Searching and Mining in Enriched Geo-Spatial Data

Klaus Arthur Schmid

München 2016

Searching and Mining in Enriched Geo-Spatial Data

Klaus Arthur Schmid

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

vorgelegt von
Klaus Arthur Schmid
aus München

München, den 30.09.2016

Erstgutachter: Prof. Dr. Matthias Renz
Zweitgutachter: Prof. Dr. Christian S. Jensen
Mündliche Prüfung: 9. Dezember 2016

Eidesstattliche Versicherung

Hiermit erkläre ich, Klaus Arthur Schmid, an Eides statt, dass die vorliegende Dissertation ohne unerlaubte Hilfe gemäß Promotionsordnung vom 12. Juli 2011, §8, Abs. 2 Pkt. 5, angefertigt worden ist. An den enthaltenen wissenschaftlichen Arbeiten wurde außer durch die genannten Koautoren keine fremde Mithilfe geleistet.

München, den 30.09.2016

Contents

List of Figures

List of Tables

Abstract

The emergence of new data collection mechanisms in geo-spatial applications paired with a heightened tendency of users to volunteer information provides an ever-increasing flow of data of high volume, complex nature, and often associated with inherent uncertainty. Such mechanisms include crowdsourcing, automated knowledge inference, tracking, and social media data repositories. Such data bearing additional information from multiple sources like probability distributions, text or numerical attributes, social context, or multimedia content can be called multi-enriched. Searching and mining this abundance of information holds many challenges, if all of the data's potential is to be released.

This thesis addresses several major issues arising in that field, namely path queries using multi-enriched data, trend mining in social media data, and handling uncertainty in geo-spatial data. In all cases, the developed methods have made significant contributions and have appeared in or were accepted into various renowned international peer-reviewed venues.

A common use of geo-spatial data is path queries in road networks where traditional methods optimise results based on absolute and ofttimes singular metrics, i.e., finding the shortest paths based on distance or the best trade-off between distance and travel time. Integrating additional aspects like qualitative or social data by enriching the data model with knowledge derived from sources as mentioned above allows for queries that can be issued to fit a broader scope of needs or preferences. This thesis presents two implementations of incorporating multi-enriched data into road networks. In one case, a range of qualitative data sources is evaluated to gain knowledge about user preferences which is subsequently matched with locations represented in a road network and integrated into its components. Several methods are presented for highly customisable path queries that incorporate a wide spectrum of data. In a second case, a framework is described for resource distribution with reappearance in road networks to serve one or more clients, resulting in paths that provide maximum gain based on a probabilistic evaluation of available resources. Applications for this include finding parking spots.

Social media trends are an emerging research area giving insight in user sentiment and important topics. Such trends consist of bursts of messages concerning a certain topic within a time frame, significantly deviating from the average appearance frequency of the same topic. By investigating the dissemination of such trends in space and time, this thesis presents methods to classify trend archetypes to predict future dissemination of a trend.

Processing and querying uncertain data is particularly demanding given the additional

knowledge required to yield results with probabilistic guarantees. Since such knowledge is not always available and queries are not easily scaled to larger datasets due to the #P-complete nature of the problem, many existing approaches reduce the data to a deterministic representation of its underlying model to eliminate uncertainty. However, data uncertainty can also provide valuable insight into the nature of the data that cannot be represented in a deterministic manner. This thesis presents techniques for clustering uncertain data as well as query processing, that take the additional information from uncertainty models into account while preserving scalability using a sampling-based approach, while previous approaches could only provide one of the two. The given solutions enable the application of various existing clustering techniques or query types to a framework that manages the uncertainty.

Zusammenfassung (Abstract in German)

Das Erscheinen neuer Methoden zur Datenerhebung in räumlichen Applikationen gepaart mit einer erhöhten Bereitschaft der Nutzer, Daten über sich preiszugeben, generiert einen stetig steigenden Fluss von Daten in großer Menge, komplexer Natur, und oft gepaart mit inhärenter Unsicherheit. Beispiele für solche Mechanismen sind Crowdsourcing, automatisierte Wissensinferenz, Tracking, und Daten aus sozialen Medien. Derartige Daten, angereichert mit mit zusätzlichen Informationen aus verschiedenen Quellen wie Wahrscheinlichkeitsverteilungen, Text- oder numerische Attribute, sozialem Kontext, oder Multimediainhalten, werden als multi-enriched bezeichnet. Suche und Datamining in dieser weiten Datenmenge hält viele Herausforderungen bereit, wenn das gesamte Potenzial der Daten genutzt werden soll.

