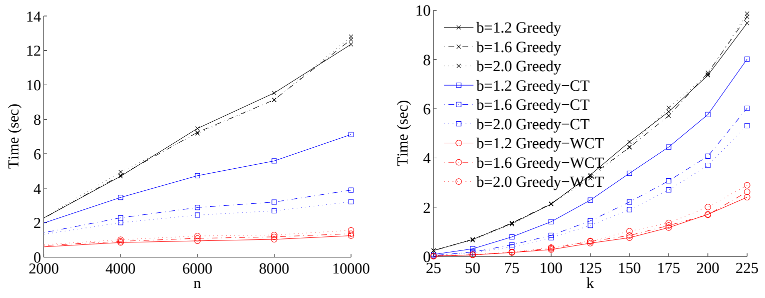
3.5.2 Computing Diverse Subsets

We next evaluate the performance of the various algorithms introduced in this chapter in terms of the quality of the retrieved diverse sets and the computational cost.

Diversity Algorithms. We first measure the cost savings when applying the CT PRUNING RULE ("Greedy-CT") or WCT PRUNING RULE ("Greedy-WCT") on an ICT vs. executing our cover tree based implementation of the greedy heuristic ("Greedy"). We use our uniform dataset to see how pruning improves the cost of Greedy with n and k and different values of b (Figure 3.8). Clearly, Greedy-WCT is more effective, since the actual distance to the

Table 3.3: BCT building cost (sec).

Uniform				
n	**step (i)**	**step (ii) *b*=1.2**	**step (ii) *b*=1.6**	**step (ii) *b*=2.0**
1,000	11.181	0.048	0.044	0.043
2,000	107.103	0.211	0.209	0.203
3,000	416.660	0.498	0.416	0.398
4,000	899.799	0.812	0.503	0.490



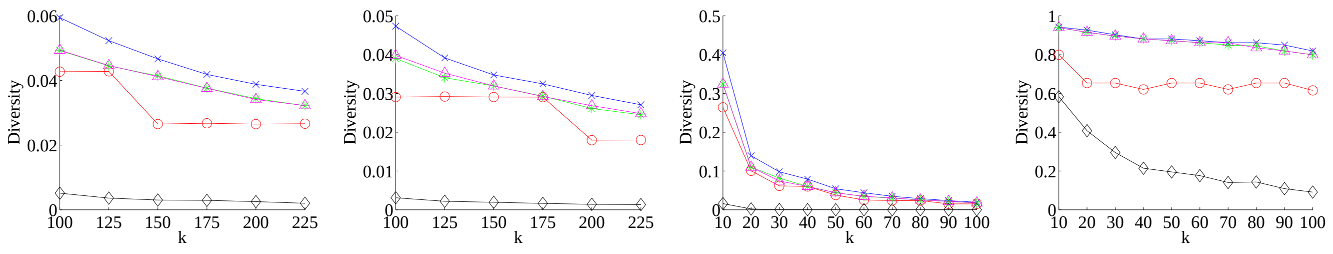
(a) Uniform (k = 150). (b) Uniform (n = 4000).

Figure 3.8: Pruning for the diversity-only case.

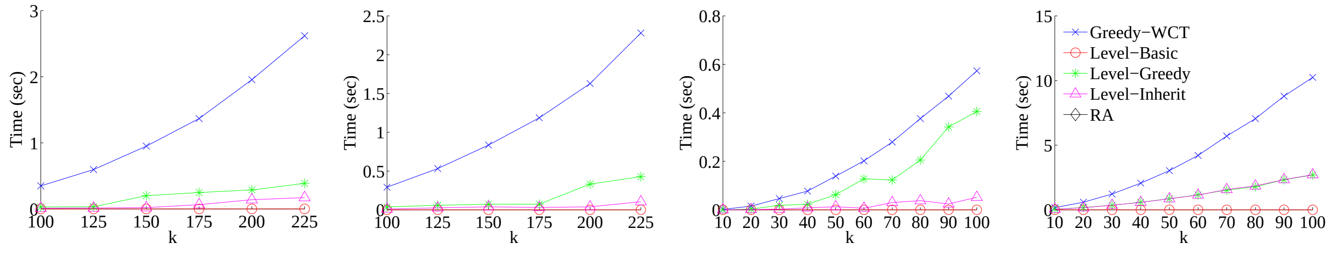
furthest descendant of each node is used for pruning. In general, pruning works better for non uniform datasets, since each selection of a diverse item results in pruning a largest number of items around it.

Next, we experimentally compare the performance of the greedy heuristic, using the Greedy-WCT implementation, and our Level algorithms, i.e., Level-Basic, Level-Greedy and Level-Inherit. Figure 3.9 depicts the achieved diversity and corresponding cost when varying k. For comparison, we also report the diversity attained by randomly selecting k of the n items (RA). Clearly, the larger the k, the less diverse the selected subset. The comparative performance of all algorithms is the same for all types of datasets. Specifically, for all datasets, Greedy-WCT achieves the best diversity at the highest cost, Level-Basic achieves the worst diversity at the lowest cost, while the other two Level algorithms lie in-between. Level-Inherit achieves similar diversity with Level-Greedy but is faster.

The Level algorithms select items from the appropriate tree level. Thus, their performance depends on the tree. Recall that, clustered datasets result in trees whose levels get narrower faster as we move up the tree. Level-Greedy and Level-Inherit perform a greedy selection among the items in the appropriate level, thus the wider the level, the worst the complexity and the better the diversity achieved. This also explains why their cost increases in “steps” as k increases, since we gradually select items from lower (and wider) levels. Level-Basic just selects any k items, thus the cost does not increase with



(a) Uniform. (b) Cities. (c) Faces. (d) Flickr.



(e) Uniform. (f) Cities. (g) Faces. (h) Flickr.

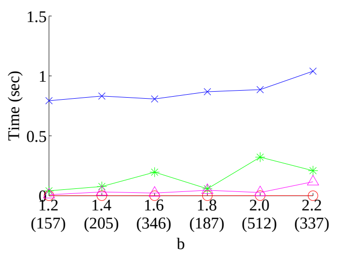
Figure 3.9: Diversity and cost for the diversity-only case with varying k.

k, while the achieved diversity decreases more rapidly as k increases, since items are selected randomly instead of greedily from wider levels.

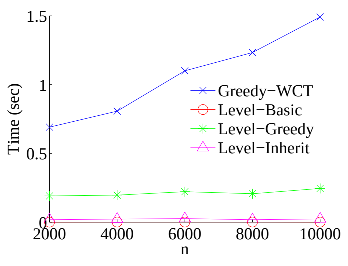
Figure 3.10(a) shows how the cost varies with b. Smaller b values may increase the building cost of the tree (Figure 3.6) but also lead to faster diverse set computation, especially for Greedy-WCT. The cost of Level-Greedy and Level-Inherit depends on the size of the level from which items are selected. Figure 3.10(b) shows how the cost scales along n. All Level algorithms scale very efficiently with n, since they depend only on the size of the corresponding tree level.

Diversity with Relevance Algorithms. Let us first evaluate pruning (Figure 3.11). In the following, we report results for MMR (similar results are attained for d_r). We consider two rules for pruning: using only relevance weights (denoted "Greedy-CT") and using both relevance and distance weights (denoted "Greedy-WCT"). Again, using distance weights improves pruning especially for small values of λ (i.e., emphasis on diversity). Pruning is more effective for clustered relevance scores, since in this case, there are large subtrees with no relevant items that are pruned early. For the same reason, pruning generally performs better for very large values of λ. Finally, pruning is less effective for "*Flickr*" whose trees are shorter due to its sparsity.

We next compare Greedy-WCT with the Level algorithms. We also consider a CT variation, called *Priority Cover Tree (PCT)* introduced in [18] for computing priority medoids. A PCT is a CT which in addition to the three invariants of a CT, satisfies a fourth one that requires each node of the tree to have relevance score larger than or equal to the scores of all nodes in its subtree. To construct a PCT so that the fourth invariant is satisfied, items need to be inserted in descending order of relevance. In general, PCTs cannot be built incrementally. To illustrate, we present a simple example that shows

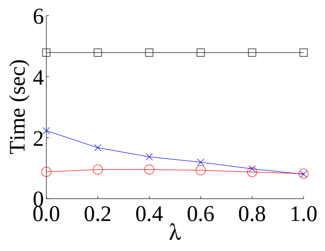


(a) Uniform.

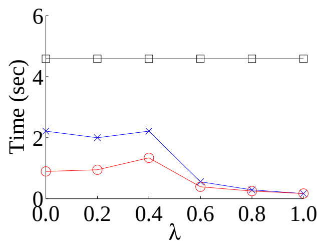


(b) Uniform.

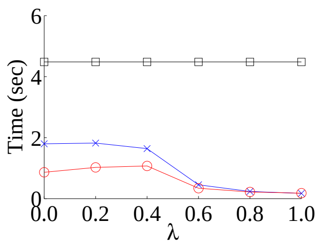
Figure 3.10: Cost for the diversity-only case when varying (a) b (the numbers in parentheses are the sizes of the highest level with at least k items) and (b) n.



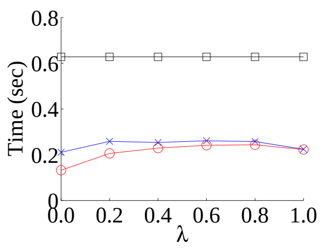
(a) Uniform-Uniform.



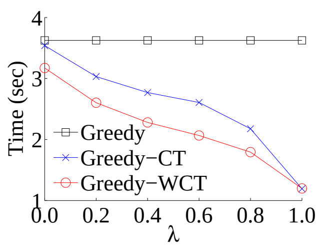
(b) Uniform-Clustered.



(c) Cities.



(d) Faces.

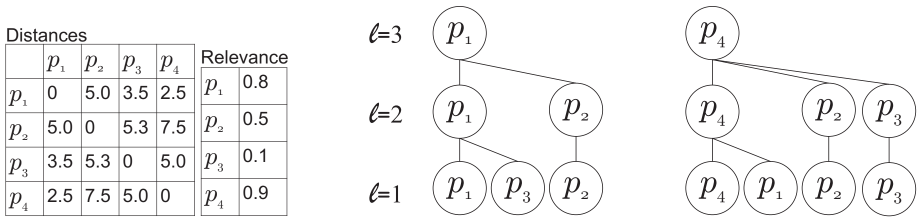


(e) Flickr.

Figure 3.11: Pruning for the diversity with relevance (MMR) case with varying λ.

that the arrival of a single item may change the relations among all nodes in a PCT. Consider the PCT of Figure 3.12(b) with $b = 2.0$ and the relevance scores and distances of Figure 3.12(a). This PCT is unique for p_1, p_2, p_3, since p_1 must appear at the top due to its relevance and p_3 cannot be covered by p_2 at ℓ = 2. Assume that p_4 arrives. Since p_4 has the largest relevance score, it must appear at the top of the tree. p_1 is not separated from p_4 at levels ℓ = 3 and ℓ = 2, therefore, it cannot appear there. p_2 and p_3 are separated from p_4 at ℓ = 2 and are placed at this level. The resulting PCT is shown in Figure 3.12(c). Notice that all pre-existing nodes p_1, p_2, p_3 now have different parent and children nodes than before the arrival of p_4, which means that the tree is in effect re-built from scratch.

Figure 3.13 shows the relevance, diversity and cost of the various algorithms when varying λ. We report results for the faster greedy heuristic, i.e., Greedy-WCT and the three Level algorithms (namely, Level-Basic, Level-Greedy and Level-Hybrid) applied on



Distances

	p_1	p_2	p_3	p_4
p_1	0	5.0	3.5	2.5
p_2	5.0	0	5.3	7.5
p_3	3.5	5.3	0	5.0
p_4	2.5	7.5	5.0	0

Relevance

p_1	0.8
p_2	0.5
p_3	0.1
p_4	0.9

(a) Items. (b) Before insertion. (c) After insertion.

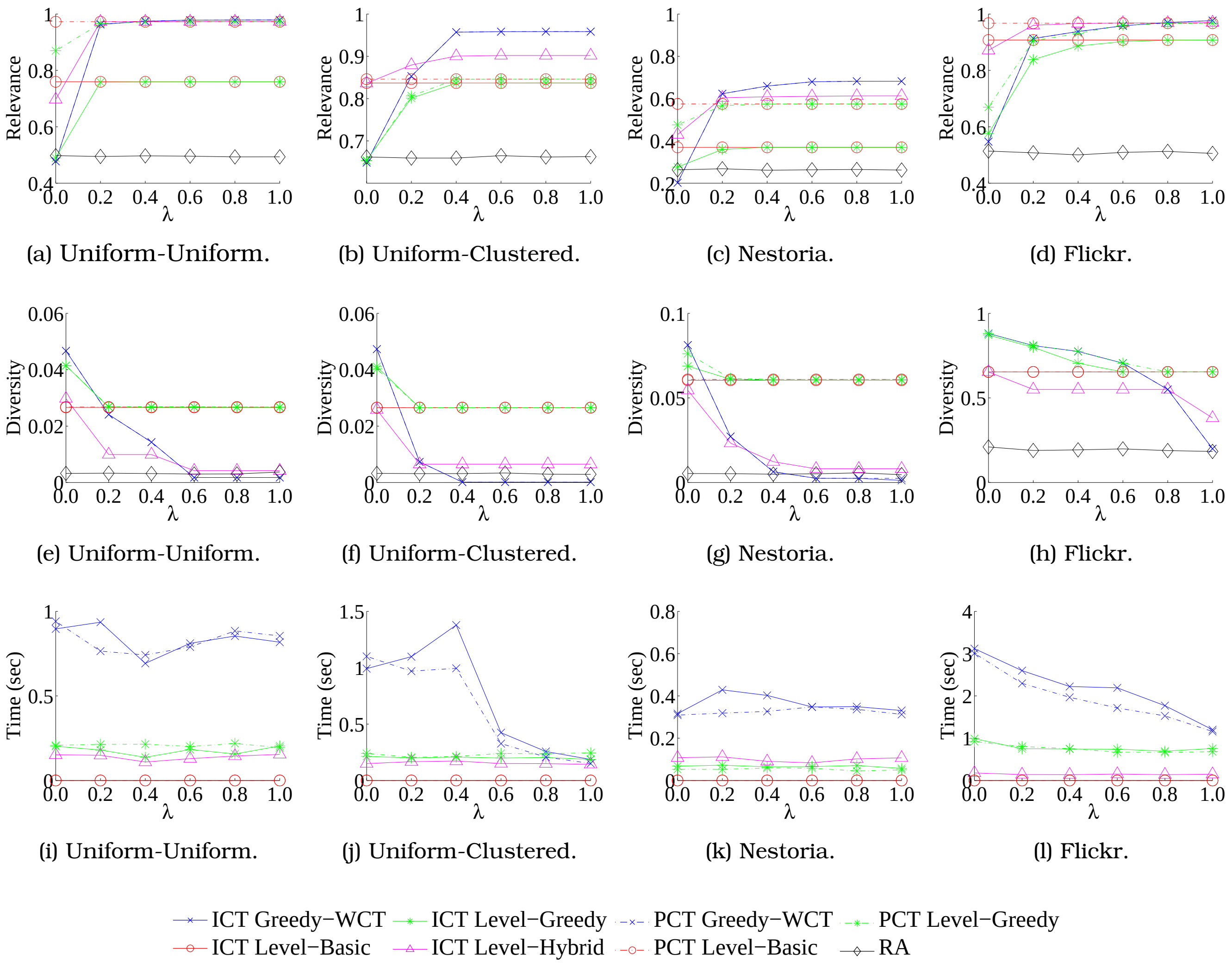
Figure 3.12: The arrival of p_4 changes the relations among all nodes of the PCT.

an ICT and a PCT for two synthetic and two real datasets. Note that due to the fourth invariant of the PCT, $best(p) = \mathcal{P}$ for every node $\mathcal{P}$ of a PCT, thus Level-Hybrid is the same with Level-Basic for PCTs and it is not depicted.

In terms of diversity and relevance, for all datasets, Level-Hybrid is the one closer to the greedy heuristic (which provides a good approximation of the optimal solution). Level-Hybrid achieves such results with much smaller cost. Among the Level algorithms, Level-Basic is clearly the fastest. Level-Hybrid performs a greedy selection among $2k$ items, while Level-Greedy performs a greedy selection among $|C_{\ell_k}|$ items, where ℓ_k is the highest level with at least k items. Therefore, the relative cost of Level-Greedy when compared with Level-Hybrid depends on the size of the level with regards to k. For example, for "*Flickr*", which has much wider levels than the other datasets, Level-Hybrid has lower cost than Level-Greedy. Note also that the cost of the level algorithms does not depend on λ. The quality and cost of the PCT solutions does not differ substantially from those of the ICT. Only pruning is slightly more efficient, since larger relevance scores appear at high levels. Due to space limitations, we omit the results for the rest of our datasets. "*Clustered-Uniform*" and "*Clustered-Clustered*" behave similarly to "*Uniform-Uniform*" and "*Uniform-Clustered*" respectively, while "*Cities*" has similar behavior to "*Uniform-Clustered*" and "*Faces*" to "*Clustered-Uniform*".

Continuous *k*-Diversity. We next focus on streaming arrivals of items and on how the application of our continuity requirements affects the retrieved solutions. We show results for "*Nestoria*", where we use the actual apartment upload time as the time in which items enter the stream. We also use the "*Clustered-Uniform*' dataset which has the most different distribution. For "*Clustered-Uniform*', the items that enter the stream are selected in a random manner.

Figure 3.14 reports results for the Unconstrained and the Constrained k-Diver- sity Problem. We vary the jump step h of the window and fix the other parameters to study the behavior of the algorithms as the number of valid diverse items from the previous window changes. We report average values over all windows as the window slides along the stream of items. In most cases, the constrained variations achieve similar relevance and diversity with the unconstrained alternatives. For all algorithms performing greedy computations, the constrained variations are executed faster, since the diverse subset of each window is initialized with the valid items of the diverse subset from the previous



(a) Uniform-Uniform. (b) Uniform-Clustered. (c) Nestoria. (d) Flickr.

(e) Uniform-Uniform. (f) Uniform-Clustered. (g) Nestoria. (h) Flickr.

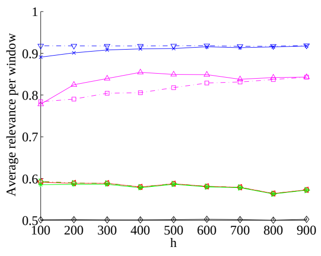
(i) Uniform-Uniform. (j) Uniform-Clustered. (k) Nestoria. (l) Flickr.

Figure 3.13: Relevance (top row), diversity (middle row) and cost (bottom row) for the MMR case with varying λ.

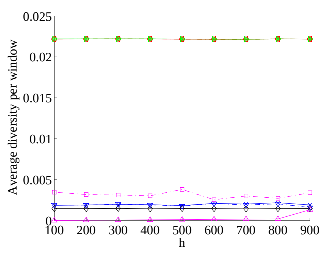
window, and thus, fewer computations are required. Level-Basic is unaffected, since it does not involve any greedy steps. Besides cost savings, another important aspect of the constrained variations is the higher sense of continuity between subsequent diverse sets seen by the users. To quantify this, we use the Jaccard similarity between the acquired diverse sets. The Jaccard similarity of two sets of items S_1, S_2 is defined as:

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

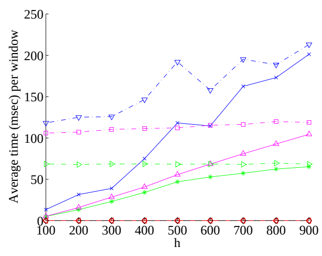
The higher the Jaccard similarity of two sets, the more common items the two sets share. In Figure 3.14, we see that the constrained variations exhibit higher Jaccard similarity.



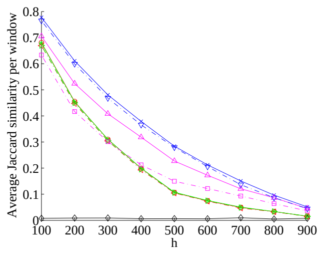
(a) Clustered-Uniform.



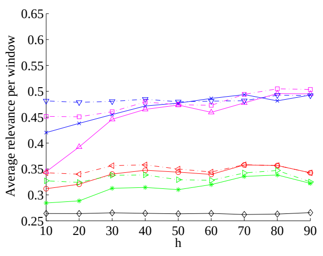
(b) Clustered-Uniform.



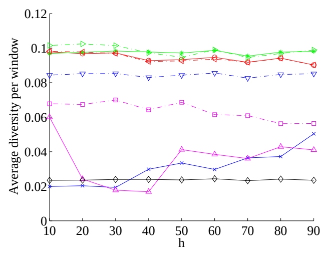
(c) Clustered-Uniform.



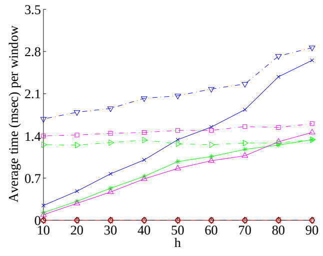
(d) Clustered-Uniform.



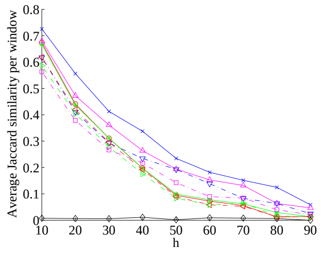
(e) Nestoria.



(f) Nestoria.



(g) Nestoria.



(h) Nestoria.

(C) Greedy−WCT (C) Level−Greedy (U) Greedy−WCT (U) Level−Greedy RA
(C) Level−Basic (C) Level−Hybrid (U) Level−Basic (U) Level−Hybrid

Figure 3.14: Relevance ((a)/(e)), diversity ((b)/(f)), cost ((c)/(g)) and Jaccard similarity ((d)/(h)) for the Unconstrained (U) and Constrained (C) MMR case in a streaming setting when varying h ($k = 150$ (resp. $k = 15$), $w = 1000$ (resp. $w = 100$) for the synthetic (resp. real) dataset, $b = 1.6$, $\lambda = 0.2$).

3.6 Comparison with Related Work

There are a couple of approaches in the related literature that consider indexing to assist diversification. Most such works consider structured data. Relational data are considered in [107]. Attributes are *totally ordered* by importance in terms of diversity, so that two tuples that differ in a highly important attribute are considered highly diverse, even if they share common values in other less important attributes. This diversity measure allows the exploitation of a Dewey encoding of tuples that enables a tree structure which is later exploited to select the k most diverse tuples. Contrary to our approach, the proposed method is limited to this specific diversity measure and cannot be applied in the general case.

Spatial index are often exploited to locate those relevant nearest neighbors of an item that are the most distant to each other (e.g., [59]). Our work is different since our goal is not to locate the nearest diverse neighbors of a single object but rather to locate a relevant and diverse subset of all available items.

Cover trees are employed in [79] for solving the k-medoids problem. While locating k representative medoids is a form of diversification, that work focuses on the clustering of data rather than their diversification. A variation of cover trees, called Priority Cover

Trees (PCTs), were employed in [18, 17] for computing priority medoids, i.e., medoids having a high relevance factor. Besides solving a different problem, this approach cannot be employed in dynamic environments, since all available items must be known in advance for building PCTs. Our method, can handle dynamic insertions and deletions and provide an evolving diverse set of items.

The related literature focusing on continuous data is limited. None of the existing proposals considers an index-based approach to diversification as we do here. In our previous work [41], we evaluated various heuristics in case of continuous data, and a greedy heuristic that enforces durability was shown to outperform the other methods. A method based on interchange heuristics is proposed in [84]. Upon the arrival of a new item p, all possible interchanges between p and the items in the current solution are performed and p replaces an item in the solution, if this replacement increases diversity. A similar technique was also proposed in [46]. However, with these methods, old items do not expire, and a new item may enter the solution only upon its arrival.

The MaxSum diversification problem is studied in [19], in the setting of streaming data and monotone submodular diversification functions. A $1/2$-approximation greedy algorithm is proposed which is faster than the usual greedy heuristic. Dynamic updates are also considered in the sense that when the underlying set of available items changes, interchanges are attempted to improve the computed solution. Our approach considers a different diversification problem, i.e., MaxMin, and is not restricted to monotone submodular functions. Finally, the online version of the diversity problem is considered in [86], that is, selecting a diverse subset without knowing the complete set of items.

3.7 Summary

Most current research addresses the static version of the diversification problem. In this chapter, we have studied the diversification problem in a dynamic setting where the items to be diversified change over time. We have proposed an index-based approach that allows the incremental evaluation of the diversified sets to reflect item updates. Our solution is based on cover trees. We have provided theoretical and experimental results regarding the quality of our solution.

CHAPTER 4

DISC DIVERSITY: RESULT DIVERSIFICATION BASED ON DISSIMILARITY AND COVERAGE

RESULT diversification has attracted considerable attention as a means of enhancing the quality of the query results presented to users (e.g., [107, 119]). 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; they can be roughly categorized [42] as based on: (i) *content* (or *similarity*), i.e., items that are dissimilar to each other (e.g., [119]), (ii) *novelty*, i.e., items that contain new information when compared to what was previously presented (e.g., [35]) and (iii) *coverage*, i.e., items that belong to different categories or topics (e.g., [11]).

Most previous approaches to diversification rely on assigning a diversity score to each item and then selecting as diverse either the k items with the largest score for a given k (e.g., [15, 24]), or the items with score larger than some predefined threshold (e.g., [113]). Diversity is often combined with other ranking criteria, such as *relevance* (e.g., [99]). In this case, the selected items must be both highly relevant individually and diverse as a set. The two criteria are often conflicting with each other, since the items most relevant to a specific user need are often similar to each other. A number of different approaches have been proposed to achieve a trade-off between high relevance and high diversity (e.g., [23, 57]), usually based on assigning weights to the two factors, resulting again in a unified score and a corresponding top-k or threshold problem.

Here, we address diversity through a different perspective. In contrary to previous approaches, we aim at selecting a representative subset of the result set that contains items that are *both* dissimilar with each other *and* cover the whole result set. Let $\mathcal{P}$ be the set of items in a query result, d a distance metric, and r a real number, that we call *radius*. We consider that two items p_i, p_j in $\mathcal{P}$ are similar to each other, if and only if, $d(p_i, p_j) \leq r$. We also say that they cover each other. Our goal is to select a subset S of $\mathcal{P}$, such that (i) for each item $p_i \in \mathcal{P}$, there is at least one item p_j in $\mathcal{P}$, such that $d(p_i, p_j) \leq r$, and (ii) for any pair of items, p_i, $p_j \in S$, it holds $d(p_i, p_j) > r$. The first condition ensures that all items in $\mathcal{P}$ are represented, or covered, by at least one similar item in the selected subset. The second condition ensures that the selected items are dissimilar to each other. We call the set S *Dissimilar and Covering* subset or *DisC* diverse subset.

A novel aspect of our approach is that, 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 a smaller, more diverse subset, while decreasing r results in a larger, less diverse subset. We call these operations *zooming-out* and *zooming-in* respectively. At one extreme, a radius equal to the diameter of the result set gives a singleton diverse subset. At the other extreme, a radius smaller than the smallest pairwise distance in the result set gives a diverse subset equal to the original result set.

Since there may be more than one DisC diverse subset of a result set, for attaining a concise representation, we aim at selecting the one with the smallest number of items, termed *Minimum r-DisC diverse subset*. Furthermore, when the items in the result set are associated with weights, besides the size, we take weights into account and select a *Minimum Weighted r-DisC diverse subset*. When all weights are equal, a Minimum Weighted r-DisC diverse subset reduces to a Minimum r-DisC diverse subset. As an example, Figure 4.1(a) and Figure 4.1(b) depict the selected diverse subset for a set of items representing major cities in our world, without and with weights respectively. In this example, weights were set based on population.

Further, we would like to allow different areas of the result set to contribute more or less items in the diverse subset. To this end, we extend the definition of DisC diverse subsets to allow each item p_i to be associated with a different radius $r(p_i)$. The radius of an item may depend on its relevance to the query, on the density of its surrounding

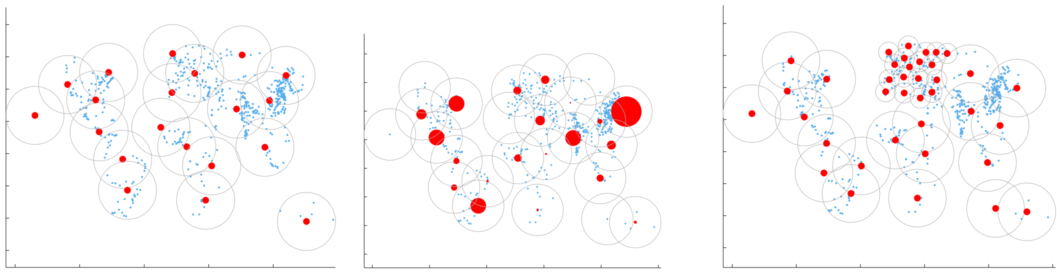

(a) No weights. (b) Weights. (c) Multiple radii.

Figure 4.1: DisC diversity: (a) no weights, single radii, (b) weights, single radii, and (c) no weights, multiple radii. Selected items are shown as solid circles with size proportional to their weight. (Non solid) circles denote the radius around the selected items.

area, or other factors. Figure 4.1(c) depicts the selected subset of the world cities example in the case of multiple radii, where a smaller radii is used for cities in Europe, resulting in more items being selected from this area.

We formalize the problem of locating minimum DisC diverse subsets as an independent dominating set problem on graphs. In the case of a single radius, the corresponding graph is undirected, whereas in the case of multiple radii, the corresponding graph is directed. We show that, locating a DisC diverse subset is equivalent to locating an independent and dominating set for the corresponding graph. However, for directed graphs, there are graphs for which there is no independent and dominating set. We show that for the graphs modeling the DisC problem, there is always such a set. Locating a minimum independent and dominating set is an NP-hard problem. We provide a suite of greedy algorithms for locating approximate solutions along with bounds for the size of the produced diverse subsets.

Then, we consider the problem of incrementally adjusting the radius r, or zooming. We explore the relation among DisC diverse subsets of different radii and provide algorithms for adapting an already computed DisC diverse subset to a new radius along with corresponding theoretical upper bounds for the size of the diverse subsets produced. Figure 4.2 shows an example of zooming-in and zooming-out.

Although the examples presented so far concern two-dimensional points, DisC diversity is applicable to any type of data set, including the case of non-categorical attributes, as long as an appropriate distance d is provided. As an example, consider searching for cameras, where diversity refers to cameras with different features. Figure 4.3 depicts an initial most diverse result and the result of zooming-in one individual camera in this result.

Since the crux of the efficiency of all proposed algorithms is locating neighbors, we take advantage of spatial data structures. In particular, we propose efficient implementations based on the M-tree [33]. We evaluate our algorithms for the different DisC diversity versions using both real and synthetic datasets and draw various conclusions

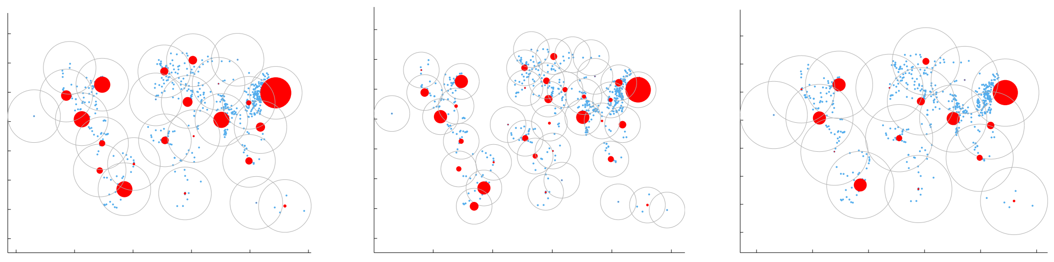

(a) Initial set. (b) Zooming-in. (c) Zooming-out.

Figure 4.2: Zooming. Selected items are shown as solid circles with size proportional to their weight. (Non solid) circles denote the radius around the selected items.

regarding their effectiveness and efficiency.

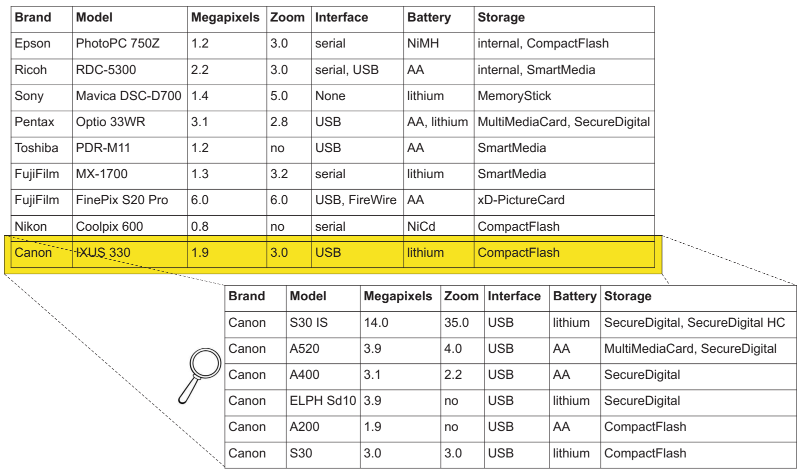
In a nutshell, we make the following contributions:

- we use a new, intuitive definition of diversity, called DisC diversity, based on using a radius r rather than a size limit k and extend it to support a different radius for each item,
- in addition to the geometrical interpretation of DisC diversity, we present an equivalent graph-based model of the problem,
- we introduce incremental diversification through zooming-in and zooming-out,
- we show that locating DisC diverse subsets is an NP-hard problem, provide efficient algorithms for their computation along with theoretical approximation bounds, present efficient M-tree tailored implementations and experimentally evaluate their performance, and
- we compare DisC diversity with other popular diversity models, both analytically and qualitatively.

The rest of this chapter is structured as follows. Section 4.1 introduces DisC diversity and Section 4.2 algorithms for computing diverse subsets, Section 4.3 introduces incremental diversification and Section 4.4 compares our approach with other diversification methods. In Section 4.5, we employ the M-tree for the efficient implementation of our algorithms, while in Section 4.6, we present experimental results. Finally, Section 4.7 presents related work and Section 4.8 concludes the chapter.

4.1 DisC Diversity

In this section, we first provide a formal definition of DisC diversity. We define the Minimum r-DisC and Minimum Weighted r-DisC diverse subsets and provide various theoretical bounds for the size of an r-DisC diverse subset with regards to the minimum



Brand	Model	Megapixels	Zoom	Interface	Battery	Storage
Epson	PhotoPC 750Z	1.2	3.0	serial	NiMH	internal, CompactFlash
Ricoh	RDC-5300	2.2	3.0	serial, USB	AA	internal, SmartMedia
Sony	Mavica DSC-D700	1.4	5.0	None	lithium	MemoryStick
Pentax	Optio 33WR	3.1	2.8	USB	AA, lithium	MultiMediaCard, SecureDigital
Toshiba	PDR-M11	1.2	no	USB	AA	SmartMedia
FujiFilm	MX-1700	1.3	3.2	serial	lithium	SmartMedia
FujiFilm	FinePix S20 Pro	6.0	6.0	USB, FireWire	AA	xD-PictureCard
Nikon	Coolpix 600	0.8	no	serial	NiCd	CompactFlash
Canon	IXUS 330	1.9	3.0	USB	lithium	CompactFlash

Brand	Model	Megapixels	Zoom	Interface	Battery	Storage
Canon	S30 IS	14.0	35.0	USB	lithium	SecureDigital, SecureDigital HC
Canon	A520	3.9	4.0	USB	AA	MultiMediaCard, SecureDigital
Canon	A400	3.1	2.2	USB	AA	SecureDigital
Canon	ELPH Sd10	3.9	no	USB	lithium	SecureDigital
Canon	A200	1.9	no	USB	AA	CompactFlash
Canon	S30	3.0	3.0	USB	lithium	CompactFlash

Figure 4.3: Zooming-in a specific camera.

ones. Then, we extend our definition of DisC diversity to support a different radius for each item. Finally, we present a graph based model of DisC diversity and show that locating a DisC diverse subset is equivalent to finding an independent and dominating set of the corresponding graph.

4.1.1 Definition of DisC Diversity

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

We define similarity between two items using a distance metric $d : \mathcal{P} \times \mathcal{P} \rightarrow \mathbb{R}^+$. For a real number r, $r \geq 0$, we use $N_r(p_i)$ to denote the set of *neighbors* (or, the *neighborhood*) of an item $p_i \in \mathcal{P}$, i.e., the items lying at distance at most r from p_i:

$$N_r(p_i) = \{p_j \mid p_i \neq p_j \wedge d(p_i, p_j) \leq r\} \tag{4.1}$$

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. Items in the neighborhood of p_i are considered similar to p_i, while items outside its neighborhood are considered dissimilar to p_i. We define an r-DisC diverse subset as follows:

Definition 4.1. (r-DisC Diverse Subset) Let $\mathcal{P}$ be a set of items and r, $r \geq 0$, a real number. A subset S of $\mathcal{P}$ is an r-Dissimilar-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 $d(p_i, p_j) > r$.

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

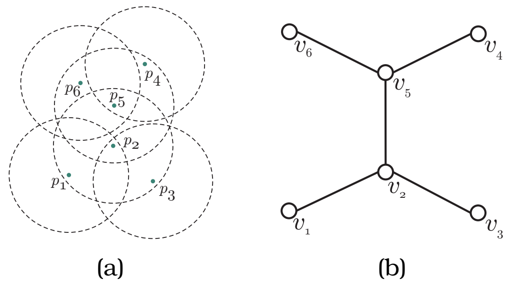


Figure 4.4: (a) A set of items: the (minimum) r-DisC diverse subset $\{p_2, p_4, p_6\}$ is preferred over the larger r-DisC diverse subset $\{p_1, p_3, p_4, p_6\}$, (b) their graph representation.

There may be more than one dissimilar and covering diverse subsets for the same set of items $\mathcal{P}$. Since we want a concise representation of $\mathcal{P}$, we select the smallest one (see Figure 4.4(a) for an example). Formally, we define the Minimum r-DisC diverse subset problem as follows:

Definition 4.2. (MINIMUM r-DISC DIVERSE SUBSET PROBLEM) Given a set $\mathcal{P}$ of items and a radius r, $r \geq 0$, 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 $f(S^*) \leq f(S)$, where $f(S) = |S|$.

Often, items are associated with a *weight* indicating their importance under some specific context, e.g., satisfying some specific user information need. We use $w(p_i)$ to denote the weight of p_i. Larger weights indicate items of higher importance. For simplicity, we consider that all weights are in $(0, 1]$. Now, given $\mathcal{P}$, we want to select items that are both diverse to each other and also highly relevant. We define the Minimum Weighted r-DisC diverse subset problem as follows:

Definition 4.3. (MINIMUM WEIGHTED r-DISC DIVERSE SUBSET PROBLEM) Given a set of items $\mathcal{P}$, with each item $p_i \in \mathcal{P}$ associated with a weight $w(p_i)$, and a radius r, $r \geq 0$, find a DisC diverse subset S^* of $\mathcal{P}$, such that, for every DisC diverse subset S of $\mathcal{P}$, it holds that $f(S^*) \leq f(S)$, where

$$f(S) = \sum_{p_i \in S} \frac{1}{w(p_i)}$$

If we consider all weights equal, i.e., when we are only interested in the diversity of the selected set and not the individual weights of the selected items, then the Minimum Weighted r-DisC diverse subset problem is reduced to the Minimum r-DisC diverse subset problem, i.e., locating the minimum sized subset of dissimilar items that can cover the available space. Between two subsets of equal size, Definition 4.3 selects the one with the largest sum of weights.

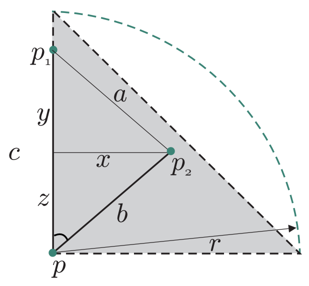


Figure 4.5: Manhattan independent neighbors.

4.1.2 General Bounds

Next, we present a number of theoretical results concerning the size of an r-DisC diverse subset. In the following, we use the terms dominance and coverage, as well as, independence and dissimilarity interchangeably. In particular, two items p_i and p_j are independent, if $d(p_i, p_j) > r$. We also say that an item *covers* all items in its neighborhood.

Theorem 4.1. *Let B be the maximum number of independent neighbors of any item in $\mathcal{P}$. Any r-DisC diverse subset S of $\mathcal{P}$ is at most B times larger than any minimum r-DisC diverse subset S^*.*

Proof. Since S is an independent set, any item in S^* can cover at most B items in S and thus $|S| \leq B|S^*|$. ∎

Note that, since a minimum weighted r-Disc subset for $\mathcal{P}$ cannot be smaller than a minimum r-DisC diverse subset $\mathcal{P}$, it also holds that any r-DisC diverse subset S of $\mathcal{P}$ is at most B times larger than any minimum weighted r-DisC diverse subset S^*.

The value of B depends on the distance metric used and also on the dimensionality dim 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 and data dimensionality.

Lemma 4.1. *If d is the Euclidean distance and $dim = 2$, each item p_i in $\mathcal{P}$ has at most $B = 5$ neighbors that are independent from each other.*

Proof. Let p_1, p_2 be two independent neighbors of p. Then, it must hold that $\angle p_1pp_2$ is larger than $\frac{\pi}{3}$. Otherwise, $d(p_1, p_2) \leq \max\{d(p, p_1), d(p, p_2)\} \leq r$ which contradicts the independence of p_1 and p_2. Therefore, p can have at most $(2\pi/\frac{\pi}{3}) - 1 = 5$ independent neighbors. ∎

Lemma 4.2. *If d is the Manhattan distance and $dim = 2$, each item p_i in $\mathcal{P}$ has at most $B = 7$ neighbors that are independent from each other.*

Proof. Let p_1, p_2 be two independent neighbors of p. Then, it must hold that $\angle p_1pp_2$ (in the Euclidean space) is larger than $\frac{\pi}{4}$. We will prove this using contradiction. p_1, p_2 are neighbors of p so they must reside in the shaded area of Figure 4.5. Without loss

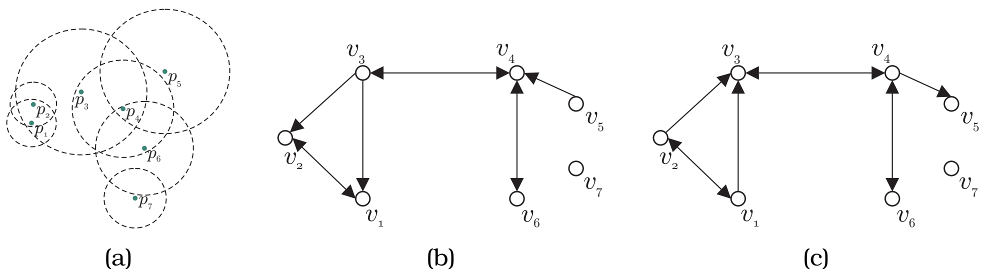


Figure 4.6: (a) A set of items associated with different radii and their graph representation for the (b) Covering and (c) CoveredBy problems. A directed edge from v_i to v_j indicates that $d(p_i, p_j) \leq r(p_i)$ and $d(p_i, p_j) \leq r(p_j)$ respectively.

of generality, assume that one of them, say p_1, is aligned to the vertical axis. Assume that $\angle p_1 p p_2 \leq \frac{\pi}{4}$. Then $\cos(\angle p_1 p_i p_2) \geq \frac{\sqrt{2}}{2}$. It holds that $b \leq r$ and $c \leq r$, thus, using the cosine law we get that $a^2 \leq r^2(2 - \sqrt{2})$ (1). The Manhattan distance of p_1, p_2 is equal to $x + y = \sqrt{a^2 + 2xy}$ (2). Also, the following hold: $x = \sqrt{b^2 - z^2}$, $y = c - z$ and $z = b\cos(\angle p_1 p_i p_2) \geq \frac{b\sqrt{2}}{2}$. Substituting z and c in the first two equations, we get $x \leq \frac{b}{\sqrt{2}}$ and $y \leq r - \frac{b\sqrt{2}}{2}$. From (1), (2) we now get that $x + y \leq r$, which contradicts the independence of p_1 and p_2. Therefore, p can have at most $(2\pi/\frac{\pi}{4}) - 1 = 7$ independent neighbors. ∎

Lemma 4.3. *If d is the Euclidean distance and $dim = 3$, each item p_i in $\mathcal{P}$ has at most $B = 24$ neighbors that are independent from each other.*

Proof. Assume a sphere of radius r centered at p_i. To fit as many independent items in the sphere as possible, we place them on the surface of the sphere at distance r from each other. Let p_1, p_2 be two such items. Since the radius of the sphere is also r, it holds that $\angle p_1 p_i p_2 = \frac{\pi}{3}$. Thus, the arc on the surface of the sphere between p_1 and p_2 is equal to $\frac{\pi}{3}r$. The problem of how many such independent items can be placed on the surface of the sphere is equivalent to that of how many equilateral spherical triangles of side length $\frac{\pi}{3}r$ can be packed on the surface of the sphere, without overlap except at the edges. This number is not known exactly but it has been shown to be between 20 and 22 (e.g., [111]). The proof is based on dividing the area of the surface of the sphere by the the area of such a triangle. To form these triangles, 24 items are required (3 for the first triangle plus 1 for each of the rest of the triangles), which proves the lemma. ∎

4.1.3 DisC Diversity with Multiple Radii

So far, we considered that the radius r is global, i.e., r is the same for all items in $\mathcal{P}$. Radius r specifies the granularity with which the selected DisC diverse subset represents the underlying result space. A large r results in a small subset, whereas a small r results in a large subset. There may be cases, however, in which we want different parts of the data space to be represent with more or less items in the DisC diverse subset. To allow this, we consider the more general case where each item p_i is associated with a different radius $r(p_i)$, i.e., r is not a constant but, instead, a function $r : \mathcal{P} \rightarrow \mathbb{R}^+$ assigning a radius $r(p_i) \in \mathbb{R}^+$ to each item $p_i \in \mathcal{P}$.

The problem now loses its symmetry, since an item p_i may be in the neighborhood of an item p_j, while p_j is not in the neighborhood of p_i. This gives rise to two different interpretations of radius. One interpretation is that p_i can represent all items in its neighborhood (i.e., all items lying at a distance at most $r(p_i)$ around it). The other interpretation is that p_i can be represented by all items its neighborhood. We call the first problem *Covering DisC diverse subset problem* and the second one *CoveredBy DisC diverse subset problem*.

For example, in Figure 4.6(a), under the Covering DisC semantics, the radius of p_3 means that p_3 represents, or covers, p_2 and p_1, whereas, based on their radius, neither p_2 nor p_1 can represent, or cover, p_3. On the contrary, under the CoveredBy semantics, the radius of p_3 means that p_3 can be represented, or covered, by p_2 or p_1, but neither p_2 nor p_1 can be represented, or covered, by p_3.

Next, we present the corresponding formal definitions.

Definition 4.4. (COVERING DISC DIVERSE SUBSET) Let $\mathcal{P}$ be a set of items and $r : \mathcal{P} \rightarrow \mathbb{R}^+$ be a function determining the radius of each item in $\mathcal{P}$. A subset S of $\mathcal{P}$ is a Covering Dissimilar-and-Covering diverse subset, or Covering DisC diverse subset, of $\mathcal{P}$, if the following two conditions hold: (i) (coverage condition) $\forall p_i \in \mathcal{P}$, $\exists$ p_j with $d(p_i, p_j) \leq r(p_j)$, 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 $d(p_i, p_j) > \max\{r(p_i), r(p_j)\}$.

For example, in Figure 4.6, $\{p_3, p_5, p_6, p_7\}$ is a Covering DisC subset of the depicted set of items.

Definition 4.5. (COVEREDBY DISC DIVERSE SUBSET) Let $\mathcal{P}$ be a set of items and $r : \mathcal{P} \rightarrow \mathbb{R}^+$ be a function determining the radius of each item in $\mathcal{P}$. A subset S of $\mathcal{P}$ is a CoveredBy Dissimilar-and-Covering diverse subset, or CoveredBy DisC diverse subset, of $\mathcal{P}$, if the following two conditions hold: (i) (coverage condition) $\forall p_i \in \mathcal{P}$, $\exists$ p_j with $d(p_i, p_j) \leq r(p_i)$, 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 $d(p_i, p_j) > \max\{r(p_i), r(p_j)\}$.

For example, in Figure 4.6, $\{p_2, p_4, p_7\}$ is a CoveredBy DisC subset of the depicted set of items.

4.1.4 Graph Representation and NP-hardness

The various DisC subsets presented so far all have a corresponding graph representation. Consider first a single radius r and let $G_{\mathcal{P},r}$ = (V, E) be an undirected graph such that there is a vertex $v_i \in V$ for each item $p_i \in \mathcal{P}$ and an edge $(v_i, v_j) \in E$, if and only if, $d(p_i, p_j) \leq r$ for the corresponding items p_i, p_j. An example is shown in Figure 4.4(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

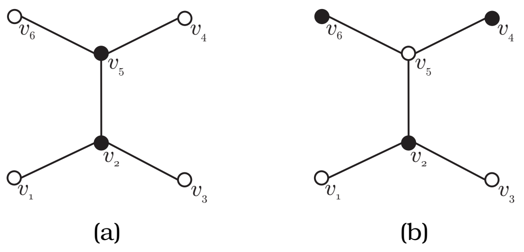


Figure 4.7: (a) Minimum dominating set ($\{v_2, v_5\}$) and (b) a minimum independent dominating set ($\{v_2, v_4, v_6\}$) of the depicted graph.

vertex in D by some edge. An *independent set* I for a graph G is a set of vertices of G such that for every two vertices in I, there is no edge connecting them. It is easy to see that a dominating set of $G_{\mathcal{P},r}$ satisfies the covering condition of Definition 4.1, whereas an independent set of $G_{\mathcal{P},r}$ satisfies the dissimilarity condition of Definition 4.1. Thus:

Lemma 4.4. *Finding an r-DisC diverse subset for a set $\mathcal{P}$ is equivalent to finding an independent dominating set of the corresponding graph $G_{\mathcal{P},r}$.*

We next present some useful properties that relate the coverage (i.e., dominance) and dissimilarity (i.e., independence) conditions. A *maximal independent set* of a graph is an independent set such that adding any other vertex to the set forces the set to contain an edge, that is, an independent set that is not a subset of any other independent set. It is known that:

Lemma 4.5. *An independent set of a graph is maximal, if and only if, it is dominating.*

From Lemma 4.5, we conclude that:

Observation 4.1. *A minimum maximal independent set is also a minimum independent dominating set.*

However,

Observation 4.2. *A minimum dominating set is not necessarily independent.*

For example, in Figure 4.7, the minimum dominating set of the depicted items is of size 2, while the minimum independent dominating set is of size 3.

The above also holds for the Minimum Weighted r-DisC diverse subset problem.

We next consider the multiple radii case. Our graph-based view of the problem is now the following. Let $G_{\mathcal{P},r(.)}$ = (V, E) be a directed graph such that there is a vertex $v_i \in V$ for each item $p_i \in \mathcal{P}$ and a (directed) edge $(v_i, v_j) \in E$, if and only if, for the corresponding items p_i, p_j, it holds that $d(p_i, p_j) \leq r(p_i)$ (Covering problem) or $d(p_i, p_j) \leq r(p_i)$ (CoveredBy problem). In Figure 4.6, we see an example. The coverage relationship is not symmetric anymore. In Figure 4.6(b), for example, item p_3 covers p_1 and p_2, but neither p_1 nor p_2 cover p_3. Independence between two items means that none of them covers the other. In Figure 4.6(b), p_3 and p_7 are independent, but points p_3 and p_1 are not.

A dominating, or covering, set D for a directed 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 incoming edge. An independent set I for a directed graph G is a set of vertices of G such that, for every two vertices in I, there is no edge connecting them.

Lemma 4.6. *Finding a Covering or CoveredBy DisC diverse subset for a set $\mathcal{P}$ is equivalent to finding an independent dominating set of the corresponding graph $G_{\mathcal{P},r(.)}$.*

Proof. Let S be a DisC diverse subset for $\mathcal{P}$. Due to the coverage condition, for every item p_i not in S, there must be an item p_j in S with $d(p_i, p_j) \leq r(p_j)$ (Covering problem) or $d(p_i, p_j) \leq r(p_i)$ (CoveredBy problem), thus S is a dominating set of $G_{\mathcal{P},r(.)}$. Also, due to the dissimilarity condition, no item p_i in S can cover some other item p_j in S. Thus, S is also an independent set. Now, let S be an independent dominating set S of the directed graph $G_{\mathcal{P},r(.)}$. Then, for every item p_i not in S there is some item p_j in S such that an edge (p_j, p_i) exists, i.e., $d(p_i, p_j) \leq r(p_j)$ (Covering problem) or $d(p_i, p_j) \leq r(p_i)$ (CoveredBy problem). Also, there is no edge connecting p_i, p_j in S, i.e., $d(p_i, p_j) > \max\{r(p_i), r(p_j)\}$. Thus, both the coverage and dissimilarity conditions of Definition 4.4 (Covering problem) or Definition 4.5 (CoveredBy problem) hold and S is also a DisC diverse subset for $\mathcal{P}$. ∎

Note that, when all items are associated with equal radii, it holds that $d(p_i, p_j) \leq r(p_i)$ if an only if $d(p_i, p_j) \leq r(p_j)$. In this case, the graph representation of a set $\mathcal{P}$ for the Covering problem is equivalent to that for the CoveredBy problem and, in addition, all edges of the graph are bidirectional, i.e., the graph can be reduced to an undirected graph.

Finding a minimum independent dominating set of a graph has been proven to be NP-hard [53], even for special cases of graphs such as unit disk graphs [34], i.e., graphs in the Euclidean space 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. Next, we provide a suite of algorithms for computing approximate solutions.

4.2 Computing DisC Diverse Subsets

Next, we present a suite of algorithms for locating DisC diverse subsets. We first present general algorithms for both the single radius and multiple radii cases and then a number of variations.

Algorithm 4.1 Locating DisC diverse subsets.

Input: A set of items $\mathcal{P}$, a radius function $r(.)$ and a selection criterion $\mathcal{C}(.)$.

Output: A DisC diverse subset S of $\mathcal{P}$.

```
1: $S \leftarrow \emptyset$
2: for all $p_i \in \mathcal{P}$ do
3:     color $p_i$ white
4: while there exist white items do
5:     select the white item $p_i$ with the largest value of $\mathcal{C}(p_i)$
6:     $S = S \cup \{p_i\}$
7:     color $p_i$ black
8:     for all $p_j \in N^W_{r(p_i)}(p_i)$ (Covering) or $p_j$ s.t. $p_i \in N_{r(p_j)}(p_j)$ (CoveredBy) do
9:         color $p_j$ grey
10: return $S$
```

4.2.1 General Algorithms

Our various algorithms are based on Algorithm 4.1. For presentation convenience, let us call *black* the items of $\mathcal{P}$ that are in the diverse subset S, *grey* the items covered by some item in S and *white* the items that are neither black nor grey. $N^W_r(p_i)$ denotes the set of white neighbors of p_i. Initially, S is empty and all items are white. Items are selected for inclusion in S in rounds based on some selection criterion C.

Lemma 4.7. *In the single radius case, Algorithm 4.1 produces an r-DisC diverse subset S of $\mathcal{P}$ for any selection criterion.*

Proof. At first all items are white. Once an item enters S, all its neighbors become grey and are withdrawn from consideration. Any white item is independent from all selected items in S and thus can be selected to be included in S. To see that, assume for the purpose of contradiction, that for a white item p_j and black item p_i, it holds $d(p_i, p_j) \leq r$, then $p_j \in N_r(p_i)$, thus it should have been colored grey, when p_i was selected for inclusion in S. Thus, the set produced by selecting any white item is an independent set. It is also a maximal independent set, since at the end there are only grey items left, thus adding any of them to S would violate the independence of S. From Lemma 4.5, S is an r-DisC diverse subset. ∎

While an undirected graph always has an independent dominating subset, this is not the case for directed graphs (e.g., [83]). To illustrate this, consider the following simple example of Figure 4.9 where $V = \{v_1, v_2, v_3\}$ and $E = \{(v_1, v_2), (v_2, v_3), (v_3, v_1)\}$. In this graph, no single item is able to cover the whole set, while, at the same time, no two items are independent from each other. Thus, this graph has no independent dominating subset. However, such a graph cannot exist in our case. Assume, for the purposes of contradiction, that such as a graph exists. For the Covering case, since there is an edge (v_1, v_2) but there is no edge (v_2, v_1), we get that $r(p_1) > d(p_1, p_2) > r(p_2)$. Similarly, it also holds that $r(p_2) > r(p_3)$ and $r(p_3) > r(p_1)$. Therefore, we get that

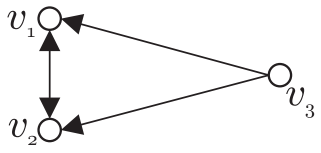


Figure 4.8: A directed graph example.