Diese Arbeit geht auf mehrere große Fragestellungen in diesem Feld ein, insbesondere Pfadanfragen in multi-enriched Daten, Trend-mining in Daten aus sozialen Netzwerken, und die Beherrschung von Unsicherheit in räumlichen Daten. In all diesen Fällen haben die entwickelten Methoden signifikante Forschungsbeiträge geleistet und wurden veröffentlicht oder angenommen zu diversen renommierten internationalen, von Experten begutachteten Konferenzen und Journals.

Ein gängiges Anwendungsgebiet räumlicher Daten sind Pfadanfragen in Straßennetzwerken, wo traditionelle Methoden die Resultate anhand absoluter und oft auch singulärer Maße optimieren, d.h., der kürzeste Pfad in Bezug auf die Distanz oder der beste Kompromiss zwischen Distanz und Reisezeit. Durch die Integration zusätzlicher Aspekte wie qualitativer Daten oder Daten aus sozialen Netzwerken als Anreicherung des Datenmodells mit aus diesen Quellen abgeleitetem Wissen werden Anfragen möglich, die ein breiteres Spektrum an Anforderungen oder Präferenzen erfüllen. Diese Arbeit präsentiert zwei Ansätze, solche multi-enriched Daten in Straßennetze einzufügen. Zum einen wird eine Reihe qualitativer Datenquellen ausgewertet, um Wissen über Nutzerpräferenzen zu generieren, welches darauf mit Örtlichkeiten im Straßennetz abgeglichen und in das Netz integriert wird. Diverse Methoden werden präsentiert, die stark personalisierbare Pfadanfragen ermöglichen, die ein weites Spektrum an Daten mit einbeziehen. Im zweiten Fall wird ein Framework präsentiert, das eine Ressourcenverteilung im Straßennetzwerk modelliert, bei der einmal verbrauchte Ressourcen erneut auftauchen können. Resultierende Pfade ergeben einen maximalen Ertrag basieren auf einer probabilistischen Evaluation der

verfügbaren Ressourcen. Eine Anwendung ist die Suche nach Parkplätzen.

Trends in sozialen Medien sind ein entstehendes Forscchungsgebiet, das Einblicke in Benutzerverhalten und wichtige Themen zulässt. Solche Trends bestehen aus großen Mengen an Nachrichten zu einem bestimmten Thema innerhalb eines Zeitfensters, so dass die Auftrittsfrequenz signifikant über den durchschnittlichen Level liegt. Durch die Untersuchung der Fortpflanzung solcher Trends in Raum und Zeit präsentiert diese Arbeit Methoden, um Trends nach Archetypen zu klassifizieren und ihren zukünftigen Weg vorherzusagen.

Die Anfragebearbeitung und Datamining in unsicheren Daten ist besonders herausfordernd, insbesondere im Hinblick auf das notwendige Zusatzwissen, um Resultate mit probabilistischen Garantien zu erzielen. Solches Wissen ist nicht immer verfügbar und Anfragen lassen sich aufgrund der #P-Vollständigkeit des Problems nicht ohne Weiteres auf größere Datensätze skalieren. Dennoch kann Datenunsicherheit wertvollen Einblick in die Struktur der Daten liefern, der mit deterministischen Methoden nicht erreichbar wäre. Diese Arbeit präsentiert Techniken zum Clustering unsicherer Daten sowie zur Anfragebearbeitung, die die Zusatzinformation aus dem Unsicherheitsmodell in Betracht ziehen, jedoch gleichzeitig die Skalierbarkeit des Ansatzes auf große Datenmengen sicherstellen.

Part I

Preface

Chapter 1

Large Volumes of Complex Data

The amount of information available to humanity is ever increasing. With all knowledge accumulated by mankind by the year One CE as one unit of information, R. Buckminster Fuller estimated that doubling this knowledge took until the sixteenth century – with the ensuing growth at an exponential level, doubling again in 1750 and 1900 CE. [63] Digital data processing and spread of the internet brought about additional boosts to knowledge accumulation, as exchanging and storing information became cheaper than ever before. However, data was still largely collected and curated manually.