$r(p_1) > r(p_1)$ which cannot be true. Similarly, for the CoveredBy problem, we get that $r(p_1) < r(p_1)$ which, also, cannot be true.

However, in our case, where d is a distance metric, we can always construct an independent dominating subset for our directed graphs. To achieve this, we need to select white items using some *specific* criterion as the following lemma shows. Note, that the proof of Lemma 4.7 does not hold, since not all white items are necessarily independent from the selected diverse items. For example, consider the Covering problem for the graph of Figure 4.8 and assume that v_1 is selected first. Then, v_1 will be colored black and v_2 will be colored grey, while v_3 will remain white. However, v_3 cannot enter the selected subset in a following step, since it is not independent from v_1. Thus, Algorithm 4.1 does not produce an r-DisC diverse subset for *any* selection criterion.

Lemma 4.8. *Algorithm 4.1 produces a (multiple radii) DisC diverse subset S of $\mathcal{P}$ when selecting white items in (i) decreasing order of their radius for the Covering problem and (ii) increasing order of their radius for the CoveredBy problem.*

Proof. We prove the lemma for the Covering problem. The proof for the CoveredBy problem is similar. At first all items are white. Upon selecting an item for inclusion, all its neighbors become grey and are thus withdrawn from consideration. Let p_i be a white item considered at some round and p_j be an already selected, i.e., black, item. Since p_i is still white, there can be no directed edge (p_j, p_i). Since p_j was considered for inclusion in S prior to p_i, it holds that $r(p_i) \leq r(p_j)$. Moreover, since p_i is still white, it holds that $r(p_i) < r(p_j)$ and there can be no directed edge (p_i, p_j). Therefore, each white item selected to be colored black at some round is independent from all previously selected items, i.e., the produced set is an independent one. It is also a maximal independent set, since at the end there are only grey items left (line 5), thus selecting any of them would violate the independence of S. From Lemma 4.5, S is a DisC diverse subset. ∎

Now, Algorithm 4.1 for the example of Figure 4.8, would select v_3 first, since v_3 has the largest radius (it covers both v_1 and v_2) and the Covering DisC diverse subset $\{p_3\}$ will be produced.

Furthermore, in our example of Figure 4.6, by visiting white items in decreasing order of their radius (solving the Covering problem), Algorithm 4.1 would first select p_3, followed by p_5, p_6 and p_7, in that order, resulting in a DisC diverse subset of $\mathcal{P}$. Visiting white items in increasing order of their radius (solving the CoveredBy problem) would result in the selection of p_2, p_7 and p_4, in that order.

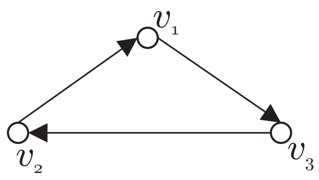


Figure 4.9: A directed graph with no independent dominating set.

4.2.2 Greedy Algorithms

As shown, we can compute a DisC subset by selecting any of the white items in the single radii case, or by selecting any among the white items having the largest (smallest) radius in the Covering (resp. CoveredBy) multiple radii case. We call these algorithms `Basic-DisC` algorithms. As shown by Lemma 4.1, the size of any DisC subset, and thus the size of the DisC subset produced by `Basic-DisC` is at most B times larger than that of a minimum r-DisC diverse subset or a minimum weighted r-DisC diverse subset.

We now consider the following intuitive greedy variation of `Basic-DisC`, that we call `Greedy-DisC`. Instead of selecting white items arbitrarily at each step for inclusion in the diverse subset S, we select the white item that minimizes the objective function f. That is, in the case of the Minimum r-DisC diverse subset problem, we set $C(p_i) = |N_r^W(p_i)|$, i.e., we select the white item that covers the largest number of uncovered items. In the case of the Minimum Weighted r-DisC diverse subset problem, we select the white item that has the best combination of weight and white neighborhood size. We normalize the size of a white neighborhood in $[0, 1]$ (recall that we have assumed that weights are in $(0, 1]$) and use $C(p_i) = w(p_i) \left({|N_r^W(p_i)|}/{\max_{p_j \in \mathcal{P} \setminus S} |N_r^W(p_j)|} \right)$. In case of ties, we select the white item with the largest number of white neighbors.

4.2.3 Greedy Algorithms for Coverage Only

While the size of the subsets produced by `Greedy-DisC` is expected to be smaller than that of the subsets produced by `Basic-DisC`, the fact that we consider for inclusion in S only white, i.e., independent, items may still not reduce the size of S as much as expected. From Observation 4.2, it is possible that an independent covering set is larger than a covering set that also includes dependent items. For example, consider the vertices (or equivalently the corresponding items) in Figure 4.7. Assume that v_2 is inserted in S first, resulting in v_1, v_3 and v_5 becoming grey. Then, we need two more vertices, namely, v_4 and v_6, for covering the whole set. However, if we consider for inclusion grey items as well, then v_5 can join S, resulting in a smaller covering set.

Motivated by this observation, we also define r-C diverse subsets that satisfy only the coverage condition of Definition 4.1 and modify `Greedy-DisC` accordingly to compute r-C diverse sets. The only change required is that in line 6 of Algorithm 4.1, we select both white and grey items. This allows us to select at each step the item that covers the largest possible number of uncovered items, even if this item is grey. We call this variation `Greedy-C`.

For `Greedy-C`, we get a covering but not necessarily dissimilar subset of $\mathcal{P}$, whose size is generally different than the size of the subset produced by `Greedy-DisC` for the same radius r. In this case, we get a different bound for the size of the produced r-C diverse subset S.

Theorem 4.2. *Let Δ be the maximum number of neighbors of any item in $\mathcal{P}$. The weighted r-C diverse subset produced by* `Greedy-C` *is at most $\ln \Delta$ times larger than the minimum weighted r-DisC diverse subset S^*.*

Proof. We consider that inserting a vertex (item) p into S has cost ${}^{1}/_{w(p)}$. We distribute this cost equally among all covered vertices, i.e., after being labeled grey, vertices are not charged anymore. Assume an optimal minimum dominating set S^*. The graph G can be decomposed into a number of star-shaped subgraphs, each of which has one vertex from S^* at its center. The cost of an optimal minimum dominating set is exactly ${}^{1}/_{w(p)}$ for each star-shaped subgraph centered around p. We show that for a non-optimal set S, the cost for each star-shaped subgraph is at most $\ln \Delta$, where Δ is the maximum degree of the graph. Consider a star-shaped subgraph of S^* with p at its center and let $N_r^W(p)$ be the number of white vertices in it. If a vertex in the star is labeled grey by `Greedy-C`, these vertices are charged some cost. By the greedy condition of the algorithm, this cost can be at most $1/w(p)|N_r^W(p)|$ per newly covered vertex. Otherwise, the algorithm would rather have chosen p for the dominating set because p would contribute $1/w(p)|N_r^W(p)|$ to the selected set. In the worst case, no two vertices in the star of p are covered at the same iteration. In this case, the first vertex that is labeled grey is charged at most $1/w(p)(\delta(p)+1)$, the second vertex is charged at most $1/w(p)\delta(p)$ and so on, where $\delta(p)$ is the degree of p. Therefore, the total cost for covering the star of p is at most:

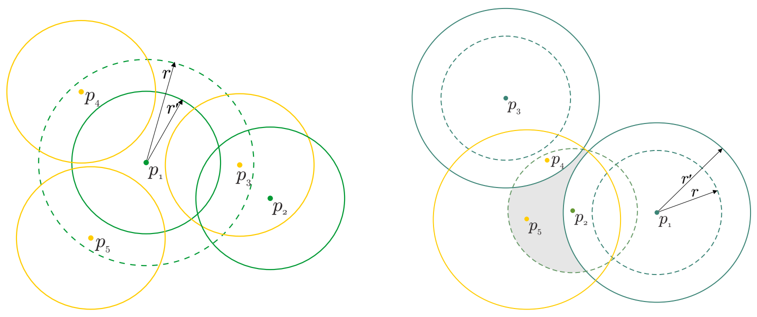
$$\frac{1}{w(p)}\left(\frac{1}{\delta(p)+1}+\frac{1}{\delta(p)}+\ldots+\frac{1}{2}+1\right)=\frac{1}{w(p)}H(\delta(p)+1)\leq\frac{1}{w(p)}H(\Delta+1)$$

where $H(i)$ is the i^{th} harmonic number. The total cost of the set S produced by `Greedy-C` for covering all the stars of S^* is:

$$f(S)\leq\sum_{p_i\in S^*}\frac{1}{w(p_i)}H(\Delta+1)=H(\Delta+1)f(S^*)\approx\ln\Delta f(S^*)$$

Since the size of a minimum weighted dominating set is equal or smaller than the size of a minimum weighted independent dominating set, the theorem holds. ∎

Note that, Theorem 4.2 also holds in the case of the (unweighted) Minimum r-DisC diverse subset problem, where we simply consider the cost of each star-shaped subgraph around p to be equal to 1.



(a) Zooming-in. (b) Zooming-out.

Figure 4.10: Zooming (a) in and (b) out. Dashed and solid circles correspond to radius r and r' respectively.

4.3 Incremental DisC

The radius r determines the desired degree of diversification. A large radius corresponds to fewer and less similar to each other representative items, whereas a small radius results in more and less dissimilar representative items. At one extreme, a radius equal to the largest distance between any two items results in a single item being selected and at the other extreme, a radius smaller than the smallest pairwise distance in the result set results in all items of $\mathcal{P}$ being selected.

In this section, we consider an interactive mode of operation where, after being presented with an initial set of results for some radius r, a user asks to see either more or less results by correspondingly decreasing or increasing the radius. For simplicity, we shall focus on zooming in the case of a single radius. The results are easily transferred to the case of multiple radii. First, we present results relating the size of DisC diverse subsets for different radii and then propose algorithms for incrementally changing the radius.

4.3.1 Zooming

Given a set of items $\mathcal{P}$ and an r-DisC diverse subset S of $\mathcal{P}$ for some specific radius, we want to compute an r'-DisC diverse subset S' of $\mathcal{P}$. There are two cases: (i) $r' < r$ and (ii) $r' > r$ which we call *zooming-in* and *zooming-out* respectively.

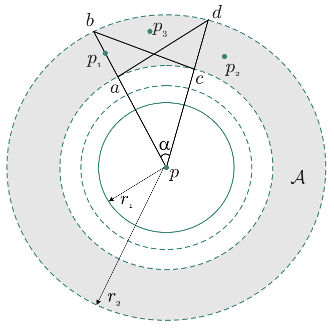
Since we want to support an incremental mode of operation, the set S' should be as close as possible to the already seen result S. Ideally, $S' \supseteq S$, for $r' < r$ and $S' \subseteq S$, for $r' > r$. However, this is not always possible as the following lemma shows.

Lemma 4.9. *Let S be a covering and dissimilar subset of $\mathcal{P}$ for r.*

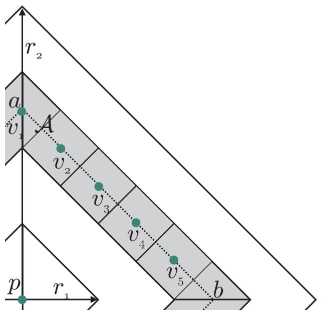
(i) S is a covering but not necessarily dissimilar subset of $\mathcal{P}$ for $r' > r$,

(ii) S is a dissimilar but not necessarily covering subset of $\mathcal{P}$ for $r' < r$.

Proof. (i) Let $r' > r$. Since S is a covering subset of $\mathcal{P}$ for r, for each item $p_i \in \mathcal{P}$, there is an item $p_j \in S$ such that $d(p_i, p_j) \leq r < r'$, thus S is also a covering subset of $\mathcal{P}$



(a) Euclidean plane.



(b) Manhattan plane.

Figure 4.11: Independent neighbors.

for r'. However, for two items p_i, $p_j \in S$, it is possible that $p_j \in N_{r'}(p_i)$. Therefore, S may not be a dissimilar subset of $\mathcal{P}$ for r'. (ii) Let $r' < r$. Since S is a dissimilar subset of $\mathcal{P}$ for r, for any two items p_i, $p_j \in \mathcal{P}$, it holds $d(p_i, p_j) > r > r'$, thus S is also a dissimilar subset of $\mathcal{P}$ for r'. However, it is possible that there exists some item $p_i \in \mathcal{P}$ for which there does not exist an item $p_j \in S$ with $p_i \in N_{r'}(p_j)$. Therefore, S may not be a covering subset of $\mathcal{P}$ for r'. ∎

To study the relationship between S and S' when changing the radius, we focus on the items lying at distance between r and r' from the selected items of the initial DisC diverse subset (Figure 4.10). These are items of interest since they are possibly either left uncovered when the radius decreases (zooming-in), thus violating the covering property, or covered by other diverse items when the radius increases (zooming-out), thus violating the dissimilarity property.

For two radii r_1, r_2, $r_2 \geq r_1$, we define the set $N^I_{r_1,r_2}(p_i)$, as the set of items at distance at most r_2 from p_i which are at distance at least r_1 from each other, i.e., items in $N_{r_2}(p_i) \backslash N_{r_1}(p_i)$ that are independent from each other considering the radius r_1. The following lemma bounds the size of $N^I_{r_1,r_2}(p_i)$ for specific distance metrics and dimensionality.

Lemma 4.10. *Let* r_1, r_2 *be two radii with* $r_2 \geq r_1$. *Then, for* $dim = 2$*:*

(i) if d *is the Euclidean distance:*

$$\left|N^I_{r_1,r_2}(p_i)\right| \leq 9 \left\lceil \log_\beta (r_2/r_1) \right\rceil, \text{ where } \beta = \frac{1+\sqrt{5}}{2}$$

(ii) if d *is the Manhattan distance:*

$$\left|N^I_{r_1,r_2}(p_i)\right| \leq 4 \sum_{i=1}^{\gamma} (2i+1), \text{ where } \gamma = \left\lceil \frac{r_2 - r_1}{r_1} \right\rceil$$

Proof. Euclidean distance: For the proof, we use a technique for partitioning the annulus between r_1 and r_2 similar to the one in [102] and [112]. Let r_1 be the radius of an item p (Figure 4.11(a)) and α a real number with $0 < \alpha < \frac{\pi}{3}$. We draw circles around the item p with radii $(2cos\alpha)^{x_p}$, $(2cos\alpha)^{x_p+1}$, $(2cos\alpha)^{x_p+2}$, $\ldots$, $(2cos\alpha)^{y_p-1}$, $(2cos\alpha)^{y_p}$, such that $(2cos\alpha)^{x_p} \leq r_1$ and $(2cos\alpha)^{x_p+1} > r_2$ and $(2cos\alpha)^{y_p-1} < r_2$ and $(2cos\alpha)^{y_p} \geq r_2$. It holds that $x_p = \left\lfloor \frac{\ln r_1}{\ln(2\cos\alpha)} \right\rfloor$ and $y_p = \left\lceil \frac{\ln r_2}{\ln(2\cos\alpha)} \right\rceil$. In this way, the area around p is partitioned into $y_p - x_p$ annuli plus the r_1-disk around p. Consider an annulus $\mathcal{A}$. Let p_1 and p_2 be two neighbors of p in $\mathcal{A}$ with $d(p_1, p_2) > r_1$. Then, it must hold that $\angle p_1pp_2 > \alpha$. To see this, we draw two segments from p crossing the inner and outer circles of $\mathcal{A}$ at a, b and 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 $\frac{|pb|}{|pc|} = \frac{|pd|}{|pa|} = 2\cos\alpha$. From the cosine law for $\overset{\triangle}{pad}$, we get that $|ad| = |pa|$ and, therefore, it holds that $|cb| = |ad| = |pa| = |pc|$. Therefore, for any item p_3 in the area $abcd$ of $\mathcal{A}$, it holds that $|pp_3| > |bp_3|$ which means that all items 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 $\angle p_1pp_2 > \alpha$. Based on this, we see that there exist at most $\frac{2\pi}{\alpha} - 1$ independent (for r_1) nodes in $\mathcal{A}$. The same holds for all annuli. Therefore, we have at most $(y_p - x_p)\left(\frac{2\pi}{\alpha} - 1\right)$ independent nodes in the annuli. For $0 < \alpha < \frac{\pi}{3}$, this has a minimum when α is close to $\frac{\pi}{5}$ and that minimum value is $9\left\lceil \frac{\ln(r_2/r_1)}{\ln(2\cos(\pi/5))} \right\rceil = 9\left\lceil \log_\beta(r_2/r_1) \right\rceil$, where $\beta = \frac{1+\sqrt{5}}{2}$.

Manhattan distance: Let r_1 be the radius of an item p. We draw Manhattan circles around the item 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 $\gamma = \left\lceil \frac{r_2 - r_1}{r_1} \right\rceil$ Manhattan annuluses plus the r_1-Manhattan-disk around p. Consider an annulus $\mathcal{A}$. The items shown in Figure 4.11(b) 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, $|ab| = \sqrt{2\left(ir_1 + r_1/2\right)^2}$. Also, for two items p_1, p_2 it holds that $|p_1p_2| = \sqrt{2\left(r_1/2\right)^2}$. Therefore, at one quadrant of the annulus there are $\frac{|ab|}{|p_1p_2|} = 2i + 1$ independent neighbors which means that there are $4(2i+1)$ independent neighbors in $\mathcal{A}$. Therefore, there are in total $\sum_{i=1}^{\gamma} 4(2i+1)$ independent (for r_1) neighbors of p. ∎

4.3.2 Incremental Zooming Algorithms

Next, we describe our algorithms for incrementaly adapting an r-DisC diverse subset S to an r'-DisC diverse subset S' and provide bounds concerning the size relationship of S and S'.

Zooming-in

Let us first consider the case of zooming-in to 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 construct r'-DisC diverse sets that are supersets of S by adding items to S to make it maximal.

Consider an item of S, for example p_1 in Figure 4.10(a). Items at distance at most

Algorithm 4.2 Greedy-Zoom-In.

Input: A set of items $\mathcal{P}$, a solution S and initial and new radii $r(p_i)$, $r'(p_i)$, $r'(p_i) < r(p_i)$, for each item p_i in $\mathcal{P}$.

Output: An adapted DisC diverse subset of $\mathcal{P}$.

```
 1: $S' \leftarrow S$
 2: **for all** $p_i \in S$ **do**
 3:    color items in $\{N_{r(p_i)}(p_i) \backslash N_{r'(p_i)}(p_i)\}$ white
 4: **while** there exist white items **do**
 5:    select the white item $p_i$ with the largest $\left|N^W_{r(p_i)}(p_i)\right|$
 6:    color $p_i$ black
 7:    $S' = S' \cup \{p_i\}$
 8:    **for all** $p_j \in N^W_{r(p_i)}(p_i)$ **do**
 9:       color $p_j$ grey
10: **return** $S'$
```

r' from p_1 are still covered by p_1 and cannot enter S'. Items at distance greater than r' and at most r may be uncovered and join S'. Each of these items can enter S' as long as it is not covered by some other item of S that lays outside the former neighborhood of p_1. For example, in Figure 4.10(a), p_4 and p_5 may enter S' while p_3 can not, since, even with the smaller radius r', p_3 is covered by p_2.

To adapt a DisC diverse subset, we consider such items in turn. This turn can be either arbitrary (`Basic-Zoom-In` algorithm) or proceed in a greedy way, where at each turn the item that covers the largest number of uncovered items is selected (`Greedy-Zoom-In`, Algorithm 4.2).

Concerning the size relationship between S and S', the following lemma holds.

Lemma 4.11. *Let S be the initial DisC diverse subset and S' be the adapted one generated by the `Basic-Zoom-In` or `Greedy-Zoom-In` algorithm. It holds that:*

(i) $S \subseteq S'$ and

(ii) $|S'| \leq |S| + \sum_{p_i \in S} |N^I_{r',r}(p_i)|$

Proof. Condition (i) trivially holds from step 1 of the algorithm. Condition (ii) holds since for each item in S there are at most $|N^I_{r',r}(p_i)|$ independent items at distance greater than $r'(p_i)$ from each other that can enter S'. ∎

In practice, items selected to enter S', such as p_4 and p_5 in Figure 4.10(a), are likely to cover other items left uncovered by the same or similar items in S. Therefore, the size difference between S and S' is expected to be smaller than this theoretical upper bound.

Zooming-out

Next, we consider zooming-out to a larger radius, i.e., $r' > r$. In this case, the user is interested in seeing less and more dissimilar items, ideally a subset of the already seen results for r, that is, $S' \subseteq S$. However, in this case, in contrast to zooming-in, it may not be possible to construct a diverse subset S' that is a subset of S (take, for example, the items of Figure 4.10(b) with $S = \{p_1, p_2, p_3\}$; no subset of S is an r'-DisC diverse subset for this set of items).

We focus on the following sets of items: (i) $S \backslash S'$ and (ii) $S' \backslash S$. The first set consists of the items that belong to the previous diverse subset but are removed from the new one, while the second set consists of the new items added to S'. To illustrate, let us consider the items of Figure 4.10(b) and that p_1, p_2, $p_3 \in S$. Since the radius becomes larger, p_1 now covers all items at distance at most r' from it. This may include a number of items that also belonged to S, such as p_2. These items have to be removed from the solution, since they are no longer dissimilar to p_1. However, removing such an item, say p_2 in our example, can potentially leave uncovered a number of items that were previously covered by p_2 (these items lie in the shaded area of Figure 4.10(b)). In our example, requiring p_1 to remain in S' means than p_5 should be now added to S'.

To produce the new adapted DisC diverse subset, we proceed in two passes. In the first pass, we examine all items of S in some order and remove their diverse neighbors that are now covered by them. At the second pass, items 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 4.3 shows the greedy variation; the first pass (lines 4-11) considers $S \backslash S'$, while the second pass (lines 12-19) considers $S' \backslash S$. Initially, we color all previously black items red. All other items are colored white. We consider three variations for the first pass of the greedy algorithm: selecting the red items 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 items to be added in the second pass, that is, $S' \backslash S$, while variation (b) aims at maximizing $S \cap S'$. Algorithm 4.3 depicts variation (a), where $N^R_{r'}(p_i)$ denotes the set of red neighbors of item p_i.

Concerning the size relationship between S and S', the following lemma holds.

Lemma 4.12. *For the solution S' generated by the `Basic-Zoom-Out` or `Greedy-Zoom-Out` algorithm, it holds that:*

(i) There are at most $\sum_{p_i \in S} |N^I_{r,r'}(p_i)|$ items in $S \backslash S'$.

(ii) For each item of S not included in S', at most $B - 1$ items are added to S'.

Proof. Condition (i) is a direct consequence of the definition of $N^I_{r,r'}(p_i)$. Concerning condition (ii), recall that each removed item p_i has at most B independent neighbors for $r'(p_i)$. Since p_i is covered by some neighbor, there are at most $B - 1$ other independent items that can potentially enter S'. ∎

Algorithm 4.3 Greedy-Zoom-Out(a).

Input: A set of items $\mathcal{P}$, a solution S and initial and new radii $r(p_i)$, $r'(p_i)$, $r'(p_i) > r(p_i)$, for each item p_i in $\mathcal{P}$.

Output: An adapted DisC diverse subset of $\mathcal{P}$.

```
1: $S' \leftarrow \emptyset$
2: color all black items red
3: color all grey items white
4: while there exist red items do
5:     select the red item $p_i$ with the largest $|N^R_{r'(p_i)}(p_i)|$
6:     color $p_i$ black
7:     $S' = S' \cup \{p_i\}$
8:     for all $p_j \in N_{r'(p_i)}(p_i)$ do
9:         color $p_j$ grey
10: while there exist white items do
11:     select the white item $p_i$ with the larger $|N^W_{r'(p_i)}(p_i)|$
12:     color $p_i$ black
13:     $S' = S' \cup \{p_i\}$
14:     for all $p_j \in N^W_{r'(p_i)}(p_i)$ do
15:         color $p_j$ grey
16: return $S'$
```

4.4 Comparison of Diversification Models

In this section, we first show how our DisC model relates to other widely used diversification methods and then compare the various variations of the DisC model.

4.4.1 Comparison with Other Models

Next, we present both theoretical and qualitative results concerning the relation between DisC diversity and various other diversification approaches.

Theoretical Results

Two widely used diversification models are MaxMin and MaxSum that aim at selecting a subset S of $\mathcal{P}$ so as the minimum or the average pairwise distance of the selected items is maximized (e.g., [57, 109, 19]). More formally, an optimal MaxMin (resp., MaxSum) subset of $\mathcal{P}$ is a subset with the maximum $f_{\text{Min}}(S) = \min_{\substack{p_i,p_j \in S \\ p_i \neq p_j}} dist(p_i, p_j)$ (resp., $f_{\text{Sum}}(S) = \sum_{\substack{p_i,p_j \in S \\ p_i \neq p_j}} dist(p_i, p_j)$) over all subsets of the same size. Input in both approaches is the size k of the diverse subset.

The following lemma provides a bound for the f_{Min} distance of the items in any r-DisC set with regards to the optimal distance f_{Min} of a subset of the same size.

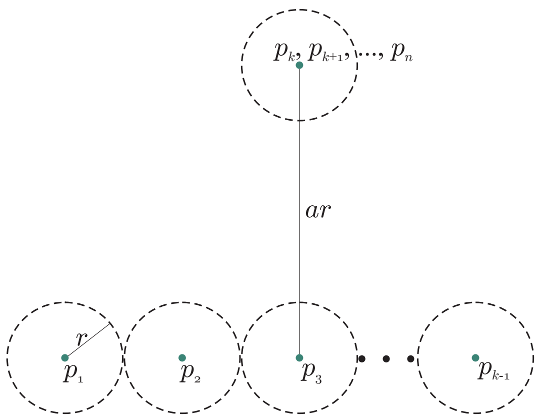


Figure 4.12: Counter example for the MaxSum case ($a \gg 1$).

Lemma 4.13. *Let $\mathcal{P}$ be a set of items, S be an r-DisC diverse subset of $\mathcal{P}$ and λ be the f_{Min} distance between the items of S. Let S^* be an optimal* MaxMin *subset of $\mathcal{P}$ for $k = |S|$ and λ^* be the f_{Min} distance of S^*. Then, $\lambda^* \leq 3\ \lambda$.*

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

Lemma 4.13 asks how much smaller the f_{Min} distance of an r-DisC subset is with regards to the optimal f_{Min} of a subset of the same size. The following lemma looks into the size of an r-DisC subset that attains the same f_{Min} distance as an optimal MaxMin subset of size k.

Lemma 4.14. *Let $\mathcal{P}$ be a set of items, S^* be an optimal subset of $\mathcal{P}$ of size k and λ^* be the f_{Min} distance of S^*. Let S be an r-DisC diverse subset with $r = \lambda^*$. It holds that $|S| < k'$, where k' is the first integer larger than k for which the corresponding optimal* MaxMin *subset of $\mathcal{P}$ $S^{*\prime}$ has f_{Min} distance equal to $\lambda^{*\prime}$, with $\lambda^{*\prime} < \lambda^*$.*

Proof. Since the optimal (i.e., maximum) minimum distance for k' is smaller than λ^*, then there can be no DisC set for $r = \lambda^*$ with size equal or larger than k'. Therefore, a DisC diverse subset for $r = \lambda^*$ can be of size up to k'. ∎

A bound similar to that of Lemma 4.13 does not exist for the MaxSum case[1]. To illustrate this, consider the example of Figure 4.12, where $k-1$ of the items are located

[1]We would like to thank Dr. Anirban Dasgupta from Yahoo! Labs for suggesting this example.

in a line, at distance r from each other, while the rest of the items are very close to each other and located at a large distance from the other items, i.e., $a \gg 1$. Let us call the two groups of items "group X" and "group Y" respectively. For simplicity, let $r = 1$.

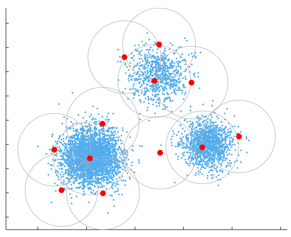
An optimal DisC diverse set for $r = 1$ is of size k and consists of the $k - 1$ items of group X plus one item from group Y. Let G be the sum of the pairwise distances of all items of group X. The sum of pairwise distances f_{DisC} of the optimal DisC set is approximately equal to $a(k - 1) + G$. A MaxSum solution for k would instead select $k/2$ items from group X and $k/2$ items from group Y. The corresponding sum of pairwise distances f_{MaxSum} in this case would be $a(k/2)^2 + G'$, where $G' < G$ is the sum of the pairwise distances of all items from group X. Since a can be arbitrarily large, for a sufficiently large value of a, we can assume that $f_{DisC} \simeq a(k - 1)$ and $f_{\text{MaxSum}} \simeq a(k/2)^2$. Therefore, for a sufficiently large value of a, it holds that $\frac{f_{\text{MaxSum}}}{f_{DisC}} = \frac{k^2}{4(k-1)}$. Thus, $\frac{f_{\text{MaxSum}}}{f_{DisC}}$ can grow arbitrarily large as k increases.

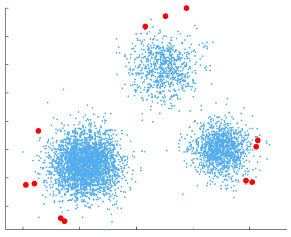
Qualitative Results

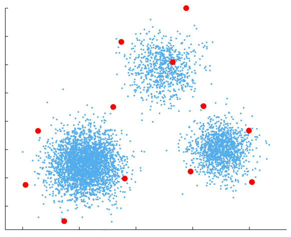
Next, we present some qualitative results of applying different approaches for selecting diverse items, namely DisC, MaxMin and MaxSum. We also show results for k-medoids, a widespread clustering algorithm that seeks to minimize $\frac{1}{|\mathcal{P}|} \sum_{p_i \in \mathcal{P}} d(p_i, c(p_i))$, where $c(p_i)$ is the closest item of p_i in the selected subset, since the located medoids can be viewed as a representative subset of the dataset. We used a 2-dimensional "Clustered" dataset. To implement MaxMin and MaxSum, we used greedy heuristics which have been shown to achieve good solutions [42]. To allow for a comparison, we first run `Greedy-DisC` for a given r and then use the size of the produced diverse subset as the input k of the other approaches. In this example, k = 12 for r = 0.15 (Figure 4.13).

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 items, ignoring items that are further away. MaxMin performs better in this aspect. However, since MaxMin seeks to retrieve items that are as far apart as possible, it fails to retrieve items from dense areas; see, for example, the central areas of the clusters in Figure 4.13. 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 [109] but the results did not differ substantially from the ones in Figure 4.13(b).

In Figure 4.14, we see how the DisC solution (Figure 4.14(a)) is affected when the dissimilarity condition is raised, i.e., we have a covering but not necessarily independent subset of the data (Figure 4.14(b)). Raising the dissimilarity condition slightly decreases the size (by one item) in this example. However, the selected items are close together (see, for example, the cluster on the right of the dataset).


(a) r-DisC.


(b) MaxSum.


(c) MaxMin.

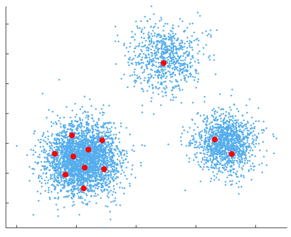
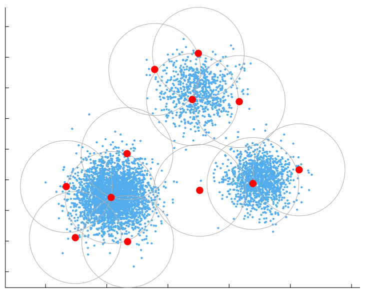

(d) k-medoids.

Figure 4.13: Solutions by the various diversification methods for a clustered dataset. All items are associated with equal weights and radii. Selected items are shown as solid circles. Circles around items of the DisC solution denote the radius r of the selected items.


(a) r-DisC.

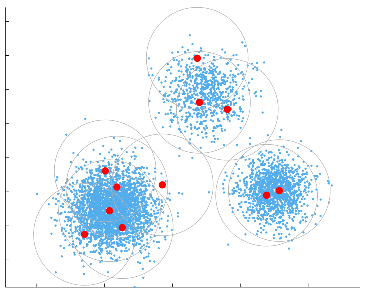

(b) r-C.

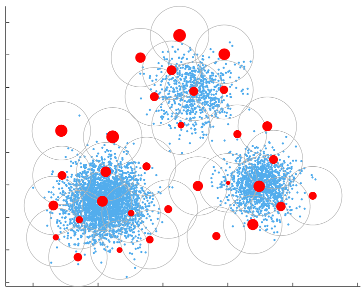
Figure 4.14: Solutions for (a) the Dissimilar-and-Covering (r-DisC) and (b) Covering-only (r-C) problems. Selected items are shown as solid circles. Circles around items denote the radius r of the selected items.

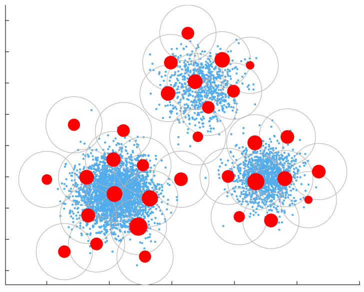
4.4.2 Comparison of DisC Models

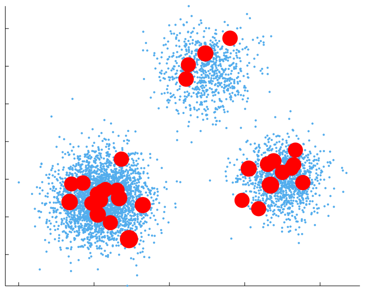
We next present a qualitative comparison of how the assignment of different weights and/or different radii to each item affect the retrieved DisC diverse subsets. To do this, we use again our "Clustered" dataset. Unless otherwise noted, we use the `Greedy-DisC` algorithm.

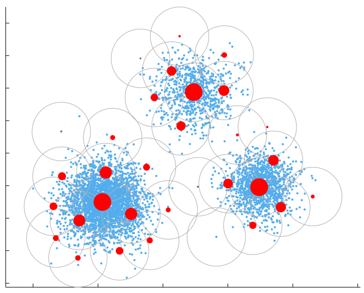
Using weights. We first compare the unweighted and weighted problems for a single radius r. We use two different ways to assign weights: (i) assigning a uniformly distributed weight to each item (uniform case) and (ii) assigning larger weights to items closer to the center of their cluster (clustered case). The second approach models the common case in which we have a number of different interpretations of the query and items close to one of these interpretations are more important. All weights are in $(0, 1]$.

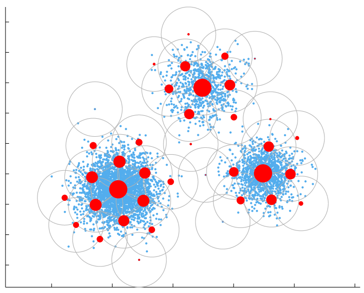
We used the same radius (r = 0.09) to generate unweighted and weighted r-DisC diverse subsets for both the uniform and the clustered cases. Figure 4.15(a) and Figure 4.15(d) depict the unweighted r-DisC diverse subsets, while Figure 4.15(b) and Figure 4.15(e) depict the weighted r-DisC diverse subsets. For comparison, we used


(a) Uniform weights: Unweighted r-DisC.


(b) Uniform weights: Weighted r-DisC.


(c) Uniform weights: Top items.


(d) Clustered weights: Unweighted r-DisC.


(e) Clustered weights: Weighted r-DisC.

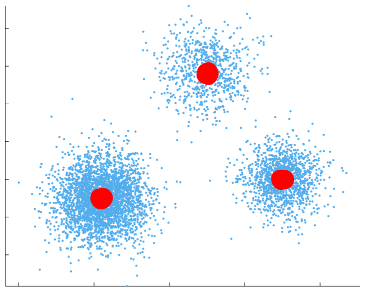
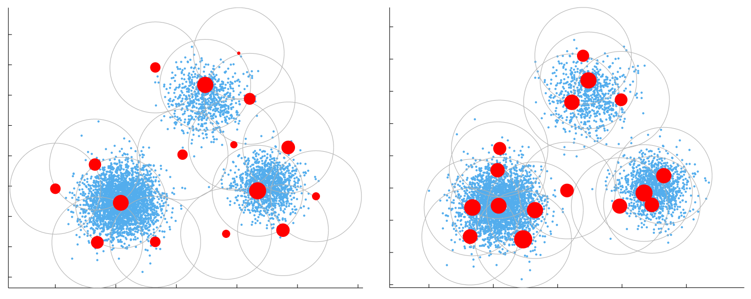

(f) Clustered weights: Top items.

Figure 4.15: Unweighted r-DisC diverse subsets (left column), weighted r-DisC diverse subsets (middle column) and top largest-weighted items (right column) for roughly the same number of items ($r = 0.09$ and $k \approx 30$). Selected items are shown as solid circles. Larger circles correspond to items with larger weights. Circles around items denote the radius r of the selected items.

the size k of the weighted r-DisC diverse subsets as the input for retrieving the top-k items with the largest weights (without enforcing diversity), i.e., 33 and 35 items for the uniform and clustered case respectively. Figure 4.15(c) and Figure 4.15(f) show the corresponding results. Clearly, the top-weighted items in the clustered case are very close to each other. In the uniform case they are more spread but, still, not highly diverse. In both cases, the weighted subsets are roughly of the same size as the unweighted ones. However, the selected items have clearly larger weights as expected and are diverse.

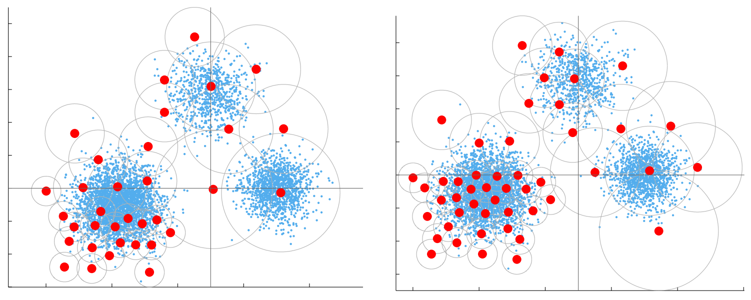
In Figure 4.16, we see how the weighted DisC solution (Figure 4.16(a)) is affected when the dissimilarity condition is raised (Figure 4.16(b)). While, in our example, the size remains the same, the selected items are closed together, i.e., not as dissimilar. However, this allows us to get more items with larger weights, since the selected items are not required to be dissimilar to each other.

Using multiple radii. Next, we present a qualitative view of various options of assigning radii to items. Our motivation behind multiple radii is to place different importance to different items in the dataset. We present three different scenaria for assigning radii to



(a) r-DisC. (b) r-C.

Figure 4.16: Weighted solutions for the Dissimilar-and-Covering and Covering-only problems. Selected items are shown as solid circles. Larger circles correspond to items with larger weights. Circles around items denote the radius r of the selected items.



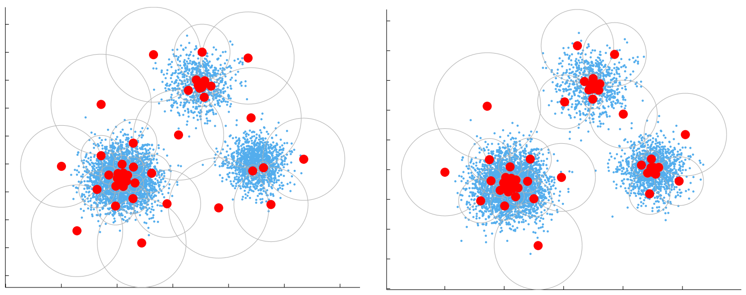
(a) Covering. (b) CoveredBy.

Figure 4.17: Using multiple radii based on areas of interest. Selected items are shown as solid circles. More items from areas of higher interest enter the diverse set. (In the CoveredBy case, each item p_i not in the diverse subset is represented by an item in the diverse subset within distance $r(p_i)$ from it. Since there is a large number of such items, we do not draw their radius for clarity.)

the items.

The first one corresponds to the case where some parts of the dataset are considered more important than others and we want them to be represented with more items in the selected diverse subset. In Figure 4.17, we see such an example, where each of the four quadrants is assumed to have different importance, with the most important one being the bottom left quadrant and importance decreasing as we move clockwise. To achieve a representation corresponding to importance, we assign to each area clockwise increasing radius values. As seen, areas associated with smaller radii (i.e., more important ones) are represented by more items in the diverse set, since items in these areas have to be closer together to be considered similar. The basic difference between the results of the CoveredBy and the Covering approach is near the boundaries of the quadrants. In the Covering approach, items in the quadrant with the larger radii cover the items in the neighboring quadrant, thus excluding them from the diverse set.

The second scenario corresponds to the case in which we want to relate representa-

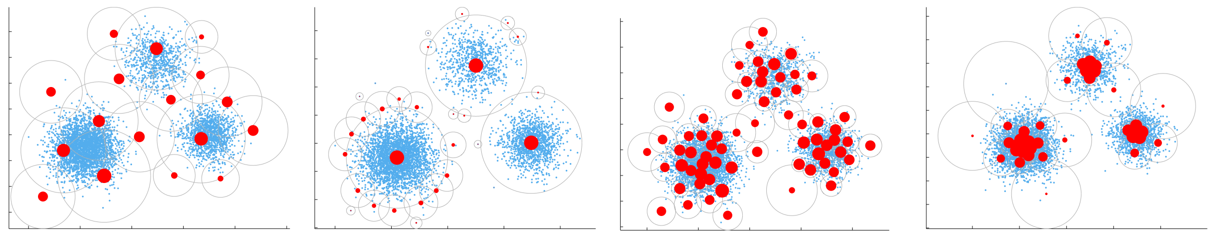


(a) Covering. (b) CoveredBy.

Figure 4.18: Using multiple radii based on density. Items in denser areas are associated with smaller radii. Selected items are shown as solid circles. Larger circles correspond to items in denser areas. (In the CoveredBy case, each item p_i not in the diverse subset is represented by an item in the diverse subset within distance $r(p_i)$ from it. Since there is a large number of such items, we do not draw their radius for clarity.)

tion with density, so that dense areas are not under-represented in the diverse subset. To achieve this, we assign smaller radii to items in denser areas of the dataset. Figure 4.18 shows the retrieved solutions for the Covering and CoveredBy variations of the multiple radii problem (in our example, items closer to the centers of their clusters are in denser areas and, thus, are assigned smaller radii). In both cases, the dense areas of the dataset are better represented in the diverse set due to their items being associated with smaller radii. Again, as explained above, in the CoveredBy case, more items from dense areas enter the diverse subset (see, for example, the items selected from the outskirts of the top cluster).

The third scenario corresponds to the case in which we want to relate representation with weights. For the Covering problem, we assign larger radii to items wither larger weights. This is to model the case where we want highly relevant items to cover a large area around them. For the CoveredBy problem, we assign smaller radii to items wither larger weights. This ensures that each item can be covered only by items that have a larger weight than it. We consider again the uniform and clustered distribution of weights (Figure 4.19). The Covering variation of the multiple radii problem works well for uniformly distributed weights. However, this does not seem to be the case for clustered weights. This happens because, in the latter case, items with large weights, and thus large radii, block each other from entering the diverse set. The CoveredBy variation does not face this issue. In general, the Covering variation results in smaller sets, since the algorithm starts with the item with the largest radii and, thus, with the item that possibly covers the largest number of items. However, in neighboring areas with different radii, the items with the larger radii disallow items with the smaller radii from entering the diverse set. Thus, when radii is associated with relevance weights, the CoveredBy approach produces better results in the sense that it allows more relevant items to be included in the diverse set.



(a) Uniform weights: Covering (b) Clustered weights: Covering (c) Uniform weights: CoveredBy (d) Clustered weights: CoveredBy

Figure 4.19: Using multiple radii based on weights. Items with larger weights are associated with larger radii in the Covering case and with smaller radii in the CoveredBy case. Selected items are shown as solid circles. Larger circles correspond to items with larger weights. (In the CoveredBy case, each item p_i not in the diverse subset is represented by an item in the diverse subset within distance $r(p_i)$ from it. Since there is a large number of such items, we do not draw their radius for clarity.)

4.5 Implementation

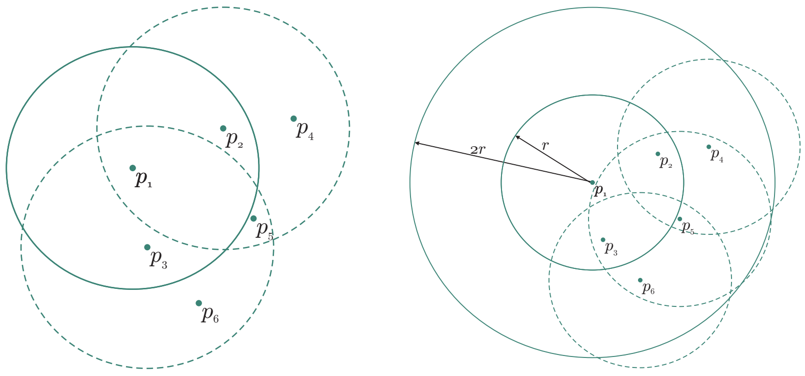
A central operation in computing DisC diverse subsets is locating neighbors. For this reason, we introduce implementations of our algorithms that exploit a spatial index structure, namely, the M-tree [33]. An M-tree is a balanced tree index that can handle large volumes of dynamic data of any dimensionality in general metric spaces. In particular, an M-tree partitions space around some of the indexed items, 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 item 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 items 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 items 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 the nodes of the trees. For computing diverse subsets:

(i) We link together all leaf nodes. This allows us to visit all items in a single left-to-right traversal of the leaf nodes and exploit some degree of locality in covering the items.

(ii) To compute the neighbors $N_r(p_i)$ of an item 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



(a) Grey-Greedy-DisC. (b) White-Greedy-DisC.

Figure 4.20: Greedy-DisC variations.

policy that resulted in the lowest overlap was (a) promoting as new pivots the pivot p_i of the overflowed node and the item p_j with the maximum distance from p_i and (b) partitioning the items by assigning each item to the node whose pivot has the closest distance with the item. We call this policy "MinOverlap".

4.5.1 Computing Diverse Subsets

Basic-DisC. The `Basic-DisC` algorithm selects white items in random order. In the M-tree implementation of `Basic- DisC`, we consider items in the order they appear in the leaves of the M-tree, thus taking advantage of locality. Upon encountering a white item 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 item 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 items in the leaf nodes from left-to-right until all items become either grey or black.

Greedy-DisC. The `Greedy-DisC` algorithm selects at each iteration the best white item according to the selection criterion C (line 6 of Algorithm 4.1). To efficiently implement this selection, we maintain a sorted list, L, of all white items ordered as C dictates, that is, by

- decreasing order of the size of their white neighborhood for the Minimum r-DisC Diverse Subset problem,
- decreasing order of the product of their weight and the (normalized) size of their white neighborhood for the Minimum Weighted r-DisC Diverse Subset problem, and
- decreasing (resp. increasing) order of their radius for the Covering (resp. CoveredBy) problem for the multiple radii case.

Remember that, in all cases, ties are resolved by selecting the item with the largest white neighborhood.

Instead of performing one range query per item after building the tree to compute the size of the white neighborhoods to initialize L, we compute such values incrementally as we build the M-tree. At first, for each item p_i, it holds that $N_r^W(p_i) = N_r(p_i)$. To compute the neighborhood size of each item incrementally, when an item p_i is inserted into the M-tree, a range query $Q(p_i, r)$ is executed, the white neighborhood of p_i is initialized to $|Q(p_i, r)|$ and the size of the white neighborhoods of all items retrieved by the range query are incremented by one. We found this incremental approach reduces node accesses up to 45%.

Considering the maintenance of L, each time an item p_i is selected and colored black, its neighbors at distance $r(p_i)$ are colored grey. Therefore, we need to update the ordering of a number of affected items. 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 $Q(p_j, r(p_j))$ 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 items within distance less than or equal to $r(p_i) + \max_{p_j \in N(p_i)} r(p_j)$ from p_i. These are the only white items whose white neighborhood may have changed. Consider, for example, the items of Figure 4.20. For simplicity, let all items have the same radius r. Assume that p_1 has just been colored black. Both variations will first execute a query $Q(p_1, r)$ to retrieve and color grey the neighbors of p_1, i.e., p_2 and p_3. After this, `Grey-Greedy-DisC` will execute the queries $Q(p_2, r)$ and $Q(p_3, r)$ (thus retrieving p_5 twice) to update the affected white neighborhoods of p_4, p_5 and p_6, while `Grey-Greedy-DisC` will execute $Q(p_1, 2r)$ instead. Since the cost of maintaining the exact size of the white neighborhoods may be large, we also consider "lazy" variations. `Lazy-Grey-Greedy-DisC` only retrieves grey neighbors at some distance smaller than $r(p_i)$, while `Lazy-White-Greedy-DisC` only retrieves white items at some distance smaller than $r(p_i) + \max_{p_j \in N(p_i)} r(p_j)$.

Pruning. We make the following observation that allows us to *prune* subtrees while executing range queries. Items 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 items 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.

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

Table 4.1: Input parameters.

Parameter	**Default value**	**Range**
M-tree node capacity	50	25 - 100
M-tree splitting policy	MinOverlap	various
Dataset cardinality	10000	579 - 50000
Dataset dimensionality	2	2 - 10
Dataset spatial distribution	clustered	uniform, clustered
Dataset weight distribution	clustered	uniform, clustered
Distance metric	Euclidean	Euclidean, Hamming
Radius assignment	uniform	uniform, weight-based

4.5.2 Adapting the Radius

For zooming-in, given an r-DisC diverse subset S of $\mathcal{P}$, we would like to compute an r'-DisC diverse subset S' of $\mathcal{P}$, $r' < r$, such that, $S' \supseteq S^r$. A naive implementation would require as a first step locating the items in $N^I_{r',r}(p_i)$ (line 3 of Algorithm 4.2) by invoking two range queries for each p_i (with radius r and r' respectively). Then, a diverse subset of the items in $N^I_{r',r}(p_i)$ is computed either in a basic or in a greedy manner. However, during the construction of S, items in the corresponding M-tree have already been colored black or grey. We use this information based on the following rule.

Zooming Rule: Black items of S maintain their color in S'. Grey items maintain their color as long as there exists a black item at distance at most r' from them.

Therefore, only grey nodes with no black neighbors at distance r' may turn black and enter S'. To apply this rule, we augment the leaf nodes of the M-tree with the distance of each indexed item p_i to its *closest* black neighbor p_j, since p_i will continue to be covered by p_j for all $r' \leq d(p_i, p_j)$.

The `Basic-Zoom-In` algorithm requires one pass of the leaf nodes. Each time a grey item 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 items for which p_i is now their closest black neighbor. At the end of the pass, the black items of the leaves form S'. The `Greedy-Zoom-In` algorithm involves the maintenance of a sorted structure L of all white items. To build this structure, the leaf nodes are traversed, grey items that are now found to be uncovered are colored white and inserted into L. After this, L is sorted accordingly.

Zooming-out algorithms are implemented similarly to the zooming-in case.

4.6 Experimental Evaluation

In this section, we evaluate the efficiency and effectiveness of our algorithms using both synthetic and real datasets. We first describe our datasets and algorithms and, then,

Table 4.2: Algorithms.

Algorithm	**Abbreviation**	**Description**
`Basic-DisC`	`B-DisC`	Single radius: Selects items in order of appearance in the leaf level of the M-tree. Multiple radii: Selects items in order of their radius.
`Greedy-DiSc`	`G-DisC`	Selects at each step the white item p_i with the largest value of $C(p_i)$.
$\rightarrow$ `Grey-Greedy-DisC`	`Gr-G-DisC`	One range query per grey node at distance $r(p_i)$ from p_i.
$\longrightarrow$ `Lazy-Grey-Greedy-DisC`	`L-Gr-G-DisC`	One range query per grey node at distance ${r(p_i)}/{2}$ from p_i
$\rightarrow$ `White-Greedy-DisC`	`Wh-G-DisC`	One range query per white node at distance $r(p_i) + \max_{p_j \in N(p_i)} r(p_j)$ from p_i.
$\longrightarrow$ `Lazy-White-Greedy-DisC`	`L-Wh-G-DisC`	One range query per white node at distance $r(p_i) + {\left(\max_{p_j \in N(p_i)} r(p_j)\right)}/{2}$ from p_i.
`Greedy-C`	`G-C`	Selects at each step the non-black item p_i with the largest value of $C(p_i)$.

present experimental results.

Datasets. Our synthetic datasets consist of multidimensional items, which are either uniformly distributed in space or form (hyper) spherical clusters of different sizes. We assign weights to items either uniformly or in a “clustered” manner around specific target items, so that items that are closer to the target items get larger weights than items further away. Clustered assignment is used to model the common case where we have large weights around specific items that correspond to different interpretations of the query. Thus, we have four combinations for our synthetic data based on the spatial and weight distributions, namely “Uniform-Uniform”, “Uniform-Clustered”, “Clustered-Uniform” and “Clustered-Clustered”. We also employ four real datasets. Three of them contain geographic information about (i) 5922 cities and villages in Greece (“Greek Cities”) [5], (ii) the 590 highest populated cities in the world (“World Cities”) [8] and (iii) 1000 apartments for sale in London (“Nestoria”) [6]. Weights are assigned uniformly for “Greek Cities”, based on higher population for “World Cities” and based on lower price for “Nestoria”. The fourth real dataset (“Cameras”) consists of 7 characteristics for 579 digital cameras, such as brand and storage type [1]. We assign weights based on a combination of the megapixels and the optical zoom of the cameras.

We normalize the values of all datasets in $[0, 1]$. We use the Euclidean distance for the synthetic and geographical datasets, while for “Cameras”, whose attributes are categorical, we use $d(p_i, p_j) = \sum_i \delta^i(p_i, p_j)$, where $\delta^i(p_i, p_j)$ 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 problem issue.

Algorithms. We next briefly summarize the algorithms used throughout this section.

- *Unweighted DisC*: We employ the `Basic-DisC` and `Greedy-DisC` algorithms. `Basic-DisC` simply selects a valid DisC diverse subset, while `Greedy-DisC` also attempts to minimize the size of the selected subset by selecting at each step the white item p_i with the largest $|N^W(p_i)|$. We use both variations of `Greedy-DisC`, i.e., `Grey-Greedy-DisC` and `White-Greedy-DisC`, as well as their lazy variations, i.e., `Lazy-Grey-Greedy-DisC` and `Lazy-White-Greedy-DisC` (for $r/2$ and $3r/2$ respectively), as described in Section 4.5.

- *Weighted DisC*: Again, we employ `Basic-DisC` and `Greedy-DisC`. The difference now is that the greedy algorithms select at each step the white item p_i with the largest $w(p_i) \left(|N_r^W(p_i)| / \max_{p_j \in \mathcal{P} \setminus S} |N_r^W(p_j)|\right)$. In case of ties, the item p_i with the largest $|N^W(p_i)|$ is selected.

- *Multiple radii DisC*: When each item is assigned a different radius, `Basic-DisC` is modified to consider items in decreasing or increasing order of their radius for the Covering and CoveredBy problems respectively. `Grey-DisC` also considers the items in order of their radius but, in addition, selects the item p_i with the largest $|N^W(p_i)|$ in case of ties. For the Covering problem, we use $r(p_j)$ and $r(p_j)/2$ for `Grey-Greedy-DisC` and `Lazy-Grey-Greedy-DisC` respectively. Note that, for the CoveredBy problem, our `White-Greedy-DisC` and `Lazy-White-Greedy-DisC` algorithms are no longer applicable, since at each step, after an item is selected for inclusion in the diverse set S, we have to check every other item p_j in $\mathcal{P} \setminus S$ to see whether $d(p_i, p_j) \leq r(p_j)$. Let r_{max} be the largest radius in the dataset. Then, the affected items at each step are at distance at most r_{max} from the selected item p_i. Our implementation of `Grey-Greedy-DisC` checks all items at distance r_{max} to update their white neighborhoods, while the lazy variation checks all items at some distance smaller than that.

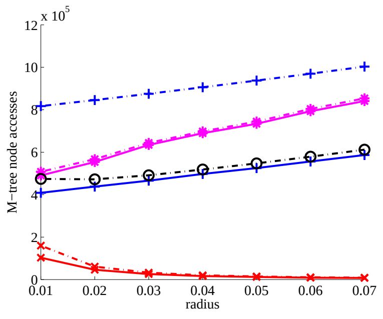
In all cases, we also consider the `Greedy-C` algorithm that produces covering but not necessarily independent sets. `Greedy-C` works as `Greedy-DisC` with the difference that both white and grey items are consider as candidates at each step.

Table 4.1 summarizes the values of the input parameters used in our experiments and Table 4.2 summarizes the algorithms employed.