Only cheaply produced microelectronics on a mass market allowed for the creation and storage of data on a scale never seen before. The advent of the internet of things as well as the availability of smartphones, wearables, sensors, and RFID- and NFC-equipped devices brought the data generation rate to double every 12 hours, according to an IBM whitepaper [162]. Nonetheless, not all of that data equals knowledge usable by humans – devices integrated into products of everyday use such as mobile phones or on-board computers in cars as well as large-scale structures like oil rigs or airplanes produce and store lots of raw sensoric information that is incomprehensible without proper processing. Harnessing the full potential of this data is a promising trend in both industry and research: McKinsey [131] reported already in 2011 that using such "Big Data" to the full could generate savings exceeding €100 Billion for European government administrations through efficiency improvements, and generate the need for 1.5 Million additional managers and analysts with according know-how by 2018 in the U.S. alone. In academia the postulated Fourth Paradigm of scientific research [77] marks the trend to data-driven research.

Aside from automatically generated data, users of connected devices and the internet produce a tremendous amount of structured data. Business intelligence provider Domo estimates that in a typical *minute* of 2016, users issue around 14 million weather forecast queries to the Weather Channel, like 2.5 million posts on Instagram, share 400 hours worth of video on YouTube, swipe 970,000 profiles on Tinder, and stream 87,000 hours of video in Netflix. In the past five years, the global internet population has grown by 60%, and there exist more mobile devices on earth than people generating 18 Terabytes of mobile data every minute, in the US alone. [92] Ubiquity and availability of services make the amount of knowledge grow faster than ever before.

Chapter 2

Enriched Geo-Spatial Data

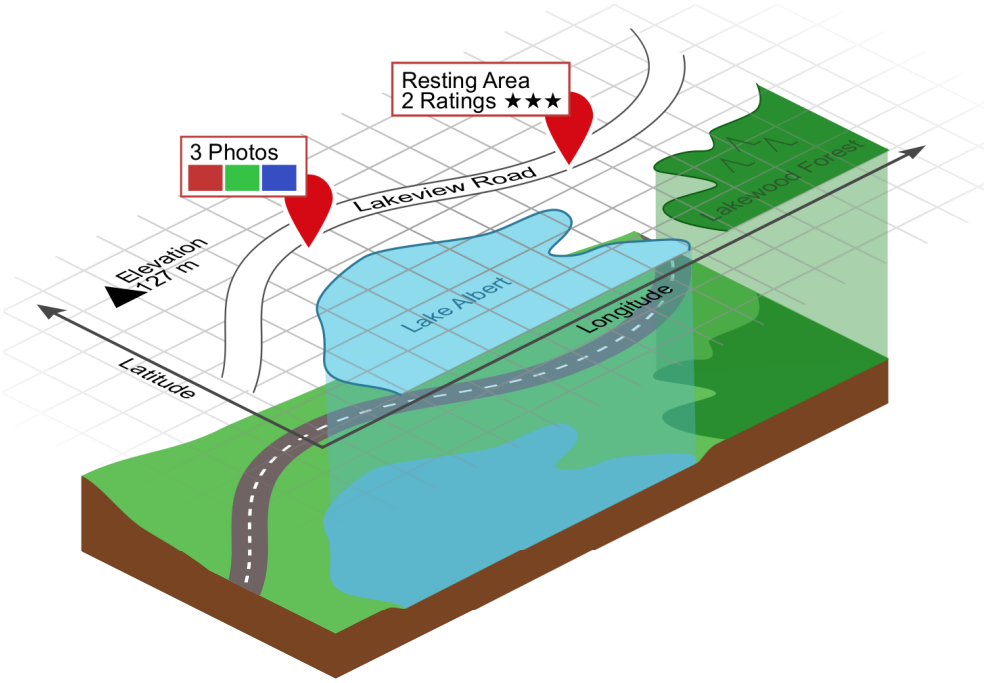


Figure 2.1: Geo-spatial data with enrichments

Most of the data described before involves some spatial domain referring to physical locations or areas on earth – be it the actual location of an entity (e.g., traffic sensor in a road, or person performing a check-in at a restaurant through a social network), or a reference to a point in space (e.g., geotagged photograph). Such data, ranging in $\mathbb{R}^2$ or $\mathbb{R}^3$ with *latitude / longitude* coordinates, a greater circle distance function, and an optional height attribute, will be referred to as *Geo-Spatial Data*.