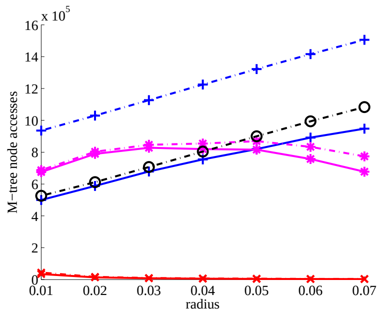
4.6.1 Unweighted DisC

We first consider the unweighted DisC case. We first compare the various algorithms for computing DisC subsets in terms of cost and of the size of the produced subset. We also evaluate the effect of specific characteristics of the datasets and of the M-tree. We conclude the evaluation with a comparison of the result of DisC with the results of other diversification methods.

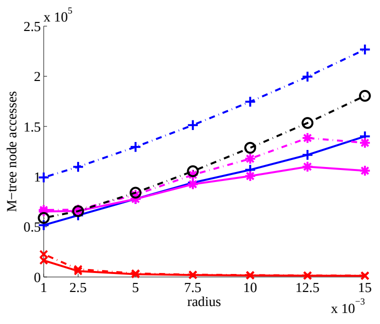
Computational Cost. We measure the computational cost of our algorithms in terms of node accesses in the employed M-trees. Figure 4.21 reports this cost, as well as, the



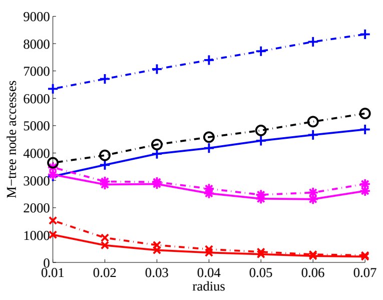
(a) Uniform.



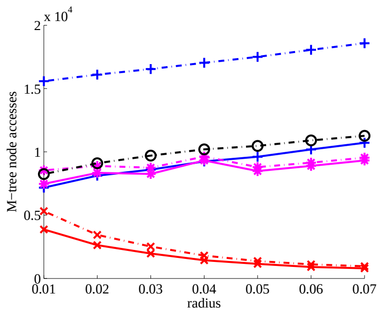
(b) Clustered.



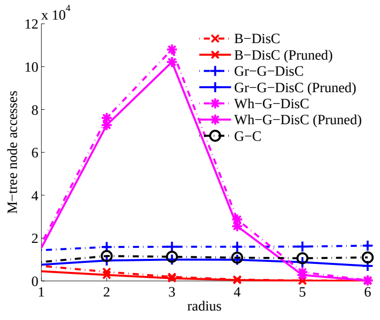
(c) Greek Cities.



(d) World Cities.



(e) Nestoria.

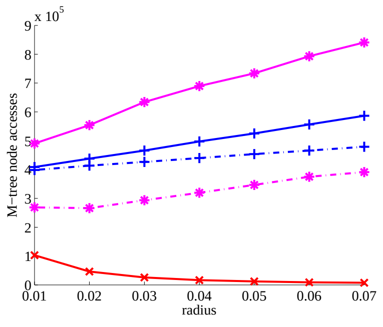


(f) Cameras.

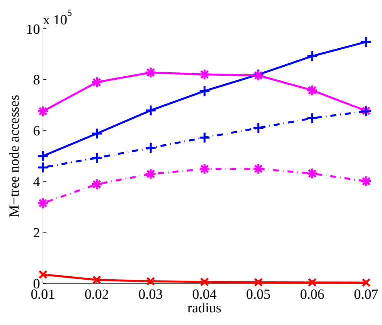
Figure 4.21: Node accesses for `Basic-DisC`, `Greedy-DisC` and `Greedy-C` with and without pruning for the unweighted case.

cost savings when the pruning rule of Section 4.5 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, items 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 items are colored grey by each selected (black) item 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%.

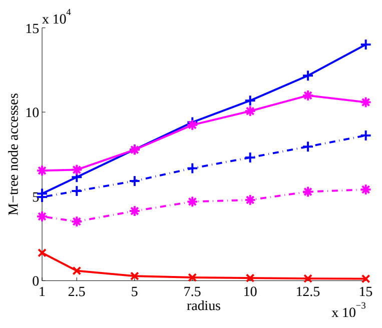
Figure 4.22 compares `Grey-Greedy-DisC` with `White-Greedy-DisC` and their corresponding lazy variations. We see that `White-Greedy-DisC` performs better than `Grey-Greedy-DisC` for the clustered dataset as r increases. This is because in this case, grey items share many common white neighbors which 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 as expected.



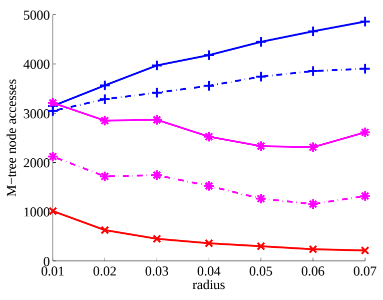
(a) Uniform.



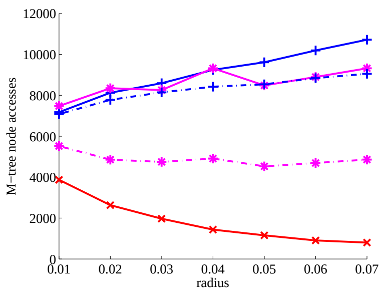
(b) Clustered.



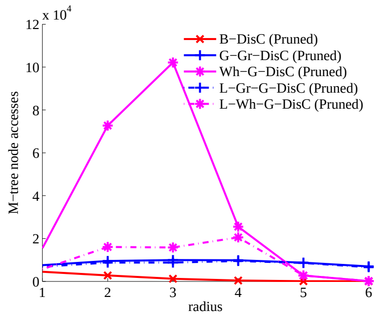
(c) Greek Cities.



(d) World Cities.



(e) Nestoria.



(f) Cameras.

Figure 4.22: Node accesses for `Basic-DisC` and all variations of `Greedy-DisC` with pruning for the unweighted case.

Solution Size. We next compare our various algorithms in terms of the size of the computed diverse subset (Table 4.3). We present results for our synthetic and one of our real datasets. Results are similar for the omitted datasets. 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 the 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-Grey-DisC` seems to approximate better the actual size of the white neighborhoods than `Lazy-Grey-Greedy-DisC` and 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.

Impact of Dataset Cardinality and Dimensionality. In the rest of this section, unless otherwise noted, we use the `(Grey-)Greedy-DisC (Pruned)` algorithm.

For this experiment, we employ the "Clustered" dataset. We vary its cardinality from 5000 to 15000 items and its dimensionality from 2 to 10 dimensions. Figure 4.23 shows the corresponding computational cost and solution size as computed by `Greedy-DisC`. We observe that the solution size is more sensitive to changes in

Table 4.3: Solution size (unweighted case).

(a) Uniform.

	r						
	0.01	**0.02**	**0.03**	**0.04**	**0.05**	**0.06**	**0.07**
B-DisC	3798	1364	679	395	272	187	145
G-DisC	3217	1133	571	352	230	170	132
L-Gr-G-DisC	3332	1250	635	378	252	180	143
L-Wh-G-DisC	3248	1160	571	354	243	167	131
G-C	3393	1113	558	347	224	163	128

(b) Clustered.

	r						
	0.01	**0.02**	**0.03**	**0.04**	**0.05**	**0.06**	**0.07**
B-DisC	1203	443	237	147	103	71	57
G-DisC	1060	376	197	127	88	66	47
L-Gr-G-DisC	1133	435	254	167	118	88	65
L-Wh-G-DisC	1059	377	194	125	86	64	47
G-C	1058	374	205	130	90	67	47

(c) World Cities.

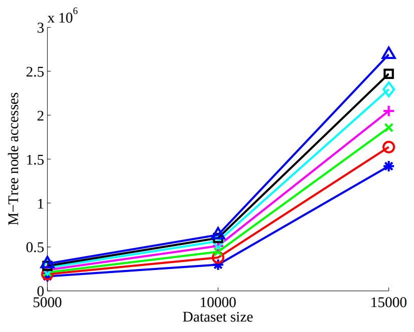
	r						
	0.01	**0.02**	**0.03**	**0.04**	**0.05**	**0.06**	**0.07**
B-DisC	288	161	104	75	56	39	33
G-DisC	274	129	87	63	44	33	28
L-Gr-G-DisC	276	141	95	68	50	35	32
L-Wh-G-DisC	274	136	87	61	48	35	30
G-C	282	137	89	64	43	34	30

cardinality when the radius is small. The reason for this is that for large radii, a selected item covers a large area in space. Therefore, even when the cardinality increases and there are many available items to choose from, these items are quickly covered by the selected ones. In Figure 4.23(a), 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.

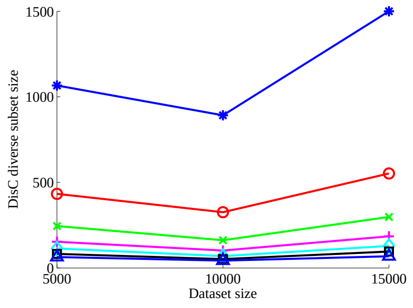
Increasing the dimensionality of the dataset causes more items to be selected as diverse as shown in Figure 4.23(d). This is due to the "curse of dimensionality" effect, since space becomes sparser at higher dimensions. The computational cost may however vary as dimensionality increases, since it is influenced by the cost of computing the neighborhood size of the items that are colored grey.

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 items are selected as diverse.

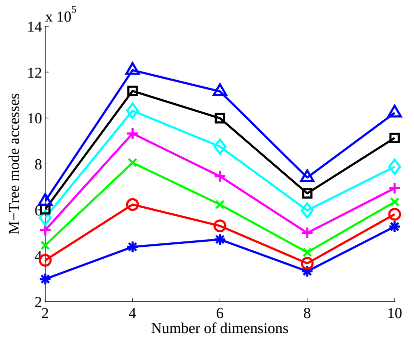
Different degree of overlap among the nodes of an M-tree may affect its efficiency for executing range queries. To quantify such overlap, we employ the *fat-factor* [63] of a



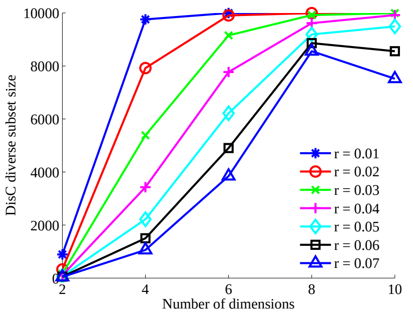
(a) Node accesses (dim = 2).



(b) Solution size (dim = 2).



(c) Node accesses (n = 10000).



(d) Solution size (n = 10000).

Figure 4.23: Varying (a)-(b) cardinality (for 2 dimensions) and (c)-(d) dimensionality (for 10000 items).

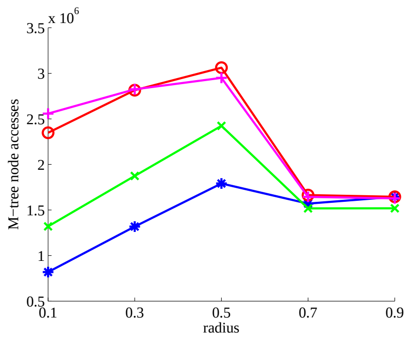
tree T defined as:

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

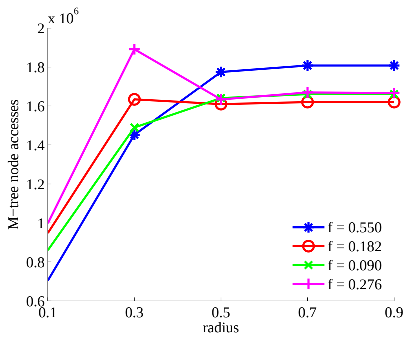
where Z denotes the total number of node accesses required to answer point queries for all items stored in the tree, n the number of these items, h the height of the tree and m the number of nodes in the tree. Ideally, the tree would require accessing one node per level for each point query which yields 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 policies which result in different fat-factors. We present results for four different policies. The lowest fat-factor was acquired by employing the “MinOverlap” policy. Selecting as new pivots the two items with the greatest distance from each other resulted in increased fat-factor. Even higher fat-factors were observed when assigning an equal number of items to each new node (instead of assigning each item 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 4.24 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 items 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



(a) Uniform.



(b) Clustered.

Figure 4.24: Varying M-tree fat-factor.

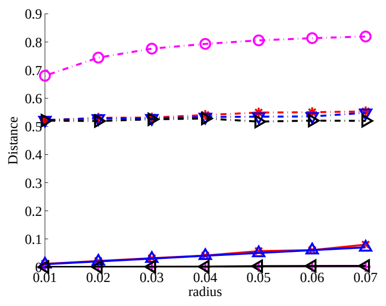
and locality. As the radius of the computed subset becomes very large, the solution size becomes very small, since a single item covers almost the entire dataset, this is why all lines of Figure 4.24 begin to converge for $r > 0.5$.

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 items; when doubling the node capacity, the computational cost was reduced by almost 45%.

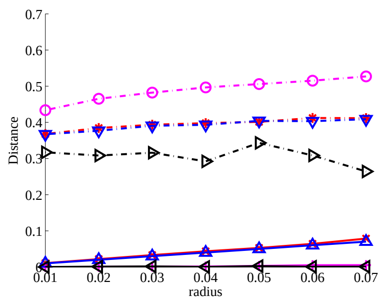
Comparison with Other Methods. Next, we compare the diverse subsets produced by DisC with those produced by other diversification methods, in particular by MaxMin and MaxSum. To implement MaxMin and MaxSum, we used the most widely used greedy algorithms for these problems. MaxMin selects items so as to maximize their minimum pairwise distance, while MaxSum so as to maximize their average pairwise distance. Both algorithms take as input the desired size k of the result. To compare the results, we compute DisC for various values of r. For each value of r, we compute MaxMin and MaxSum setting k equal to the size of the result produced by DisC. In Figure 4.25, we report the minimum and average pairwise distances of the results. The minimum distance among the items selected by `Greedy-DisC` is very close to that of MaxMin. For comparison, we also report the minimum and average distance among k random items. Note that, MaxMin and MaxSum attempt to optimize only the minimum and average distance respectively and do not consider coverage or other criteria.

4.6.2 Weighted DisC

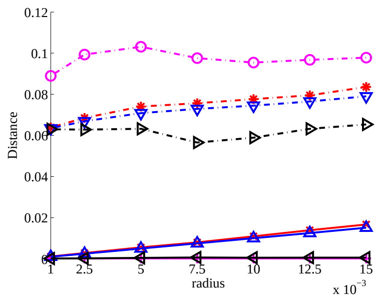
Next, we consider the weighted DisC problem and evaluate the quality (i.e, size, weight and diversity) of the results produced by various algorithms when the weight of the items is taken into account. We report results for `Basic-DisC`, `Greedy-DisC` and `Greedy-C` for the weighted version of the problem. We also report results for the lazy variations of `Greedy-DisC` (the two non-lazy versions of `Greedy-DisC` produce the same sets, only at different computational cost).



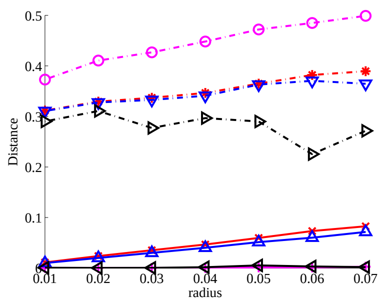
(a) Uniform.



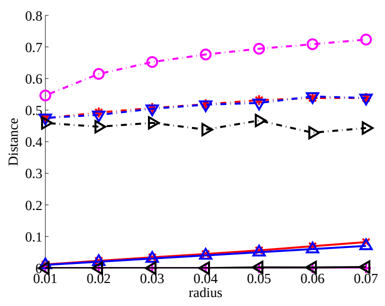
(b) Clustered.



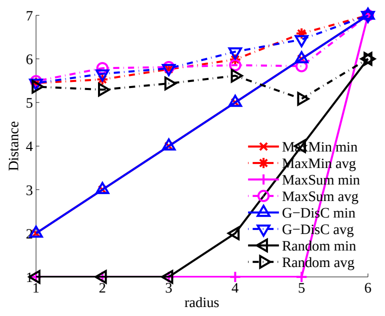
(c) Greek Cities.



(d) World Cities.



(e) Nestoria.



(f) Cameras.

Figure 4.25: Minimum and average distnace for MaxMin, MaxSum, Greedy-DisC and Greedy-C for the unweighted case.

Solution Size. The produced subsets are slightly larger than those of the unweighted version (Table 4.4) in all cases (except from Basic-DisC which is the same, since Basic-DisC produces an independent and covering subset without considering the size or the weight of the resulting subset). This happens because our algorithms are now selecting items based on both the weight and the neighborhood size of the items and, thus, items with large weights enter the diverse set even if they do not cover as many other items. Finally, note that, when weights are considered, the subsets produced for "Uniform-Clustered" are generally smaller than those of "Uniform-Uniform", since in this case items with larger weights are closer to each other.

Average Weight. Figure 4.26 shows the average weight of the subsets produced by our algorithms. We also report the average weight of the top-k items with the largest weights, for k equal to the size of the subset generated by Greedy-DisC. Greedy-C achieves a larger average weight. This happens because Greedy-C is not restricted to selecting dissimilar items and, thus, nearby items with large weights can all be selected. The Lazy-Grey-Greedy-DisC algorithm performs better than the non-lazy variation for the "Clustered-Clustered" dataset, since items with large weights are located nearby in that case and the lazy update of the white neighborhoods of the items allows more such items to enter the diverse subset.

Minimum Distance. Next, we report the minimum pairwise distance among the selected items for the weighted version of the problem as compared to selecting the top-k

Table 4.4: Solution size (weighted case).

(a) Uniform-Uniform.

	r						
	0.01	**0.02**	**0.03**	**0.04**	**0.05**	**0.06**	**0.07**
B-DisC	3798	1364	679	395	272	187	145
G-DisC	3557	1226	624	382	259	187	141
L-Gr-G-DisC	3601	1283	645	387	264	185	144
L-Wh-G-DisC	3560	1227	618	371	248	179	131
G-C	3643	1235	612	370	249	176	131

(b) Uniform-Clustered.

	r						
	0.01	**0.02**	**0.03**	**0.04**	**0.05**	**0.06**	**0.07**
B-DisC	3798	1364	679	395	272	187	145
G-DisC	3201	1133	569	358	234	168	126
L-Gr-G-DisC	3324	1250	640	378	255	180	142
L-Wh-G-DisC	3230	1149	576	359	232	173	128
G-C	3323	1103	558	336	229	162	123

(c) World Cities.

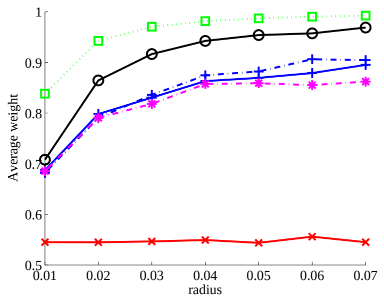
	r						
	0.01	**0.02**	**0.03**	**0.04**	**0.05**	**0.06**	**0.07**
B-DisC	288	161	104	75	56	39	33
G-DisC	288	160	94	73	55	38	33
L-Gr-G-DisC	292	158	94	73	57	41	34
L-Wh-G-DisC	289	159	95	73	55	38	33
G-C	301	165	100	82	60	40	35

most relevant items, for k equal to the size of the subset generated by `Greedy-DisC`. Figure 4.27 shows the results. We see that the results produced by just selecting the items with the top-k weights are very close to each other and thus exhibit very poor diversity. We also see that the increased average weight of `Greedy-C` (Figure 4.26) has the trade-off of selecting items that are much closer to each other, especially for smaller radii.

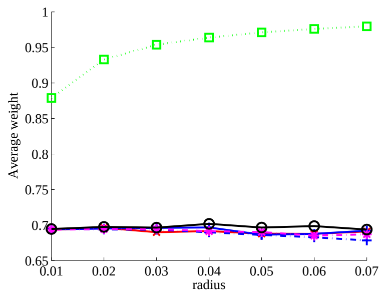
4.6.3 Multiple Radii DisC

Next, we evaluate some interesting issues concerning using multiple radii.

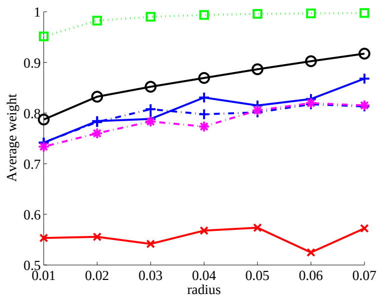
Tuning the radius of a specific area. First, we see how we can use multiple radii so as to tune the number of diverse items selected from a specific area of the dataset. For this experiment, we consider our "Uniform" dataset. We partition the dataset in four areas of equal size and set the radius of all items in the first three areas equal to 0.05, while varying the radius of the items in the fourth area from 0.01 to 0.10. Figure 4.28(a) reports the number of selected items from the fourth, "tunable" area, as well as, each of the other three "non-tunable" areas. We see that by varying the radius of the "tunable" area, we can over- or under-represent it in the diverse subset, according to our liking. The percentage of the items this tunable area contributes to the diverse set is depicted



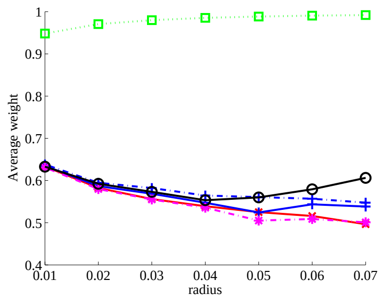
(a) Uniform-Uniform.



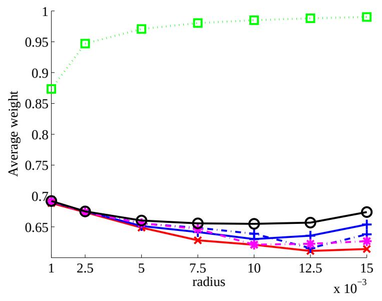
(b) Uniform-Clustered.



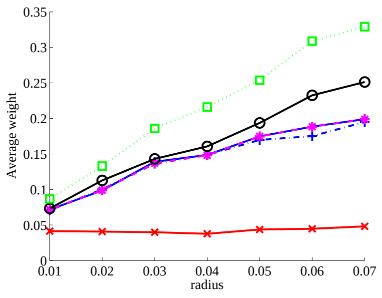
(c) Clustered-Uniform.



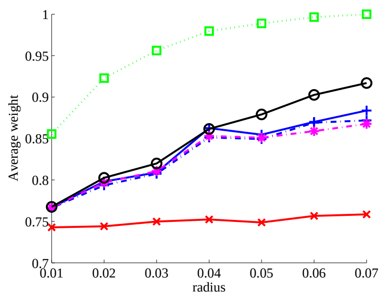
(d) Clustered-Clustered.



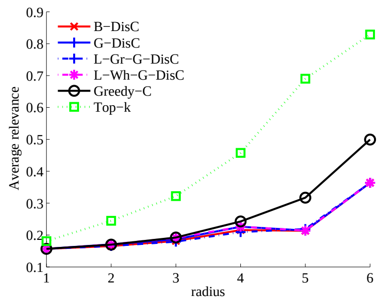
(e) Greek Cities.



(f) World Cities.



(g) Nestoria.

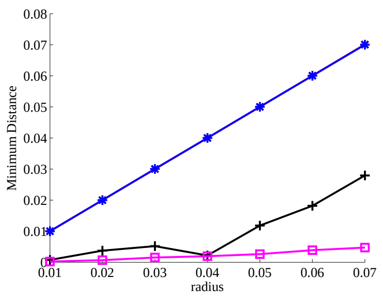


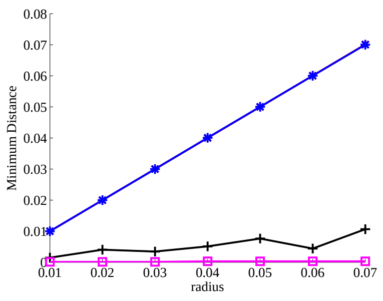
(h) Cameras.

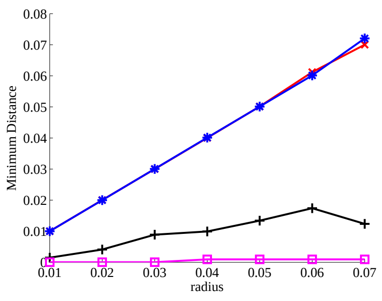
Figure 4.26: Average weight for `Basic-DisC`, `Greedy-C` and all variations of `Greedy-DisC` for the weighted case.

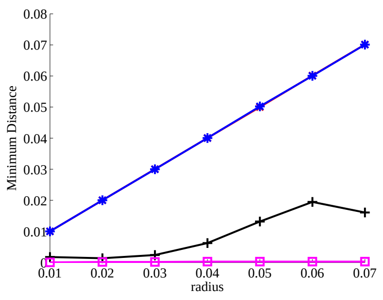
in Figure 4.28(b).

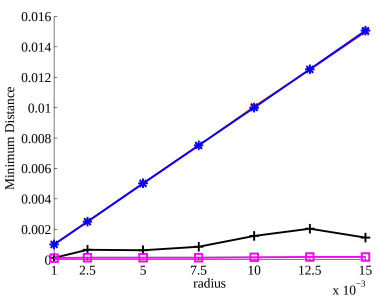
Covering vs. CoveredBy problems. Figure 4.29 shows the size and average weight for the Covering and CoveredBy problems for our uniform dataset, when weights are assigned both uniformly ("Uniform-Uniform") or in a clustered manner ("Uniform-Clustered"). For comparison, rather than assigning radii based on specific characteristics of our datasets, we assign radii uniformly in $(0, 2r]$, where r is the one shown in the x-axis of the figures. In the case of uniformly distributed weights, the Covering variation achieves a larger average weight. This, however, is not the case for clustered weights, where the CoveredBy variation performs better. This happens because, for CoveredBy, it is more difficult for items with large weight to block other items with large weight that are close to them from entering the diverse subset. This is also the reason, however, why the solutions retrieved by CoveredBy are generally larger in size.

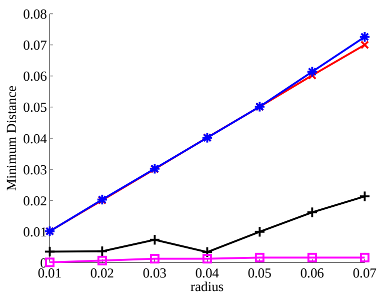

(a) Uniform-Uniform.

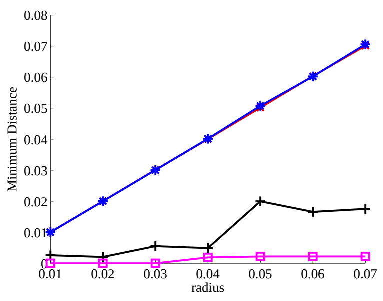

(b) Uniform-Clustered.

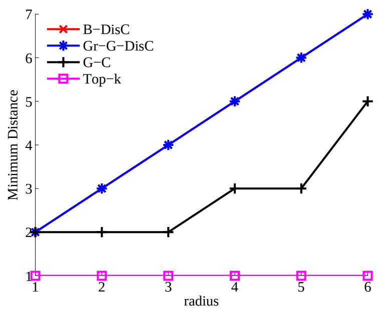

(c) Clustered-Uniform.


(d) Clustered-Clustered.


(e) Greek Cities.


(f) World Cities.


(g) Nestoria.



(h) Cameras.

Figure 4.27: Minimum distance among the selected items for `Greedy-DisC`, `Greedy-C` and the top-weighted items for the weighted case.

4.6.4 Zooming.

In the following, we evaluate our zooming algorithms. We begin with the zooming-in algorithms. 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 produced solutions. In general, we would like the solutions produced to share many common items so as to achieve a sense of continuity for the results presented to the user. We evaluate the similarity between solutions using the Jaccard distance. Given two sets S_1, S_2, their

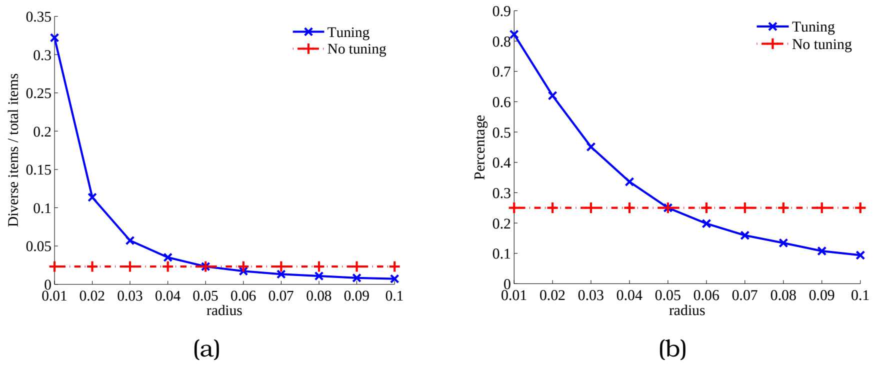


Figure 4.28: Tuning the radius of a specific area.

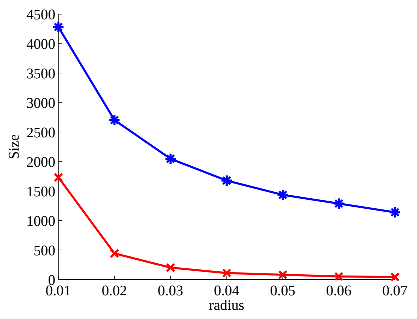
Jaccard distance is defined as:

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

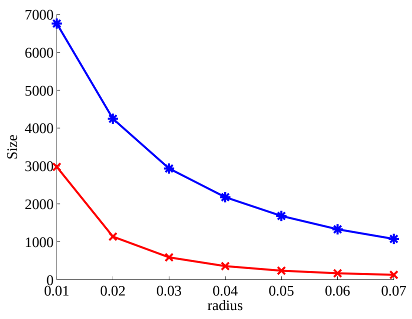
The Jacard distance is 0 when $S_1 = S_2$ and 1 when the S_1 and S_2 are disjoint. Figure 4.30 and Figure 4.31 report the corresponding results for different radii. Due to space limitations, we report results for the "Clustered" and "Cities" datasets. Similar results are obtained for the other datasets 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 4.30(a) and Figure 4.31(a) are adapted from the `Greedy-DisC` solution for $r = 0.03$.

We observe that the zooming-in algorithms provide similar solution sizes with those of `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 4.32). This means that computing a new solution for r' from scratch changes most of the items returned to the user, while a solution computed by a zooming-in algorithm maintains many common items in the new solution. Therefore, the new diverse subset is intuitively closer to what the user expects to receive.

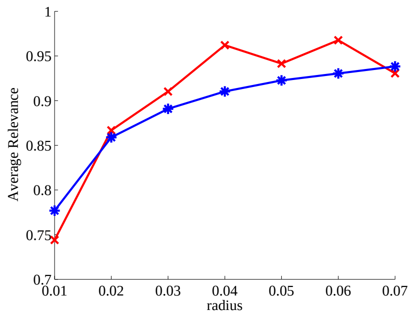
Figure 4.33 and Figure 4.34 show corresponding results for the zooming-out algorithms. The `Greedy-Zoom-Out(c)` algorithm achieves the smallest adapted DisC diverse subsets. However, its computational cost is very high and generally exceeds the cost of computing a new solution from scratch. `Greedy-Zoom-Out(a)` also achieves similar solution sizes with `Greedy-Zoom-Out(c)`, while its computational cost is much lower. The non-greedy algorithm has the lowest computational cost. Again, all the Jaccard distances of the zooming-out algorithms to the previously computed solution are smaller than that of `Greedy-DisC` (Figure 4.35), which indicates that a solution computed from scratch has only a few items in common from the initial DisC diverse set.



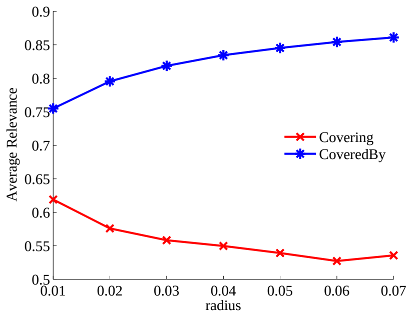
(a) Uniform-Uniform: size.



(b) Uniform-Clustered: size.



(c) Uniform-Uniform: average weight.



(d) Uniform-Clustered: average weight.

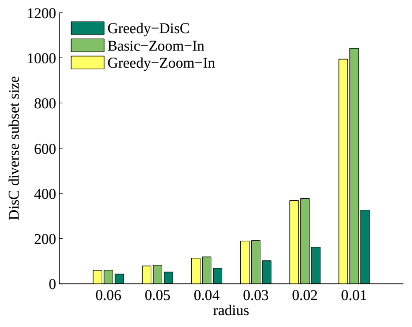
Figure 4.29: Covering vs. CoveredBy variations for the multiple radii case.

4.7 Related Work

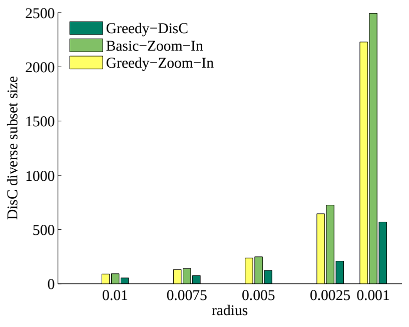
In this section, we first overview other diversity definitions proposed in the related literature and discuss how our DisC definition of diversity is related to them. Then, we review related work on the use of indices for the efficient implementation of diverse item selection. Finally, we present related work from the field of graph theory concerning independent and dominating sets.

Other Diversity Definitions. Diversity has recently attracted a lot of attention as a means of counter-acting the over-specialization problem and enhancing user satisfaction [109, 15, 57, 19]. Diverse results have been defined in various ways [42], namely in terms of *content* (or *similarity*), *novelty* and *semantic coverage*.

Most content-based definitions (e.g., [114]) interpret diversity as an instance of the p-*dispersion problem*, which is generally defined as selecting p out of n items, so that some objective function based on the chosen items is optimized. A number of variations of the p-dispersion problem have been extensively studied in the field of operations research (e.g., [48, 91, 26]). In the field of result diversification, the objective most usually employed is that of maximizing the *minimum* distance among any pair of selected items. This problem is most often referred to as the MaxMin diversification problem (e.g., [45]). Other works consider the MaxSum diversification problem instead (e.g., [109, 19]), whose objective is to select p out of n items, so that the *average* distance between the chosen items is maximized. Our approach here differs from

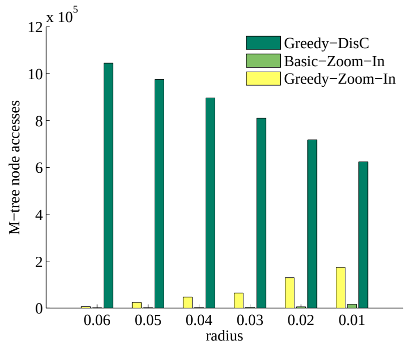


(a) Clustered.

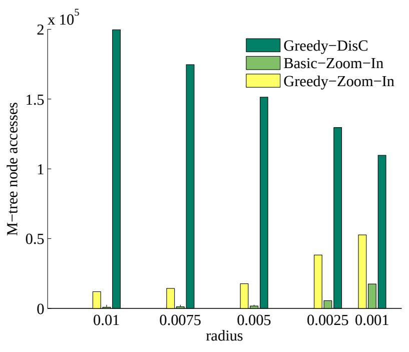


(b) Cities.

Figure 4.30: Solution size for zooming-in.



(a) Clustered.

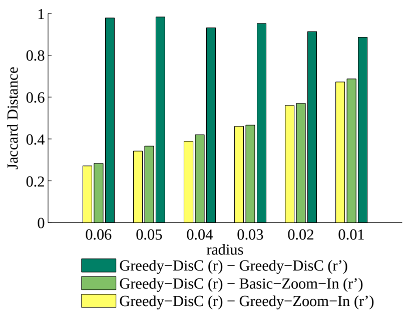


(b) Cities.

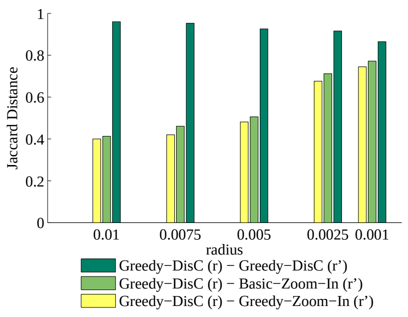
Figure 4.31: Node accesses for zooming-in.

those two traditional diversification problems, in that the size of the diverse subset is not an input parameter. Instead, users can explicitly define the desired degree of diversification via the radius r and later adapt the retrieved solutions by tuning r to see more or less diverse results.

Diversity is often combined with other criteria, most often that of *relevance*, to select items that are both highly relevant to a user query, as well as, diverse to each other. Assuming that each item p_i is associated with some weight $w(p_i)$, a common way to combine the two criteria is to use a diversification factor λ, $\lambda > 0$, and select the subset with the largest value of $\min_{p_i \in S} w(p_i) + \lambda \min_{\substack{p_i, p_j \in S \\ p_i \neq p_j}} d(p_i, p_j)$ and $(k - 1) \sum_{p_i \in S} w(p_i) + 2\lambda \sum_{p_i, p_j \in S} d(p_i, p_j)$ for the MaxMin and MaxSum problems respectively (e.g., [57]). Another common approach is Maximal Marginal Relevance (MMR) [23], in which weights and diversity are linearly combined when items are selected. For example, for the MaxMin problem, for some λ', $0 \leq \lambda' \leq 1$, at each iteration of the various greedy algorithms, the item p_i with the largest value of $\lambda' w(p_i) + (1 - \lambda') \min_{p_j \in S} d(p_i, p_j)$ is selected. Our approach here is different, since we are not restricted by the number k of selected items but, instead, seek a subset of dissimilar items that can cover the available space. Therefore, we aim at selecting a valid DisC subset that minimizes the objective of Definition 4.3, i.e., we favor smaller DisC subsets containing highly relevant items. Also, we can employ multiple radii to tune the importance of different areas in

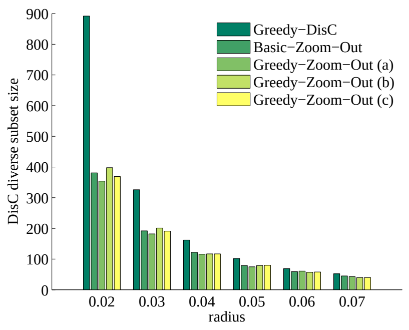


(a) Clustered.

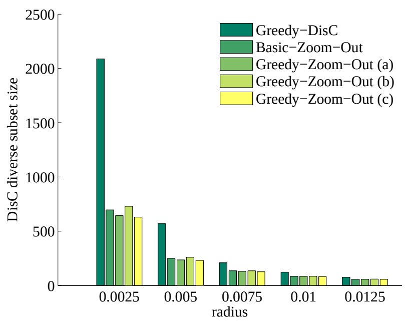


(b) Cities.

Figure 4.32: Jaccard distance for zooming-in.



(a) Clustered.

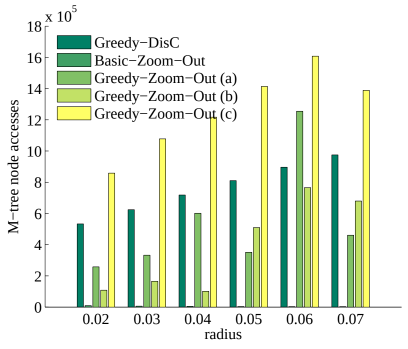


(b) Cities.

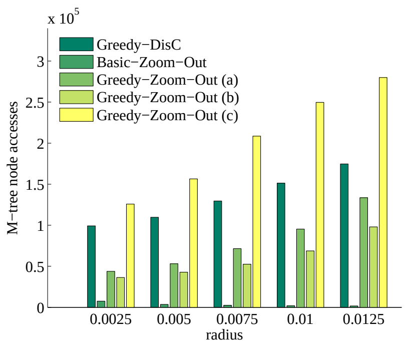
Figure 4.33: Solution size for zooming-out.

the dataset, as opposed to treating all items in the same way as in those traditional approach.

Another line of research aims at selecting diverse results similarly to top-k results by employing some sort of threshold algorithm, often attempting to incorporate weights to this threshold (e.g., [87, 22]). This approach is more common in novelty-based definitions of diversity in information retrieval (e.g., [35, 115]). There is a crucial difference between the two problems, however, in that the diversity of a single item cannot be computed independently from the other items as in the top-k case, since all diversity measures require comparing the item with any previously selected ones. In this spirit, [24] presents an approach for low-dimensional vector spaces in which the computation of the solution requires the availability of both relevance-based and distance-based sorted access methods. Items are selected in rounds; at each round a portion of the available space around the already selected items is pruned from further consideration. A number of variations of sorted and random accesses are also employed in [15] to retrieve a top-k list of relevant and diverse results. However, the focus of that work is on scheduling the order of the various accesses for cost efficiency rather than maximizing the quality of the retrieved solutions. Generally, such diversity threshold scores are hard to interpret, since they do not depend solely on the item. Instead, the score of each item is relative to which items precede it in the rank. Our approach is fundamentally

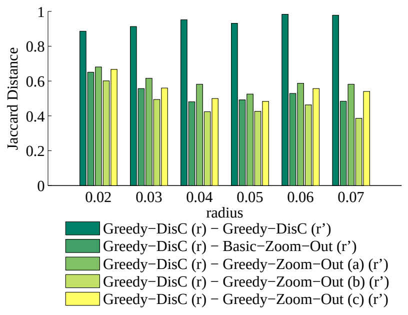


(a) Clustered.

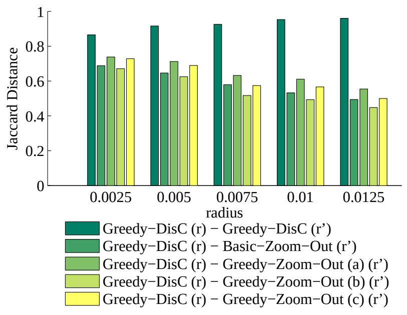


(b) Cities.

Figure 4.34: Node accesses for zooming-out.



(a) Clustered.



(b) Cities.

Figure 4.35: Jaccard distance for zooming-out.

different in that we treat the result as a whole and select DisC diverse subsets of it that fully cover it.

Another related problem is that of extending nearest neighbor search to selecting k neighbors that are not only spatially close to the query item but are also diverse to each other [92, 62, 10]. Such works usually focus on exploiting thresholds and space pruning techniques to enforce diversity during the retrieval of the nearest neighbors of a query item. Our work is different since our goal is not to locate the nearest and most diverse neighbors of a single item but rather to locate an independent and covering subset of the whole dataset. On a related issue, selecting k representative skyline items is considered in [101], where representative items are selected so that the distance between a non-selected skyline item from its nearest selected item is minimized and [105], where the dominance relationships among items are exploited to select a diverse subset of the skyline items.

Clustering is a research field related to that of diversification, since cluster medoids can be viewed as representative items (e.g., [79]). Medoids were extended in [18] to include some sense of relevance (priority medoids). However, there are fundamental differences between the two problems, since clustering aims at selecting representatives that minimize some intra-cluster distance, which leads medoids to be drawn to dense areas of items. Thus, items selected by clustering algorithms may not be as distant from each other as items selected by diversification algorithms. Furthermore, medoids may not cover all the available space. Perhaps the clustering works mostly related to our are those on detecting *distance-based outliers* (e.g., [70]). An item is considered a distance-based outlier if there are less than m other items lying at distance at most r from it. However, outliers and r-DisC diverse items are two distinctive sets. A major difference is that the decision whether an item is an outlier depends only on the size of its neighborhood and not on whether other neighbor items are outliers or not. This is not the case for r-DisC diverse items, since the decision to select an item as r-DisC diverse affects other items as well.

Finally, the problem of diversifying *continuous* data has been recently considered in [45, 86, 84] using a number of variations of the MaxMin and MaxSum diversification models.

Index-Based Implementations of Diversification Algorithms. Due to the NP-hardness of the diversification problem (e.g., [38]), many heuristics have been proposed for locating approximate solutions. Most of these can be classified as either *greedy* or *interchange* (*swap*) heuristics [42]. In greedy heuristics, items are iteratively selected in rounds in the diverse set. The complexity of greedy heuristics in terms of computed distances ranges from $O(k^2n)$ to $O(n^2)$ depending on the initialization step, while $^1/_2$-approximations of the optimal solutions can be achieved for both the MaxMin and MaxSum problems (e.g., [100]). Interchange heuristics initialize S with a random solution and then iteratively attempt to improve it by interchanging an item in S with another item that is not in S. Their worst case complexity is $O(n^k)$.

Here, we used the M-tree to implement our approach and exploited its properties, as well as our pruning rule, to reduce the computational cost of our approach. Indices have been used in the past for result diversification, most recently in [45], where a number of algorithms, as well as a fast implementation of the greedy heuristic, based on Cover Trees are proposed for the MaxMin diversification problem. A Dewey encoding of database tuples enables them to be organized in a tree structure which is later exploited to select the k most diverse of them in [107]. A similar approach is followed in [76]. However, the proposed methods are limited to a specific diversity measure and cannot be applied in the general case. A spatial index is also exploited in [59] to locate those relevant nearest neighbors of an item that are the most distant to each other.

Results from Graph Theory. The properties of independent and dominating (or covering) subsets have been extensively studied in graph theory. A number of different variations exist (e.g., [58, 55]). Among these, the *Minimum Independent Dominating Set* problem is equivalent to the r-DisC diversity problem. The problem of locating a Minimum Independent Dominating Set has been shown to have some of the strongest negative approximation results: in the general case, it cannot be approximated in polynomial time within a factor of $n^{1-\epsilon}$ for any $\epsilon > 0$ unless $P = NP$ [58]. However, some approximation results and faster algorithms have been found for special graph cases, such as bounded degree graphs and unit disk graphs (e.g, [32, 14, 52, 36, 56, 20]). 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 and their adaptation to new radii.

Weighted variations of independent and dominating graphs also exist (e.g., [120, 65]). Most commonly, graph vertices are associated with some weight and we seek to locate the independent and dominating subgraph with the maximum or the minimum sum of weights. This is usually referred as the *Maximum* (or *Minimum*) *Weighted Independent Dominating Set* problem. Clearly, the weighted case is at least as hard to be solved as the unweighted one. Moreover, since most approximation bounds for the unweighted problem rely on the average degree of the underlying graph, locating such bounds for the weighted case is even more challenging, since adding vertices of any weight can arbitrarily change the average degree of the graph.

We have seen that our multiple radii approach can be modeled via directed graphs. Related work on directed graphs is considerably more limited. An extra challenge in this case is that not all directed graphs have an independent dominating set. In [83] a number of conditions for the existence of an independent dominating set for a number of different kinds of graphs are provided. Some related work also exists on locating minimum dominating (but not independent) sets (e.g., [85, 28]).

Finally, 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 (e.g, [103]). However, allowing the dominating set to be connected has an impact on the complexity of the problem and allows different algorithms to be designed. Here, we require diverse items to be dissimilar to each other, thus, such approaches cannot be exploited.

4.8 Summary

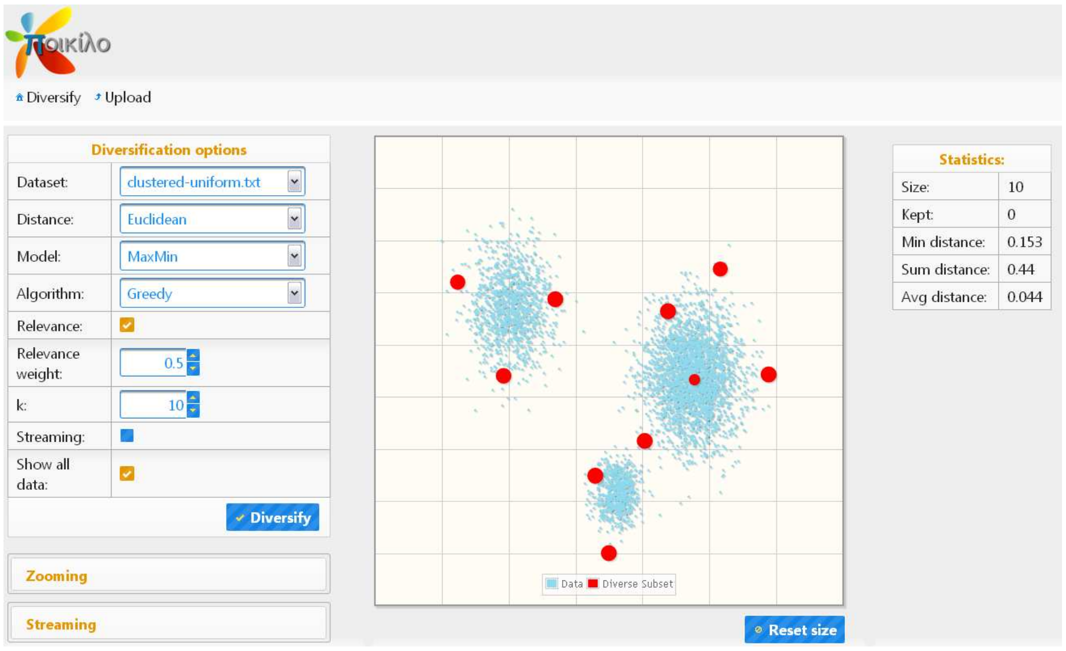
In this chapter, we proposed a novel, intuitive definition of diversity as the problem of selecting a minimum representative subset S of a result $\mathcal{P}$, such that each item in $\mathcal{P}$ is represented by a similar item in S and that the items included in S are not similar to each other. Similarity is modeled by a radius r around each item. We call such subsets r-DisC diverse subsets of $\mathcal{P}$. We introduced weighted and multiple radii variations of DisC subsets and, also, 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 efficient implementations based on spatial indexing.

Chapter 5

Poikilo: A System for Evaluating the Results of Diversification Models and Algorithms

In the previous chapters, we described result diversification in detail and detailed a wide range of its applications. Different diversification methods aim at optimizing different diversification criteria. Often, it is not clear what method is more suitable for a specific application. Here, we present Poikilo (from the greek *ποικίλο*, meaning "diverse"), a system designed to assist users in locating, visualizing and comparing diverse results based on a suite of different diversification models and algorithms. We provide implementations of a wide variety of diversification approaches for retrieving diverse results. For the case in which the degree of diversification is specified by a radius, we also provide an interactive zoom-in and zoom-out form of functionality (Figure 5.2).

Often, results are associated with a relevance score. Poikilo includes various methods for combining relevance and diversity in selecting representative results. Furthermore, we consider the case of streaming data, where the query results change over time and so does the diverse result presented to the users. We employ a sliding window streaming model and provide options to navigate between consequent windows of diverse results.



Diversification options	
Dataset:	clustered-uniform.txt
Distance:	Euclidean
Model:	MaxMin
Algorithm:	Greedy
Relevance:	✓
Relevance weight:	0.5
k:	10
Streaming:	
Show all data:	✓

Statistics:	
Size:	10
Kept:	0
Min distance:	0.153
Sum distance:	0.44
Avg distance:	0.044

Figure 5.1: POIKILO user interface. Users can upload data, configure diversification options and see diversified results. Diverse results are shown as solid circles, while their size varies based on their relevance.

Users of POIKILO can submit queries to a number of different datasets and see a visualization of a diversified subset of their query result (Figure 5.1). We provide various synthetic and real datasets. Users can also upload their own datasets. Users can choose among a wide selection of diversification algorithms and specify various configuration parameters. Furthermore, they can zoom-in and zoom-out of this initial diverse subset and navigate between consequent windows in the case of streaming data.

5.1 Diversity Models

Various models have been proposed for result diversification [42]. In this section, we describe the various models made available to users by POIKILO. Most of these models involve the use of a distance function. We have implemented the most common distance functions (e.g., Euclidean, cosine). In addition, users can select which of the attributes of each item will be used for diversification.

Dispersion models. The most widespread diversity models are related to the k-dispersion problem, defined as selecting k out of a set $\mathcal{P}$ of items in some space, such that some objective function is maximized. Common variations include the MAXMIN and MAXSUM methods. Given a distance metric d and an integer k, $k > 1$, MAXMIN aims at locating a subset S of $\mathcal{P}$ with k items, such that, the minimum pairwise distance among any items in S is maximized. MAXSUM, on the other hand, aims at maximizing the sum of the respective pairwise distances. That is, the two models aim at maximizing

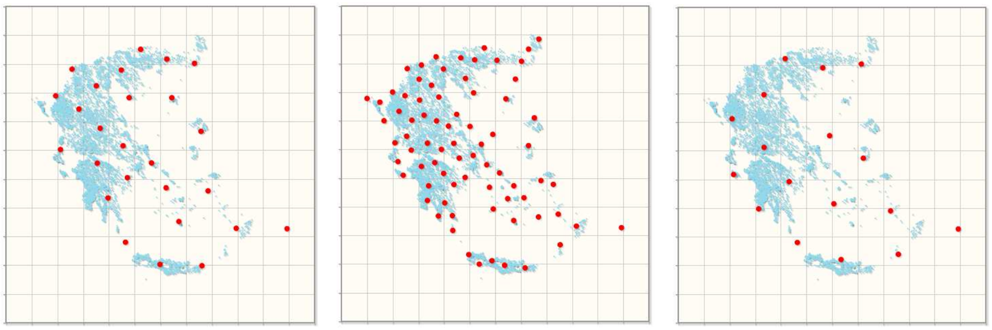

(a) Initial diverse items. (b) Zooming-in. (c) Zooming-out.

Figure 5.2: Zooming operations in action in POIKILO. Selected items are shown as solid circles.

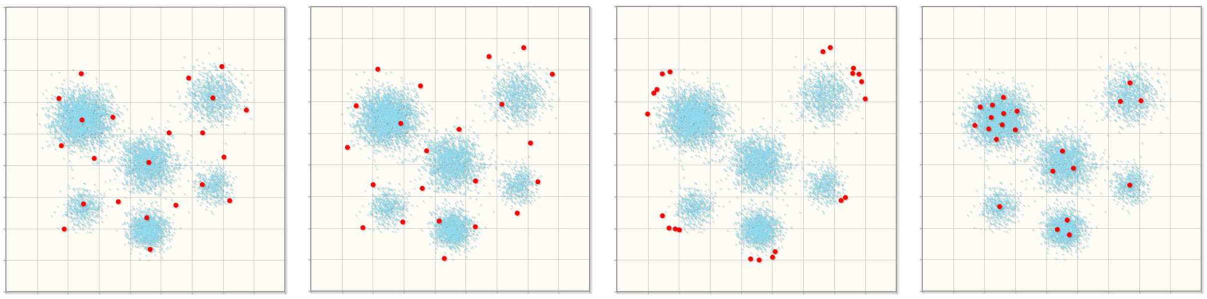

(a) DisC. (b) MaxMin. (c) MaxSum. (d) k-Medoids.

Figure 5.3: Comparison of various diversification models.

the following diversity functions:

$$f_{\text{MIN}}(S,d) = \min_{p_i,p_j \in S} d(p_i,p_j) \text{ and } f_{\text{SUM}}(S,d) = \sum_{p_i,p_j \in S} d(p_i,p_j) \tag{5.1}$$

Intuitively, MAXMIN aims at discouraging the selection of nearby items, while MAXSUM at increasing the average pairwise distance among all items.

DisC diversity. DisC is a recently proposed model that combines coverage and diversity [44]. Let $N_r(p_i)$ be the *neighborhood* of an item $p_i \in \mathcal{P}$, i.e., the items lying at distance at most r from p_i. r, $r \geq 0$, is a tuning parameter called *radius*. Let also $N_r^+(p_i)$ be the set $N_r(p_i) \cup \{p_i\}$. Intuitively, we would like to select exactly one item from each item's neighborhood.

Definition 5.1. (r-DISC DIVERSE SUBSET) Let $\mathcal{P}$ be a set of items and r, $r \geq 0$, a real number. A subset $S \subseteq \mathcal{P}$ is an r-*Dissimilar and Covering* 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 $i \neq j$, it holds that $d(p_i,p_j) > r$.

Given $\mathcal{P}$, we would like to select the smallest number of dissimilar and covering items.

Definition 5.2. (MINIMUM r-DISC DIVERSE SUBSET) Given a set $\mathcal{P}$ of items 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 $|S^*| \leq |S|$.