Representing objects in space in a database system has applications in a wide range of use cases. Such data can be stored in spatial database systems, which [74] defines as database Systems that offer spatial data types in their data model and query language,

and support spatial data types in their implementation, providing at least spatial indexing and spatial join. Figure 2 shows the representation of a part of the physical world in a map with several annotations.

- Points represent data objects without or with irrelevant extent in space, for example the location of a resting area (right red balloon in Fig. 2) or the peak of an elevation (black triangle in Fig. 2).
- Lines or polylines describe connections in space such as roads, paths, rivers, but also borders and outlines.
- Polygons represent spatial regions of arbitrary shape or size, for example lakes, forests, or area under roof.

However in the scope of later parts of this work spatial data is understood more generally. Domain experts may choose to represent the objects of a database through a set of meaningful single-valued numerical features describing their characteristics, e.g., color, weight, size, rating, etc. This operation is referred to as feature transformation and is highly dependent on application domain. Selected features span a feature space and after said transformation, each database object may be represented by one point in this space. Such coordinates are called feature vectors, where each feature value is expressed as a coordinate in the corresponding dimension in the feature space:

$$o = (o_1, \ldots, o_n), o \in \mathbb{R}^n$$

Now the distance between two objects can be interpreted as similarity (small distance) or dissimilarity (large distance) between objects.

$$O \times O \to \mathbb{R}_0^+$$

Several distance measures have been proposed, however most commonly used and prominent are the L_p-norms which for two objects x and y in $\mathbb{R}^n$ are defined as follows:

$$L_p(x, y) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p}$$

Such generalized distance measures are essential in applications like neighbor search (cf. Chapters 9 and 11), or clustering (cf. Chapter 10).

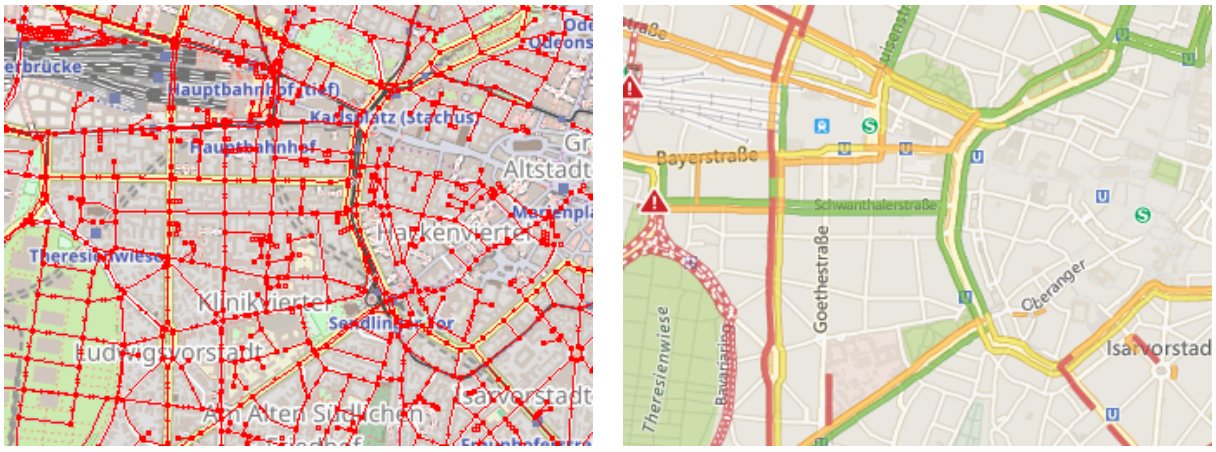
In addition to the spatial domain, geo-spatial data objects can contain numerous other attributes. Such *enriched geo-spatial data* contains further information, which is highly dependent on application. Some of those attributes may themselves be spatial, but can be anything from metric or ordinal data, complex structures, textual, to multimedia data. A few examples follow.