The DisC model allows an interactive mode of operation where, after being presented with an initial set of results for some radius r, a user can see either more or less results by decreasing or increasing r. Specifically, given a set of items $\mathcal{P}$ and an r-DisC diverse subset S^r of $\mathcal{P}$, we want to compute an r'-DisC diverse subset $S^{r'}$ of $\mathcal{P}$. Zooming can be global, in the sense that the radius r is modified similarly for all items in $\mathcal{P}$, or local, i.e., modifying the radius only for a specific area of the data set. To support an incremental mode of operation, the set $S^{r'}$ should be as close as possible to the already seen result S^r. Ideally, $S^{r'} \supseteq S^r$, for $r' < r$ and $S^{r'} \subseteq S^r$, for $r' > r$. Although in general there is no monotonic property among the optimal r-DisC diverse and r'-DisC diverse subsets of a set of items $\mathcal{P}$, for $r \neq r'$, we provide heuristics that achieve these requirements.

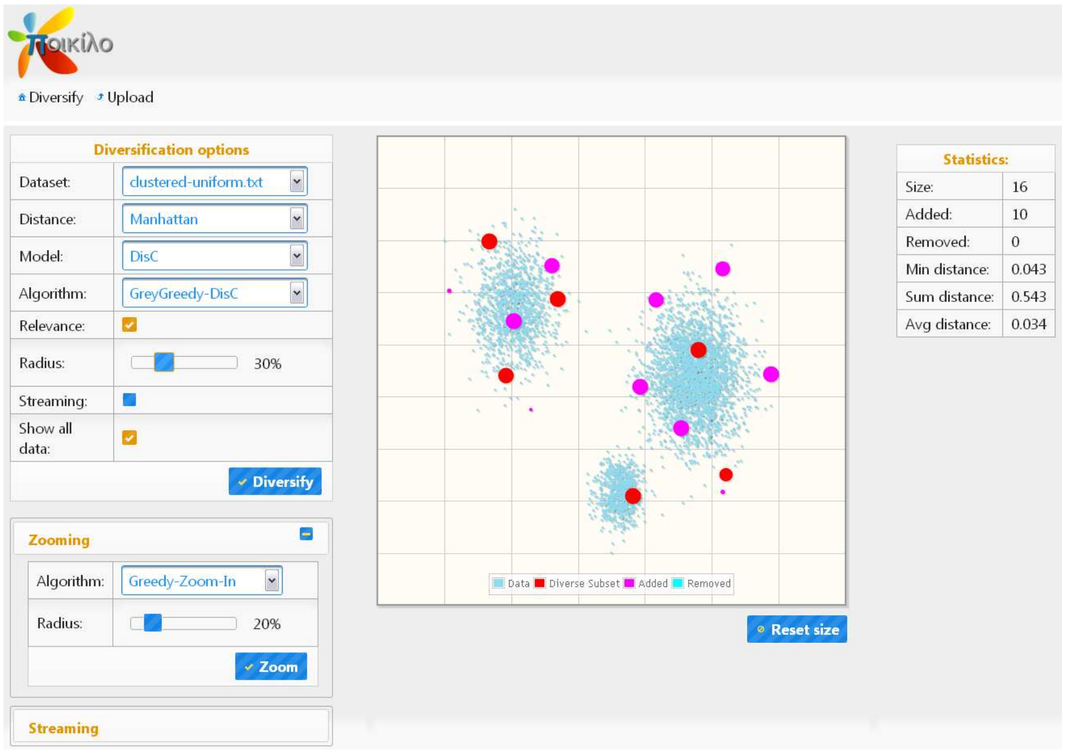
Other models. Often, clustering methods have been proposed as an alternative to selecting diverse items. In this case, the diverse set consists of representatives from each cluster. For example, k-medoids seeks to minimize $\frac{1}{|\mathcal{P}|} \sum_{p_i \in \mathcal{P}} d(p_i, c(p_i))$, where $c(p_i)$ is the closest item of p_i in the selected subset. We also consider other diversification models, such as the Greedy Marginal Contribution and Greedy Randomized with Neighborhood Expansion models presented in [109]. Our tool can be easily extended with additional methods as well.

Figure 5.3 shows the diverse sets located by POIKILO for some of the different approaches. Generally, MAXSUM 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 sparser areas. 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 central areas of the clusters in Figure 5.3. 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.

5.2 Algorithms

Due to the NP-hardness of most of the models of the diversification problem, a number of different heuristics have been proposed (e.g., see [48]). POIKILO provides various implementations of different variations of such heuristics.

For MAXMIN and MAXSUM, a simple iterative greedy heuristic has been shown to provide $1/2$-approximations of the optimal solution. In this heuristic, first, the two furthest apart items of $\mathcal{P}$ are added to S. Then, at each iteration, one more item is added to S. The item that is added is the one that has the maximum distance from



Diversification options

Dataset:	clustered-uniform.txt
Distance:	Manhattan
Model:	DisC
Algorithm:	GreyGreedy-DisC
Relevance:	✓
Radius:	30%
Streaming:	
Show all data:	✓

Zooming

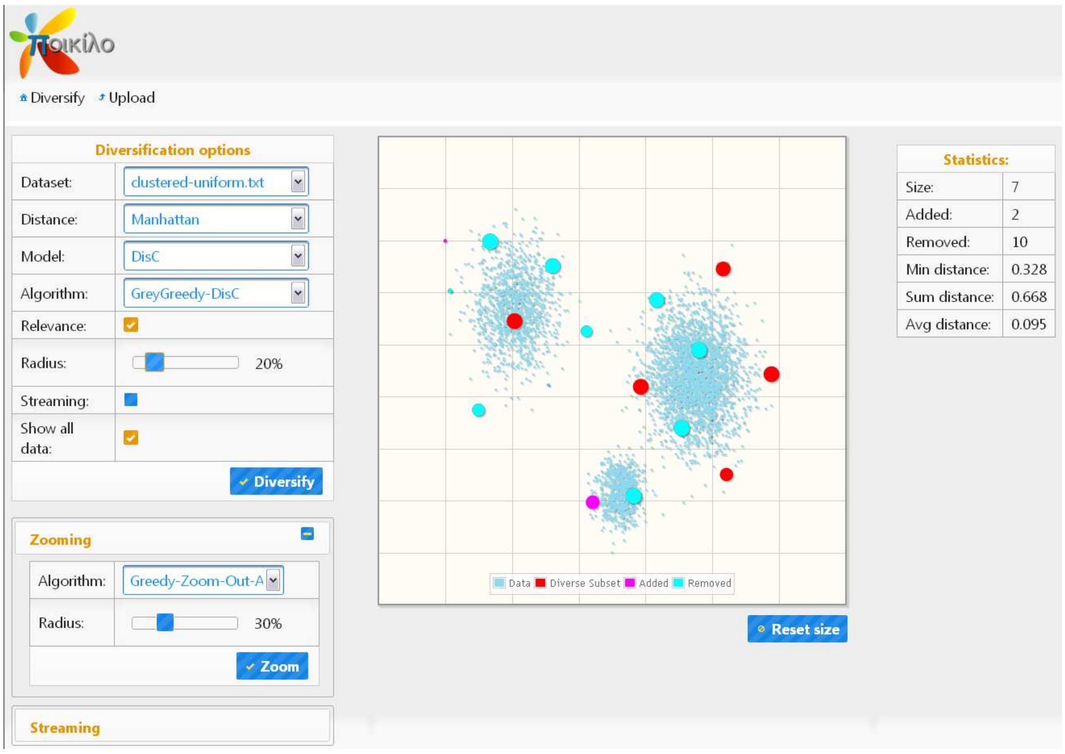
Algorithm:	Greedy-Zoom-In
Radius:	20%

Statistics:	
Size:	16
Added:	10
Removed:	0
Min distance:	0.043
Sum distance:	0.543
Avg distance:	0.034

Figure 5.4: Zooming-in in action.

the items already in S. Interchange heuristics are often used as well. Such heuristics are initialized with a random solution S and then iteratively attempt to improve that solution by interchanging an item in the solution with another item that is not in the solution. Usually, the item that is eliminated from the solution at each iteration is one of the two closest items in it. We provide various interchange heuristics, e.g., performing at each iteration the first interchange that improves the solution (*First-Interchange*) or considering all possible interchanges and perform the one that improves the solution the most (*Best-Interchange*).

Poikilo also provides an implementation of all the algorithms presented in [44] for computing DisC diverse subsets. These are graph-based algorithms that use a spatial index structure, namely the M-tree, to efficiently execute neighborhood queries. We briefly describe some of them next. Let us call *black* the items of $\mathcal{P}$ that are in S, *grey* the items covered by S and *white* the items that are neither black nor grey. The *Basic-DisC* heuristic initially considers that S is empty and all items are white. The algorithm proceeds in rounds; until there are no more white items, it selects an arbitrary white item p_i, colors p_i black and colors all items in $N_r(p_i)$ grey. The *Greedy-DisC* heuristic, instead of selecting white items arbitrarily at each round, selects the white item with the largest number of white neighbors, that is, the white item that covers the largest number of uncovered items. For zooming-in, i.e., for $r' < r$, we can construct r'-DisC diverse sets that are supersets of S^r by adding items to S^r (Figure 5.4). The items to be added are either selected randomly or in a greedy manner, where at each turn the item that covers the largest number of uncovered items is selected. For zooming-out, i.e.,



Diversification options

Dataset:	clustered-uniform.txt
Distance:	Manhattan
Model:	DisC
Algorithm:	GreyGreedy-DisC
Relevance:	✓
Radius:	20%
Streaming:	■
Show all data:	✓

Zooming

Algorithm:	Greedy-Zoom-Out-A
Radius:	30%

Statistics:

Size:	7
Added:	2
Removed:	10
Min distance:	0.328
Sum distance:	0.668
Avg distance:	0.095

Figure 5.5: Zooming-out in action.

for $r' > r$, in general, there may be no subset of S^r that is r'-DisC diverse. We provide a suite of algorithms that focus on minimizing $S^r \backslash S^{r'}$, i.e., the set of items that belong to the previous diverse subset but are removed from the new one, and $S^{r'} \backslash S^r$, i.e., the set of the new items added to $S^{r'}$ (Figure 5.5).

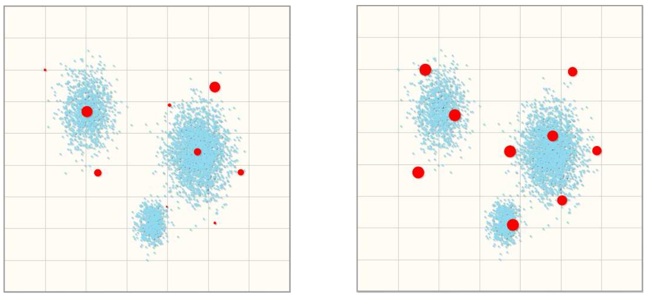
5.3 Other Features

We also consider a number of aspects complimentary to diversification, namely, combining diversity with relevance and handling streaming data.

5.3.1 Relevance

In many cases, the items in a result set are associated with a relevance score, most often based on their relevance to the user query. In such cases, it is important to retrieve items that are highly relevant to the user query. In general, the relevance score of an item is application dependent. Without loss of generality, we assume a relevance function $w : \mathcal{P} \rightarrow \mathbb{R}^+$ that assigns a relevance score to each item, where a higher value indicates that the item is more relevant to a particular query.

Dispersion-based models combine relevance and diversity using parameters for tuning the degree of diversification. Most common approaches use weights, for example a parameter σ, $0 \leq \sigma \leq 1$, to weight the relevance of each item against its distance



(a) Diversity only. (b) Diversity and relevance.

Figure 5.6: Combining diversity and relevance. Larger item size denotes higher relevance. In (a) some areas are covered by items with very low relevance, while in (b) highly relevant items are selected.

from other items during the selection process (a method called MMR [23]) or using a parameter λ, $\lambda \geq 0$ to favor the selection of diverse results among relevant ones. In the latter case, the corresponding relevance-aware diversity functions for MaxMin and MaxSum are:

$$f_{\text{MIN}}(S,d) = \min_{p_i \in S} rel(p_i) + \lambda \min_{p_i, p_j \in S} d(p_i, p_j) \text{ and} \tag{5.2}$$

$$f_{\text{SUM}}(S,d) = (k-1) \sum_{p_i \in S} rel(p_i) + 2\lambda \sum_{p_i, p_j \in S} d(p_i, p_j) \tag{5.3}$$

In Poikilo, users can select how to combine relevance with diversity and specify the value of related tuning parameters.

Concerning the DisC model, we define the Weighted r-DisC Diverse Subset Problem:

Definition 5.3. (Minimum Weighted r-DisC Diverse Subset) Given a set $\mathcal{P}$ of items with each object $p_i \in \mathcal{P}$ associated with a weight $w(p_i)$ 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 $\sum_{p_i \in S^*} \frac{1}{w(p_i)} \leq \sum_{p_i \in S} \frac{1}{w(p_i)}$.

Figure 5.6 reports solutions for the same dataset and radius when relevance is considered or not. Again, we provide implementations of many different algorithms for handling relevance.

5.3.2 Streaming data

We also consider the dynamic case in which items change over time, as for example, in the case of notification services. We adopt a sliding-window model where diverse items are computed over sliding windows of length w in the input data. The length of the window w can be defined either in time units (e.g., "the most diverse items in the last hour") or in number of items (e.g., "the most diverse items among the 100 most recent ones").

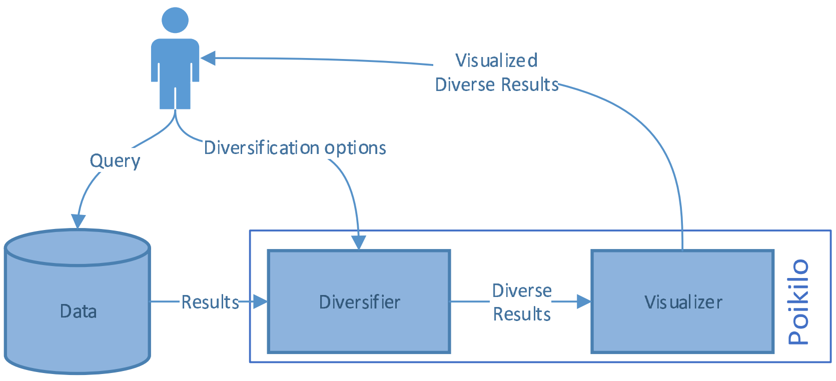


Figure 5.7: POIKILO system architecture.

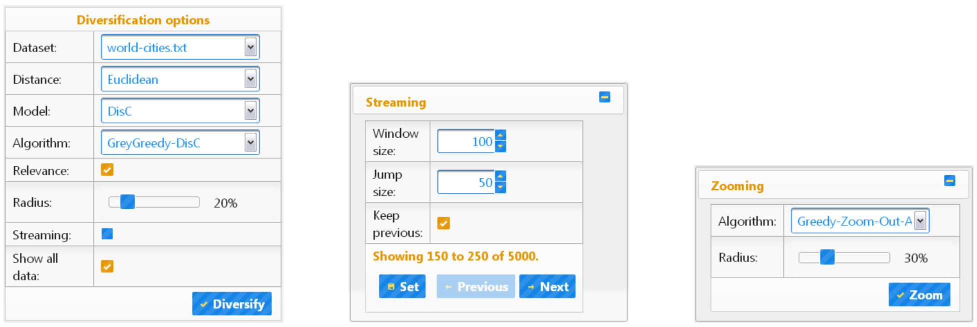
We have implemented the index-based algorithms proposed in [39, 45], using Cover Trees to dynamically update the diverse subset of each window. We also provide the option to enforce the continuity properties proposed in [39, 45] among consequent windows. For example, the order in which the diverse items are delivered to the users should follow the order of their generation. Also, an item should not appear, disappear and then re-appear in the diverse set.

5.4 System Description

POIKILO is a Web Application implemented in Java EE using JavaServer Faces 2.0. POIKILO can be accessed via a simple web browser using an intuitive GUI (Figure 5.1). The system architecture can be seen in Figure 5.7. During the demonstration, users will be allowed to submit queries to a number of different datasets, see diverse results and tune a variety of diversification parameters.

We provide a number of datasets, both real and synthetic. Our synthetic datasets consist of points in the 2D plane. Points are either uniformly distributed in space or form clusters of different sizes. Relevance scores are also assigned to items in a uniform or clustered way. We also use a number of real datasets, such as two spatial datasets containing geographic information about the location of (i) 5922 cities and villages in Greece [5], (ii) apartments in various cities (London, Paris etc.) collected from [6] and also a dataset consisting of images of people posing with different facial expressions [2]. Users can also upload their own datasets to the system via the GUI.

Upon entering the system, users are presented with a panel providing a wide variety of different diversification options (Figure 5.8). First, they select a dataset along with a distance metric (e.g., Euclidean, cosine, Harversine) and a diversification model (e.g., DisC, MAXMIN, MAXSUM). Then, according to the selected model, a number of algorithms and options become available to them. For example, they can select a diversification algorithm (e.g., *Basic-DisC* or *Best-Interchange*) and algorithm-specific parameters (e.g., r, k). Also, they can choose whether to also account for relevance or not during the



Diversification options

Dataset:	world-cities.txt
Distance:	Euclidean
Model:	DisC
Algorithm:	GreyGreedy-DisC
Relevance:	✓
Radius:	20%
Streaming:	☐
Show all data:	✓

Streaming

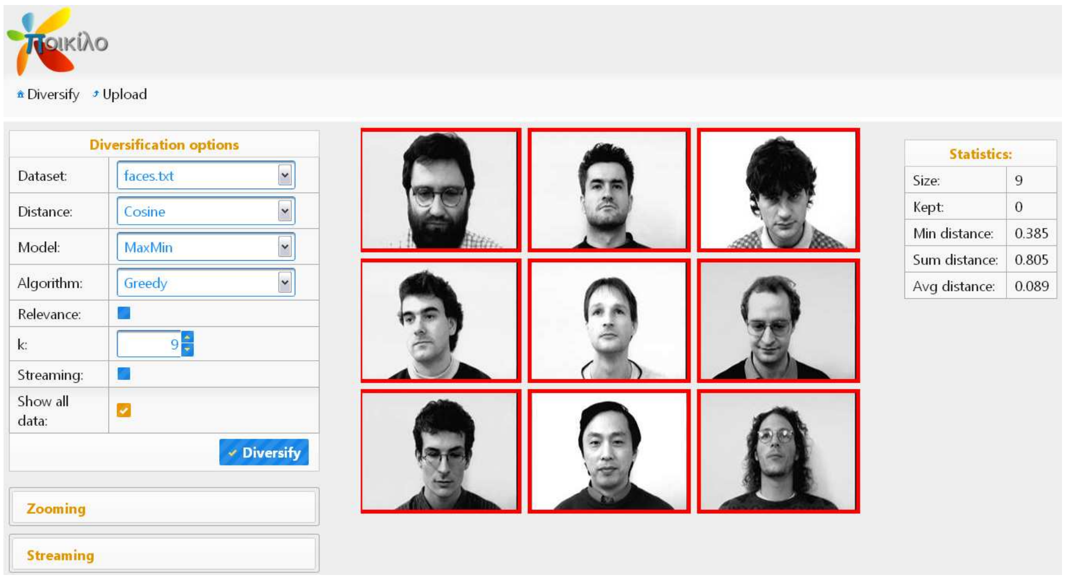
Window size:	100
Jump size:	50
Keep previous:	✓

Showing 150 to 250 of 5000.

Zooming

Algorithm:	Greedy-Zoom-Out-A
Radius:	30%

Figure 5.8: Selecting diversification parameters.



Diversification options

Dataset:	faces.txt
Distance:	Cosine
Model:	MaxMin
Algorithm:	Greedy
Relevance:	☐
k:	9
Streaming:	☐
Show all data:	✓

Statistics:

Size:	9
Kept:	0
Min distance:	0.385
Sum distance:	0.805
Avg distance:	0.089

Figure 5.9: Diverse results for our images dataset.

selection of representative results and also whether to treat the input data as streaming by specifying a window length.

The computed diverse subset is presented to the users along with additional information, such as the size of the diverse subset and the average pairwise distance among the selected items. For point data, a visualization of the whole dataset is presented, in which diverse items are represented in a different size and color (Figure 5.1). If relevance is considered, the size of each diverse item corresponds to its relevance score, i.e., the larger this score is, the larger the item appears. Users have the option to hide the non-diverse items if they wish. For image data, the diverse set of images is presented to the user (Figure 5.9).

When the DisC model is employed, after being presented with the diverse subset, users have the option to tune the degree of diversification by zooming-in or zooming-out of the presented subset. A sliding bar is provided, which users can slide to dynamically increase or decrease the value of r without having to specify it explicitly.

Finally, when users use the streaming option, they have the opportunity to see how diverse items change as items enter and leave the current window by navigating between windows via “next” and “previous” buttons. Users can also request the enforcement of continuity properties among consequent windows.

5.5 Summary

In this chapter, we presented Poikilo, a tool to assist users in locating and evaluating diverse results. We provide implementations of a wide suite of models and algorithms found in the related literature to compute and compare diverse results. Users can tune various diversification parameters, combine diversity with relevance and also see how diverse results change over time in the case of streaming data. When the diversification model allows it, they can also zoom-in and zoom-out of the diversified results.

Chapter 6

Conclusions and Future Work

The purpose of this thesis was the development, implementation and evaluation of models, algorithms and techniques for ranking information based on both *relevance* to user information needs and *diversity*. Next, in Section 6.1, we summarize our contributions along these axes while, in Section 6.2, we describe directions for future work.

6.1 Summary of Contributions

During the elaboration of this thesis, we mainly focused on two interesting aspects of diversification. First, we considered the problem of diversifying dynamic data based on a traditional, widely used definition of diversity, namely MaxMin. Second, we introduced a novel definition of diversity, called DisC diversity, which is based on item dissimilarity and coverage. Generally, given a positive real number r, which we call radius, two items are dissimilar to each other when their distance is larger than r, while they cover each other when their distance is smaller than or equal to r. We also considered the problem of incrementally adapting a DisC diverse subset to a new radius. Finally, we developed a system prototype, called Poikilo, for evaluating the results of various diversification models and algorithms. Next, we briefly summarize the contributions of this thesis.

Despite the considerable recent interest in diversification, most previous research considers the static version of the problem. This means that the available items out of which a diverse subset is selected do not change over time. Here, we focused on the dynamic MaxMin diversification problem, where insertions and deletions of items are allowed and the diverse subset needs to be refreshed to reflect such updates.

We proposed an index-based approach based on cover trees and introduced a suite of algorithms that exploit the cover tree to provide solutions with varying accuracy and complexity. We also provided theoretical results that bound the accuracy of the solutions achieved with regards to the optimal solution.

Concerning dynamic data diversification, our main contributions can be outlined as follows:

- We proposed a novel, index-based approach for solving the MaxMin diversification problem. We also introduced a number of continuity requirements for increasing the quality of diverse results in a streaming scenario.
- We showed that locating optimal solutions is an NP-hard problem and provided a number of cover tree based algorithms for locating approximate solutions. We presented an efficient implementation of a traditional greedy algorithm which provides an $1/2$-approximation of the optimal solution. We also introduced a new family of algorithms which provide a $b-1/2b^2$-approximation of the optimal solution, where b is the base of the cover tree.
- We extended our algorithms for selecting items that are both relevant and diverse. We considered two different approaches. The first one considers the relevance of the items when inserting them into the index, while the second one combines relevance with diversity when selecting items from the index.

We also addressed diversity through a different perspective, via defining *r-Dissimilar and covering* or *r-DisC* diverse subsets, based on some radius r. To ensure that all items are represented by at least one similar item in the diverse subset, we require all available items to be covered by at least one diverse item. We also require diverse items to be dissimilar to each other.

Instead of specifying a required size k of the diverse set or a threshold, as is the case with most other diversity definitions, our tuning parameter r explicitly expresses the degree of diversification and determines the size of the diverse set. To retrieve a concise representation of all items, among all DisC diverse subsets that answer the user query, we aim at selecting the one containing the smallest number of items. In case items are also associated with weights, we also take them into consideration when selecting our diverse items. Also, to allow different areas of the data to contribute more or less items to the selected diverse subset, we extended the definition of DisC diverse subsets to allow each item to be associated with a different radius.

We formalized the problem of locating minimum and minimum weighted DisC diverse subsets as an independent dominating set problem on graphs. We showed that locating minimum DisC diverse subsets is an NP-hard problem and provided a suite of algorithms for locating approximate solutions. We explored the relation among DisC diverse subsets of different radii and provided algorithms for incrementally adapting a DisC diverse subset to a new radius (zooming). We provided 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.

Concerning diversification based on dissimilarity and coverage, our main contributions can be outlined as follows:

- We introduced a novel, intuitive definition of diversity, called DisC diversity, based on using a radius r rather than a size limit k to select diverse items. We extended DisC diversity to the weighted case, in which items are associated with a weight indicating their importance, or relevance. We also considered the case in which different items are associated with different radii and defined two variations of DisC diversity.

- We exploited a graph-based view of the problem and showed that locating minimum DisC diverse subsets is an NP-hard problem.

- We provided a suite of algorithms for locating approximate solutions. We showed that our solutions are at most B times larger than any optimal solution, where B be the maximum number of independent neighbors of any item in our data. B depends on the dimensionality of the data and the employed distance metric. We derived the value of B for a number of different dimensionalities and distance metrics.

- We introduced incremental diversification to a new radius through zooming-in and zooming-out and studied the size relationship between the diverse subsets for the two radii. Again, we derived specific bounds for a number of different dimensionalities and distance metrics.

- We provided efficient implementations of our algorithms based on spatial index structures, namely the M-Tree.

- We provided a thorough qualitative comparison of the various DisC variations and also compared DisC diversity with other popular diversity models, both analytically and qualitatively.

Finally, we also developed a system prototype, called Poikilo, which is a system designed to assist users in locating, visualizing and comparing diverse results based on a suite of different diversification models and algorithms. We provided implementations of a wide variety of diversification approaches found in the related literature for retrieving diverse results. Users of Poikilo can submit queries to a number of different datasets and see a visualization of a diversified subset of their query result. They can choose among a wide selection of diversification algorithms and specify various configuration parameters. Furthermore, they can also zoom-in and zoom-out of this initial diverse subset and navigate between consequent windows in the case of streaming data.

6.2 Directions for Future Work

Next, we offer some insights on a number of open issues related to this thesis that are the subject of our on-going and future work. We make a distinction between short term plans, that consist of extensions to work done during the elaboration of this thesis, and long term plans, that outline ideas for future research related to our work.

Short term plans

Diversification has a wide range of applications, ranging from database and web search to notification systems and recommenders. We next briefly describe our previous work on *database exploration*, *multiple search results diversification* and *keyword search* and present future work ideas for these lines of research.

Diversification in Database Exploration. Users usually interact with databases by formulating queries. This query-response mode of interaction assumes that users are to some extent familiar with the content of the database and, also, that they have a clear understanding of their information needs. However, as the volume of information becomes larger and accessible to a more diverse and less technically-oriented audience, a more exploratory/recommendation-based mode of information seeking seems relevant and useful.

Previous approaches for assisting users in querying a database mainly focus on query rewriting for retrieving more or less results, for example, by adding constraints to the query (e.g., [95]) or automatically ranking query results and presenting to users only the top-k most highly ranked among them (e.g., [30]). With *facet search* (e.g., [90, 66, 54]), users start with a general query and progressively narrow its results down to a specific item by specifying at each step restrictions on attribute values.

In our previous work [40, 43, 97], we introduced a novel exploratory mode of database interaction that allows users to discover items that, although not part of the result of their original query, are highly correlated to this result. For example, assume a user asking about movies by a specific director, e.g., M. Scorsese. We want to highlight interesting aspects of these results, e.g., interesting years, production countries, pairs of genre and years etc. To do this, at first, *interesting* parts of the result of the initial user query are identified. These are sets of (attribute, value) pairs, called *faSets*, that are highly distinctive for the query. The *interestingness* of a faSet is based on its frequency, both in the query result and the database. Intuitively, the more frequent a faSet in the query result and the less frequent a faSet in the database, the more interesting it is for the query.

To avoid the costly on-line computation of the frequency of each faSet, in [40, 43], we proposed maintaining an appropriate summary that allows us to *estimate* such frequencies when needed. Our approach is based on storing the frequencies of a set of representative rare faSets which are then used to estimate the interestingness of the faSets that appear in the result of any given user query.

After the k most interesting faSets have been located for a specific user query, *exploratory queries* are constructed whose results possess these interesting faSets. The results of the exploratory queries, called YMAL ("You May Also Like") results, are also presented to the user. For example, by clicking on each important aspect of the query about movies by M. Scorsese, the user gets additional recommended YMAL results, i.e., other directors who have directed movies with characteristics similar to the selected ones. This way, users get to know other, possibly unknown to the them, directors who have directed movies similar to those of M. Scorsese in our example.