Numeric, categorical, or ordinal attribute data: an object identifier, class or category information, descriptive features such as age, elevation, speed limits, or average rating. An example application could be the rent index of a city – polygonal regions or single entities are annotated with an average rent price for this location. Such a database could yield the average rent around a certain query location, the lowest rent in a range, or a list of the $top-k$ most expensive areas in town.

Complex data structures: graphs or network data, such as public transit connections, or social network data describing relationships between objects. Another example, which will receive particular focus in this work, is data uncertainty in the form of probability distributions. Here the objects' attributes are associated with additional information about their distribution. This will be discussed in depth in Chapter 8. Social network data is another common enrichment type. Here the spatial entities, i.e., users and objects, are connected with an overlay of annotated connections. This could be $1-to-1$ friendship relations between users, $1-to-many$ associations between an object and users, e.g., check-in actions at a restaurant, and $many-to-many$ relations between users, e.g., user groups. Possible queries include a list of nearby friends or restaurants that have been rated high by friends. A common data mining task on geo-spatial data enriched with social networks is co-location mining [190] – finding groups of people that repeatedly frequent certain locations.

Textual attributes: names, descriptions, or annotations of the object. A common example are Twitter messages (tweets) that have an attached geo-location. Aside from also forming a social network, Twitter data is incredibly useful in tracing sentiment and trends due to its global spread and public availability in both creating and receiving content. Chapter 7 uses such data to track event dissemination on a global scale and compare patterns that emerge.
Since textual attributes may contain an abundance of qualitative information, evaluation can become a challenging task. Chapter 5 presents use cases where road networks are enriched with sentiment from tourist blogs. Here, a natural language processing approach was used to extract toponyms and sentiment from user-generated full text.

Multimedia data: image, audio, or video data that is associated to the object. Such data can originate from large content aggregators that also record geo-tags, e.g., flickr for photos or YouTube for video. Regarding the content, the same limitations as to textual data apply: qualitative information from multimedia data has to be interpreted before evaluation, unless the content is presented to the user in raw form.



(a) Road map with embedded graph information. Source: MARiO-framework with data from Open-streetmap

(b) Road map enriched with traffic information. Source: Bing Maps

Figure 2.2: Two examples for enriched map data

2.1 Application Examples

A very widespread application of enriched geo-spatial data are various types of maps containing additional information. Figure 2.2(a) shows a very intuitive example – a road graph embedded into the map connecting vertices in space. Such enrichments are vital to applications like navigation systems or route planning, where feasible connections need to be drawn between points on the map. Each of those road segments may contain further attributes relevant to the user, such as speed limits, elevation, or an average travel time.

More advanced systems provide time-dependent enrichments that can be either a live feed, such as sensor readings, or an estimation based on historic data. An example is shown in Figure 2.2(b), where some of the road segments of a map contain information about current traffic flow. Handling time-dependent enriched geo-spatial data is a very active research topic and has attracted much attention in the past years.

Related to the work presented in Chapter 5, [98] presents a system for touristic routing through a city, where the routes computed are based on road graphs containing numerous enrichments from relevant sources, such as travel blogs or picture databases. Scores and sentiment derived from this data is used to modify the routing cost function. In addition, the user is the able to access further information that enriches the result path, in this case user trajectories, photos of the area, and texts describing places nearby.

This thesis examines several kinds of data enrichment to spatial and geo-spatial data, initially subdivided in enrichment types of deterministic nature (cf. Part II) or with uncertainty information (cf. Part III). The following outline gives an overview of the individual topics.

Chapter 3

Outline

This thesis is subdivided into two main content parts for different types of enrichment data – Part II discussing deterministic data types and Part III explaining enrichment using uncertain data. Each part presents three use cases for different querying or mining techniques in their respective data spaces. An additional summary part concludes the thesis.

Part II, Chapter 5 is entitled **Trip Planning on Touristic Data**. Here, different types of crowd-sourced and mostly qualitative data are used to enrich a road network for touristic routing with the objective of guiding tourists along paths that may be more interesting to them. Data sources include textual information from blogs as well as geo-tagged photo databases. The proposed algorithms either make use of an enriched road network or construct a meta-network from the enrichment knowledge, and provide paths which reflect qualitative knowledge without sacrificing their quantitative aspect.