In some respect, exploratory queries may be seen as recommendations. Extending database queries with recommendations has been suggested in some recent works, namely [72] and [29, 12]. [72] proposes a general framework and a related engine for the declarative specification of the recommendation process. The recommendations in our case are of a very specific form. Recommendations in [29, 12] have the form of queries and are based on the relations they involve and the similarity of their structure to that of the original user query. Those recommendations are based on the past behavior of similar users, whereas we consider only the content of the database and the query result. The functionality we propose is complementary to query-response and recommendation systems. Contrary to facet search and related approaches, our goal is not to refine the original query so as to narrow its results. Instead, we provide users with items that do not belong to the results of their original query but are highly related to them. We do this based solely on the database content and the initial query and not based on any log of previous user queries or results.

However, the most interesting faSets may often be similar to each other, since interesting faSets are often sub-faSets of other interesting faSets. For this reason, diversification can greatly improve the quality of presented faSets. In our previous user studies [40], we saw that most users find larger faSets, i.e., faSets containing more (attribute, value) pairs, to be more interesting. We reckon that some coverage-based definition of diversity can be employed to select a concise subset of faSets by promoting faSets that cover more (attribute, value) pairs. Another interesting direction is to employ some notion of novelty to dynamically change the ordering of presented faSets after the user has clicked on one or more faSets and returned back to the list, to reflect the fact that the user has already explored some part of the database and did or did not find it to their liking.

Besides incorporating diversity, another interesting extension of this line of research is to consider using information from resources external to the database, such as the web, when forming exploratory queries. For example, consider that "Italy" is an interesting faSet when searching for movies of M. Scorsese. Then, besides retrieving other directors related to Italy, we could also adjust our exploratory query to retrieve further information about the country itself from some external source, such as Wikipedia [7].

Diversification of Multiple Search Results. The explosion of accessible data emphasizes the need for data diversification in a wide range of web, scientific, and business applications. The challenge of efficient diversification is further accentuated in a multi-user environment, in which multiple search queries are to be executed and diversified at the same time. The focus of previous research, however, is on diversifying the results of a single query. In our previous work [68, 69], we addressed the problem of scalable diversification of multiple search results. Our approach leverages the natural overlap in results of different queries for the concurrent diversification of those overlapping results. This allows us to reduce computational costs. Moreover, we extend our approach to exploit a spatial grid structure that allows us to further reduce computational costs while maintaining comparable quality of diversification.

A number of issues are open in this line of research. One of them concerns the efficient detection of overlapping results of different queries by exploiting spatial indices to quickly browse similar results. Another one concerns the trade-off of *accuracy* for *efficiency* in such multi-user environments. For example, consider two queries q_1 and q_2 and their corresponding result sets $\mathcal{P}_1$ and $\mathcal{P}_2$. Let q_1 and q_2 be very similar to each other. In this case, a diversified subset of $\mathcal{P}_1$ is probably a good answer for q_2 as well. However, there may be items in $\mathcal{P}_2$ that do not belong to $\mathcal{P}_1$ and, thus, are never retrieved for q_2, even if they are diverse to the rest of the returned items. However, users may be willing to disregard such items in exchange for quick result retrieval, as long as the overall quality of the retrieved results does not degrade considerably. Studying the relation between the similarity of two queries and their optimal diversified subsets is an interesting open problem.

Diversification of Keyword Search Results in Databases. In our previous work [98], we considered keyword-based search in relational databases. Keyword-based search allows users to discover relevant information without knowing the database schema or using complicated queries. However, such searches may return an overwhelming number of results, often loosely related to the user intent. We proposed ranking results based on both their relevance to the query and user preferences and presenting users with only the top-k most highly ranked results. We also considered diversifying the retrieved results based on content. To do this, first, the top-m most highly ranked results are retrieved, for $m \geq k$. Then, the k most diverse items are presented to the user. The larger the value of m, the larger the computational cost but, also, the larger the diversity of the retrieved results.

Although common in the related literature, computing first m results out of which the final k ones are retrieved induces extra computational overheads. An interesting direction for future research is moving diversification in the initial ranking phase. One approach is to use some mono-objective formulation of the problem. Also, a coverage-based definition of diversity may be better suited in this context, due to the distinctive form of results, which are in the form of *joining trees of tuples*, i.e., tuples from the database linked through foreign key constraints.

Long term plans

In our future work, we plan to focus on result diversification in a distributed setting, where the available items are not gathered in a single site but are rather distributed over a set of different sources.

Our motivation emanates from a number of reasons. First, nowadays, a large number of information sources are distributed over the network. This, in addition to the huge volume of accessible information, renders the transferring of all relevant information to a single site as a pre-processing step before computing diverse subsets impractical. Diversifying information at the source and later aggregating partial results seems a much more attractive approach. Second, computing diverse subsets has inherently high computational cost, since the diversity of a single item is not an individual property but rather depends on the selection of other items. Computing distances among all available items is the main factor for this high computational cost. Computing diverse subsets of partitions of data and later aggregating such partial results can, thus, also greatly decrease cost overheads. Finally, another reason for "pushing" diversification closer to the information sources is the possible exploitation of shared computations for diversifying items corresponding to results of different queries, issued by one or more users. This scenario finds many applications in modern notification services, in which users are both generators and consumers of information. Locating optimal nodes in such an overlay network of users to conduct result diversification can reduce both the computational as well as the communication costs in the network.

A great challenge of the distributed computation of diverse items is that the optimal solutions computed for different partitions of the data may be generally very different from the optimal solution for the whole data. However, in many applications, such as web search and recommenders, diverse items are often used as a first step towards presenting the user with a representative, concise overview of the underlying information. Users can then refine such diverse results to their liking. Therefore, *quickly* locating diverse subsets is of high importance. We believe that efficient algorithms for fast retrieval of approximate solutions, withing some acceptable bound, are of great practical interest.

BIBLIOGRAPHY

[1] Acme digital cameras database. http://acme.com/digicams.

[2] Faces dataset. http://www.informedia.cs.cmu.edu.

[3] Flickr. http://www.flickr.com.

[4] Flickr dataset. http://www.tagora-project.eu.

[5] Greek cities dataset. http://www.rtreeportal.org.

[6] Nestoria. http://www.nestoria.co.uk.

[7] Wikipedia. http://www.wikipedia.org.

[8] World cities dataset. http://nordpil.com/go/resources/world-database-of-large-cities.

[9] S. Abbar, S. Amer-Yahia, P. Indyk, and S. Mahabadi. Real-time recommendation of diverse related articles. In *WWW*, 2013.

[10] S. Abbar, S. Amer-Yahia, P. Indyk, S. Mahabadi, and K. R. Varadarajan. Diverse near neighbor problem. In *SoCG*, 2013.

[11] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *WSDM*, 2009.

[12] J. Akbarnejad, G. Chatzopoulou, M. Eirinaki, S. Koshy, S. Mittal, D. On, N. Polyzotis, and J. S. V. Varman. Sql querie recommendations. *PVLDB*, 3(2):1597–1600, 2010.

[13] E. Aktolga and J. Allan. Sentiment diversification with different biases. In *SIGIR*, 2013.

[14] P. Alimonti and T. Calamoneri. Improved approximations of independent dominating set in bounded degree graphs. In *WG*, 1996.

[15] A. Angel and N. Koudas. Efficient diversity-aware search. In *SIGMOD Conference*, 2011.

[16] A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. In *ICML*, 2006.

[17] R. Boim, T. Milo, and S. Novgorodov. Direc: Diversified recommendations for semantic-less collaborative filtering. In *ICDE*, 2011.

[18] R. Boim, T. Milo, and S. Novgorodov. Diversification and refinement in collaborative filtering recommender. In *CIKM*, 2011.

[19] A. Borodin, H. C. Lee, and Y. Ye. Max-sum diversification, monotone submodular functions and dynamic updates. In *PODS*, 2012.

[20] N. Bourgeois, F. D. Croce, B. Escoffier, and V. T. Paschos. Fast algorithms for min independent dominating set. *Discrete Applied Mathematics*, 161(4-5):558–572, 2013.

[21] A. Bozzon, M. Brambilla, P. Fraternali, and M. Tagliasacchi. Diversification for multi-domain result sets. In *ICWE*, 2012.

[22] G. Capannini, F. M. Nardini, R. Perego, and F. Silvestri. Efficient diversification of web search results. *PVLDB*, 4(7):451–459, 2011.

[23] J. G. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, 1998.

[24] I. Catallo, E. Ciceri, P. Fraternali, D. Martinenghi, and M. Tagliasacchi. Top-k diversity queries over bounded regions. *ACM Trans. Database Syst.*, 38(2):10, 2013.

[25] P. Chandar and B. Carterette. Preference based evaluation measures for novelty and diversity. In *SIGIR*, 2013.

[26] B. Chandra and M. M. Halldórsson. Facility dispersion and remote subgraphs. In *SWAT*, 1996.

[27] O. Chapelle, S. Ji, C. Liao, E. Velipasaoglu, L. Lai, and S.-L. Wu. Intent-based diversification of web search results: metrics and algorithms. *Inf. Retr.*, 14(6):572–592, 2011.

[28] G. Chartrand, F. Harary, and B. Q. Yue. On the out-domination and in-domination numbers of a digraph. *Discrete Mathematics*, 197-198:179–183, 1999.

[29] G. Chatzopoulou, M. Eirinaki, and N. Polyzotis. Query recommendations for interactive database exploration. In *SSDBM*, 2009.

[30] S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum. Probabilistic information retrieval approach for ranking of database query results. *ACM Trans. Database Syst.*, 31(3), 2006.

[31] H. Chen and D. R. Karger. Less is more: probabilistic models for retrieving fewer relevant documents. In *SIGIR*, 2006.

[32] M. Chlebík and J. Chlebíková. Approximation hardness of dominating set problems in bounded degree graphs. *Inf. Comput.*, 206(11):1264–1275, 2008.

[33] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *VLDB*, pages 426–435, 1997.

[34] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990.

[35] C. L. A. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *SIGIR*, 2008.

[36] G. D. da Fonseca, C. M. H. de Figueiredo, V. G. P. de Sá, and R. Machado. Linear time approximation for dominating sets and independent dominating sets in unit disk graphs. *CoRR*, abs/1204.3488, 2012.

[37] V. Dang and W. B. Croft. Diversity by proportionality: an election-based approach to search result diversification. In *SIGIR*, 2012.

[38] T. Deng and W. Fan. On the complexity of query result diversification. *PVLDB*, 6(8):577–588, 2013.

[39] M. Drosou and E. Pitoura. Diverse set selection over dynamic data. *IEEE Trans. Knowl. Data Eng. (to appear)*.

[40] M. Drosou and E. Pitoura. Ymaldb: Exploring relational databases via result-driven recommendations. *VLDBJ (to appear)*.

[41] M. Drosou and E. Pitoura. Diversity over continuous data. *IEEE Data Eng. Bull.*, 32(4):49–56, 2009.

[42] M. Drosou and E. Pitoura. Search result diversification. *SIGMOD Record*, 39(1):41–47, 2010.

[43] M. Drosou and E. Pitoura. Redrive: result-driven database exploration through recommendations. In *CIKM*, 2011.

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

[45] M. Drosou and E. Pitoura. Dynamic diversification of continuous data. In *EDBT*, 2012.

[46] M. Drosou, K. Stefanidis, and E. Pitoura. Preference-aware publish/subscribe delivery with diversity. In *DEBS*, 2009.

[47] E. Erkut. The discrete p-dispersion problem. *European Journal of Operational Research*, 46(1), 1990.

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

[49] R. Fagin. Fuzzy queries in multimedia database systems. In *PODS*, 1998.

[50] R. Fagin. Combining fuzzy information from multiple systems. *J. Comput. Syst. Sci.*, 58(1):83–99, 1999.

[51] P. Fraternali, D. Martinenghi, and M. Tagliasacchi. Top-k bounded diversification. In *SIGMOD Conference*, 2012.

[52] D. Gamarnik, D. Goldberg, and T. Weber. Ptas for maximum weight independent set problem with random weights in bounded degree graphs. In *SODA*, 2010.

[53] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[54] S. Garg, K. Ramamritham, and S. Chakrabarti. Web-cam: Monitoring the dynamic web to respond to continual queries. In *SIGMOD*, 2004.

[55] M. Gibson and I. A. Pirwani. Approximation algorithms for dominating set in disk graphs. *CoRR*, abs/1004.3320, 2010.

[56] W. Goddard and J. Lyle. Independent dominating sets in triangle-free graphs. *J. Comb. Optim.*, 23(1):9–20, 2012.

[57] S. Gollapudi and A. Sharma. An axiomatic approach for result diversification. In *WWW*, 2009.

[58] M. M. Halldórsson. Approximating the minimum maximal independence number. *Inf. Process. Lett.*, 46(4):169–172, 1993.

[59] J. R. Haritsa. The kndn problem: A quest for unity in diversity. *IEEE Data Eng. Bull.*, 32(4):15–22, 2009.

[60] M. Hasan, A. Mueen, V. J. Tsotras, and E. J. Keogh. Diversifying query results on semi-structured data. In *CIKM*, 2012.

[61] R. Hassin, S. Rubinstein, and A. Tamir. Approximation algorithms for maximum dispersion. *Oper. Res. Lett.*, 21(3), 1997.

[62] A. Jain, P. Sarda, and J. R. Haritsa. Providing diversity in k-nearest neighbor query results. In *PAKDD*, 2004.

[63] C. T. Jr., A. J. M. Traina, C. Faloutsos, and B. Seeger. Fast indexing and visualization of metric data sets using slim-trees. *IEEE Trans. Knowl. Data Eng.*, 14(2):244–260, 2002.

[64] M. Kacimi and J. Gamper. Diversifying search results of controversial queries. In *CIKM*, 2011.

[65] A. Kako, T. Ono, T. Hirata, and M. M. Halldórsson. Approximation algorithms for the weighted independent set problem in sparse graphs. *Discrete Applied Mathematics*, 157(4):617–626, 2009.

[66] A. Kashyap, V. Hristidis, and M. Petropoulos. Facetor: cost-driven exploration of faceted query results. In *CIKM*, 2010.

[67] M. Keikha, F. Crestani, and W. B. Croft. Diversity in blog feed retrieval. In *CIKM*, 2012.

[68] H. A. Khan, M. Drosou, and M. A. Sharaf. Dos: an efficient scheme for the diversification of multiple search results. In *SSDBM*, 2013.

[69] H. A. Khan, M. Drosou, and M. A. Sharaf. Scalable diversification of multiple search results. *CIKM*, 2013 (to appear).

[70] E. M. Knorr and R. T. Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB*, 1998.

[71] T. Kollar. Fast Nearest Neighbors. http://nicksgroup.csail.mit.edu/TK/Technical_Reports/covertrees.pdf.

[72] G. Koutrika, B. Bercovitz, and H. Garcia-Molina. Flexrecs: expressing and combining flexible recommendations. In *SIGMOD*, 2009.

[73] G. Koutrika and Y. E. Ioannidis. Personalized queries under a generalized preference model. In *ICDE*, 2005.

[74] O. Küçüktunç, E. Saule, K. Kaya, and Ü. V. Çatalyürek. Diversified recommendation on graphs: pitfalls, measures, and algorithms. In *WWW*, 2013.

[75] N. Lathia, S. Hailes, L. Capra, and X. Amatriain. Temporal diversity in recommender systems. In *SIGIR*, 2010.

[76] L. Li and C.-Y. Chan. Efficient indexing for diverse query results. *PVLDB*, 6(9):745–756, 2013.

[77] R. Li, B. Kao, B. Bi, R. Cheng, and E. Lo. Dqr: a probabilistic approach to diversified query recommendation. In *CIKM*, 2012.

[78] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang. Selecting stars: The k most representative skyline operator. In *ICDE*, 2007.

[79] B. Liu and H. V. Jagadish. Using trees to depict a forest. *PVLDB*, 2(1):133–144, 2009.

[80] K. Liu, E. Terzi, and T. Grandison. Highlighting diverse concepts in documents. In *SDM*, 2009.

[81] Z. Liu and Y. Chen. Differentiating search results on structured data. *ACM Trans. Database Syst.*, 37(1):4, 2012.

[82] Z. Liu, P. Sun, and Y. Chen. Structured search result differentiation. *PVLDB*, 2(1), 2009.

[83] E. C. Milner and R. E. Woodrow. On directed graphs with an independent covering set. *Graphs and Combinatorics*, 5(1):363–369, 1989.

[84] E. Minack, W. Siberski, and W. Nejdl. Incremental diversification for very large sets: a streaming-based approach. In *SIGIR*, 2011.

[85] C. Pang, R. Zhang, Q. Zhang, and J. Wang. Dominating sets in directed graphs. *Inf. Sci.*, 180(19):3647–3652, 2010.

[86] D. Panigrahi, A. D. Sarma, G. Aggarwal, and A. Tomkins. Online selection of diverse results. In *WSDM*, 2012.

[87] L. Qin, J. X. Yu, and L. Chang. Diversifying top-k results. *PVLDB*, 5(11):1124–1135, 2012.

[88] F. Radlinski and S. T. Dumais. Improving personalized web search using result diversification. In *SIGIR*, 2006.

[89] S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi. Facility dispersion problems: Heuristics and special cases. In F. K. H. A. Dehne, J.-R. Sack, and N. Santoro, editors, *WADS*, volume 519 of *Lecture Notes in Computer Science*. Springer, 1991.

[90] S. B. Roy, H. Wang, G. Das, U. Nambiar, and M. K. Mohania. Minimum-effort driven dynamic faceted search in structured databases. In *CIKM*, 2008.

[91] D. J. R. S. S. Ravi and G. K. Tayi. Heuristic and special case algorithms for dispersion problems. *Operations Research*, 42(2):299–310, 1994.

[92] L. F. D. Santos, W. D. Oliveira, M. R. P. Ferreira, A. J. M. Traina, and C. T. Jr. Parameter-free and domain-independent similarity search with diversity. In *SSDBM*, 2013.

[93] R. L. T. Santos, C. Macdonald, and I. Ounis. Selectively diversifying web search results. In *CIKM*, 2010.

[94] R. L. T. Santos, C. Macdonald, and I. Ounis. Intent-aware search result diversification. In *SIGIR*, 2011.

[95] N. Sarkas, N. Bansal, G. Das, and N. Koudas. Measure-driven keyword-query expansion. *PVLDB*, 2(1), 2009.

[96] D. Souravlias, M. Drosou, K. Stefanidis, and E. Pitoura. On novelty in publish/-subscribe delivery. In *ICDE Workshops*, 2010.

[97] K. Stefanidis, M. Drosou, and E. Pitoura. "you may also like" results in relational databases. In *PersDB*, 2009.

[98] K. Stefanidis, M. Drosou, and E. Pitoura. Perk: personalized keyword search in relational databases through preferences. In *EDBT*, 2010.

[99] I. Szpektor, Y. Maarek, and D. Pelleg. When relevance is not enough: promoting diversity and freshnessin personalized question recommendation. In *WWW*, 2013.

[100] A. Tamir. Obnoxious facility location on graphs. *SIAM J. Discrete Math.*, 4(4):550–567, 1991.

[101] Y. Tao, L. Ding, X. Lin, and J. Pei. Distance-based representative skyline. In *ICDE*, 2009.

[102] M. T. Thai, F. Wang, D. Liu, S. Zhu, and D.-Z. Du. Connected dominating sets in wireless networks with different transmission ranges. *IEEE Trans. Mob. Comput.*, 6(7):721–730, 2007.

[103] M. T. Thai, N. Zhang, R. Tiwari, and X. Xu. On approximation algorithms of k-connected m-dominating sets in disk graphs. *Theor. Comput. Sci.*, 385(1-3):49–59, 2007.

[104] P. Tsaparas, A. Ntoulas, and E. Terzi. Selecting a comprehensive set of reviews. In *KDD*, 2011.

[105] G. Valkanas, A. N. Papadopoulos, and D. Gunopulos. Skydiver: a framework for skyline diversification. In *EDBT*, 2013.

[106] D. Vallet and P. Castells. Personalized diversification of search results. In *SIGIR*, 2012.

[107] E. Vee, U. Srivastava, J. Shanmugasundaram, P. Bhat, and S. Amer-Yahia. Efficient computation of diverse query results. In *ICDE*, 2008.

[108] M. R. Vieira, H. L. Razente, M. C. N. Barioni, M. Hadjieleftheriou, D. Srivastava, C. T. Jr., and V. J. Tsotras. Divdb: A system for diversifying query results. *PVLDB*, 4(12):1395–1398, 2011.

[109] M. R. Vieira, H. L. Razente, M. C. N. Barioni, M. Hadjieleftheriou, D. Srivastava, C. T. Jr., and V. J. Tsotras. On query result diversification. In *ICDE*, 2011.

[110] M. J. Welch, J. Cho, and C. Olston. Search result diversity for informational queries. In *WWW*, 2011.

[111] I. O. Wilhelm. Packing triangles on a sphere. In http://www.math.uchicago.edu/~may/VIGRE/VIGRE2008/REUPapers/Wilhelm.pdf, 2008.

[112] K. Xing, W. Cheng, E. K. Park, and S. Rotenstreich. Distributed connected dominating set construction in geometric k-disk graphs. In *ICDCS*, 2008.

[113] C. Yu, L. V. S. Lakshmanan, and S. Amer-Yahia. It takes variety to make a world: diversification in recommender systems. In *EDBT*, 2009.

[114] M. Zhang and N. Hurley. Avoiding monotony: improving the diversity of recommendation lists. In *RecSys*, 2008.

[115] Y. Zhang, J. P. Callan, and T. P. Minka. Novelty and redundancy detection in adaptive filtering. In *SIGIR*, 2002.

[116] G. Zhao, M.-L. Lee, W. Hsu, and W. Chen. Increasing temporal diversity with purchase intervals. In *SIGIR*, 2012.

[117] W. Zheng, H. Fang, and C. Yao. Exploiting concept hierarchy for result diversification. In *CIKM*, 2012.

[118] X. Zhu, J. Guo, X. Cheng, P. Du, and H. Shen. A unified framework for recommending diverse and relevant queries. In *WWW*, 2011.

[119] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *WWW*, 2005.

[120] F. Zou, Y. Wang, X. Xu, X. Li, H. Du, P.-J. Wan, and W. Wu. New approximations for minimum-weighted dominating sets and minimum-weighted connected dominating sets on unit disk graphs. *Theor. Comput. Sci.*, 412(3):198–208, 2011.

AUTHOR'S PUBLICATIONS

Journal Publications

J1. Marina Drosou and Evaggelia Pitoura, *YmalDB: Exploring Relational Databases via Result-Driven Recommendations*, in VLDBJ (to appear)

J2. Marina Drosou and Evaggelia Pitoura, *Diverse Set Selection over Dynamic Data*, in IEEE TKDE (to appear)

J3. Marina Drosou and Evaggelia Pitoura, *DisC Diversity: Result Diversification based on Dissimilarity and Coverage*, in PVLDB, vol. 6, no.1, pp. 13-24, 2012, VLDB Endowment (**Best Paper Award**)

J4. Marina Drosou and Evaggelia Pitoura, *Search Result Diversification*, in SIGMOD Record, vol. 39, no. 1, pp. 41-47, 2010, ACM

J5. Marina Drosou and Evaggelia Pitoura, *Diversity over Continuous Data, in IEEE Data Engineering Bulletin*, vol. 32, no. 4, pp. 49-56, 2009, IEEE

Conference Publications

C1. Hina A. Khan, Marina Drosou and Mohamed A. Sharaf, *Scalable Diversification of Multiple Search Results*, in Proc. of the 22nd International Conference on Information and Knowledge Management (CIKM 2013), October 27-November 1, 2013, Burlingame, Califormnia, USA

C2. Hina A. Khan, Marina Drosou and Mohamed A. Sharaf, *DoS: An Efficient Scheme for the Diversification of Multiple Search Results*, in Proc. of the 25th International Conference on Scientific and Statistical Database Management (SSDBM 2013), July 29-31, 2013, Baltimore, Maryland, USA

C3. Marina Drosou and Evaggelia Pitoura, *Dynamic Diversification of Continuous Data*, in Proc. of the 15th International Conference on Extending Database Technology (EDBT 2012), March 27-30, 2012, Berlin, Germany

C4. Marina Drosou and Evaggelia Pitoura, *ReDrive: Result-Driven Database Exploration through Recommendations*, in Proc. of the 20th ACM Conference on Information and Knowledge Management (CIKM 2011), October 24-28, 2011, Glasgow, Scotland, UK

C5. Kostas Stefanidis, Marina Drosou and Evaggelia Pitoura, *PerK: Personalized Keyword Search in Relational Databases through Preferences*, in Proc. of the 13th International Conference on Extending Database Technology (EDBT 2010), March 22-26, 2010, Lausanne, Switzerland

C6. Marina Drosou, Kostas Stefanidis and Evaggelia Pitoura, *Preference-Aware Publish/Subscribe Delivery with Diversity*, in Proc. of the 3rd ACM International Conference on Distributed Event-Based Systems (DEBS 2009), July 6-9, 2009, Nashville, Tennessee, USA

C7. Marina Drosou, *XML Summaries for Routing in P2P systems*, in Proc. of the 1st Panhellenic Scientific Student Conference in Informatics, Computer Engineering and Related Technologies (EUREKA 2007), May 18-20, 2007, Patras, Greece

Workshops

W1. Dimitris Souravlias, Marina Drosou, Kostas Stefanidis and Evaggelia Pitoura, *On Novelty in Publish/Subscribe Delivery*, in Proc. of the 4th International Workshop on Ranking in Databases (DBRank 2010), in conjunction with the ICDE 2010 Conference, March 1, 2010, Long Beach, California, USA

W2. Kostas Stefanidis, Marina Drosou and Evaggelia Pitoura, *"You May Also Like" Results in Relational Databases*, in Proc. of the 3rd International Workshop on Personalized Access, Profile Management and Context Awareness: Databases (PersDB 2009), in conjunction with the VLDB 2009 Conference, August 28, 2009, Lyon, France

W3. Marina Drosou, *Ranked Publish/Subscribe Delivery*, PhD Workshop, in conjunction with the DEBS 2009 Conference, July 6, 2009, Nashville, Tennessee, USA

W4. Marina Drosou, Evaggelia Pitoura and Kostas Stefanidis, *Preferential Publish/Subscribe*, in Proc. of the 2nd International Workshop on Personalized Access, Profile Management and Context Awareness: Databases (PersDB 2008), in conjunction with the VLDB 2008 Conference, August 23, 2008, Auckland, New Zealand

Demos

D1. Marina Drosou and Evaggelia Pitoura, *POIKILO: A Tool for Evaluating the Results of Diversification Models and Algorithms*, 39th International Conference on Very Large Data Bases (VLDB 2013), August 26-30, 2013, Riva del Garda, Trento, Italy

D2. Marina Drosou and Evaggelia Pitoura, *YmalDB: A Result-Driven Recommendation System for Databases*, 16th International Conference on Extending Database Technology (EDBT 2013), March 18-22, 2013, Genoa, Italy

D3. Eftychia Koletsou, Marina Drosou, Kostas Stefanidis and Evaggelia Pitoura, *YMAL: A recommendation System for Relational Databases*, 9th Hellenic Data Management Symposium (HDMS 2010), June 28 - July 3, 2010, Ayia Napa, Cyprus

Short Vita

Marina Drosou was born in Ioannina, Greece in 1984. She received her BSc and MSc degrees from the Computer Science Department of the University of Ioannina in 2006 and 2008 respectively. She has been a member of the Distributed Management of Data Laboratory since 2006. Her research interests include personalization via ranking based on diversity and recommendation systems. In the past, she has worked as an Instructor at the Department of Informatics and Telecommunications of the Technological Educational Institute of Epirus, Greece, and was also a visiting trainee researcher at the School of Information Technology and Electrical Engineering of the University of Queensland, Australia. She is currently a software engineer at the Information Systems Division of the Hellenic Police Headquarters. She has received scholarships from the Bodossaki Foundation and the Greek state ("Heracleitus II"). She is the recipient of the VLDB 2013 Best Paper Award. Her work has been published in VLDBJ, TKDE, PVLDB, EDBT etc.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ
2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Η παρούσα έρευνα έχει συγχρηματοδοτηθεί από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο - ΕΚΤ) και από εθνικούς πόρους μέσω του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» του Εθνικού Στρατηγικού Πλαισίου Αναφοράς (ΕΣΠΑ) - Ερευνητικό Χρηματοδοτούμενο Έργο: Ηράκλειτος ΙΙ. Επένδυση στην κοινωνία της γνώσης μέσω του Ευρωπαϊκού Κοινωνικού Ταμείου.