Another road network application is presented in Part II, Chapter 6: **Parking Queries on Road Network Data**. Additional information on the road graph is modelled as a set of consumable resources, along which the user is guided to maximize the probability of encountering an available resource. Modelling is facilitated through continuous-time Markov chains where resources are allowed to re-appear and short-term as well as long-term observations are incorporated. We employ two different search heuristics in a greedy algorithm to achieve a trade-off between accuracy and calculation time and present evaluations in two use cases – finding parking spots and finding available charging stations for electric vehicles.

A data mining approach for deterministic data enrichment is presented in Part II, Chapter 7: **Trend Dissemination Mining on Social Media Data**. Using user-generated geo-tagged data from Twitter, we map those tweets to historic trends and study the resulting dissemination of each trend over space and time. By applying a tensor factorization approach, we extracted latent features of trends, to which we applied a clustering approach to obtain sets of trends having a similar dissemination archetype. Our qualitative evaluation of these trend archetypes on Twitter trends show meaningful dissemination archetypes, such as political trends, celebrity trends, and disaster trends with only a small number of latent features.

Leading over to uncertainty in data, Part III, Chapter 8 introduces some of the models and techniques commonly used in handling uncertainty and gives some basic definitions.

As an example for query processing on uncertain data, Part III, Chapter 9: **Approximate Probabilistic Nearest Neighbor Queries** introduces index-supported techniques approximating the shape of Voronoi-cells to support nearest neighbor queries on uncertain data. Progressive and conservative approximations are given for the space which is guaranteed to be contained in the Voronoi-cell, in addition to the space which has a non-zero probability to be contained in the Voronoi-cell. Proposed algorithms are extended to k-th order Voronoi-cells and make use of several hybrid indexing techniques which are evaluated against each other as well as a competing approach which has been extended in functionality to meet the same criteria.

A data mining approach is presented in Part III, Chapter 10: **Approximate Probabilistic Representative Pattern Mining** where the problem of the large number of possible results is addressed by using a sampling approach, applying traditional methods on each sample individually, and combining the samples to find good representative results that also provide probabilistic guarantees for their cover of the result set. To the best of our knowledge, our approach is the first to yield clusterings associated with confidence, allowing the user to assess the quality of the clustering result. Furthermore, by returning multiple representative clusterings to the user, we can improve the quality (and therefore usefulness) of results, as shown by our experimental study.

Subsequently, Part III, Chapter 11: **Approximate Probabilistic Representative Spatial Query Processing** extends the same appproach to query processing on spatial data. Again, samples are drawn from the uncertain database on which traditional query processing techniques are executed. Several results are selected as representatives to be presented to the user, each of which associated with a confidence measure.

Finally, Part IV concludes this thesis.

Chapter 4

Publications and Co-Authorship

All research in this thesis was envisioned, conceptualized, formalized, and evaluated by the author of this thesis and his supervisors – mainly Prof. Dr. Matthias Renz[1], but also PD Dr. Matthias Schubert[2] and Prof. Dr. Mario A. Nascimento[3] – in cooperation with researchers at the Database Systems Group at the Department of Computer Science at LMU Munich and researchers at cooperating universities. The published content has been restructured, revised, and amended for this thesis by the author. In the following, the authors contribution to the publications is highlighted.

Part II: Deterministic Enriched Geo-Spatial Data.

Chapter 5: Trip Planning on Touristic Data – contents of this chapter have been conceptualized and written for submission to the Springer journal *GeoInformatica* "SSTD 2015 Best Paper Special Issue" and are based on previous results published in [172, 173, 171, 98]. In all previous papers as well as this version, the author contributed great parts of idea, concept, and algorithmic structure, as well as most of implementation, preparation, and execution of experimental parts.

Chapter 6: Parking Queries on Road Network Data – works in this chapter are in part based on [99] and were published in [97] in the 18th International Conference on Extending Database Technology. The author contributed great parts of conceptualization and the majority of experimental evaluation.

Chapter 7: Trend Dissemination Mining on Social Media Data – contents of this chapter were submitted to and accepted into the 11th International Workshop on Spatial and Spatiotemporal Data Mining held at IEEE ICDM 2016 and points into future research directions of the author. While this work was developed in a larger cooperation group between the LMU Munich, the Hong Kong University of Science and Technology, and the George Mason University of Fairfax, VA, the author developed major parts of ideas and concepts as well as the theoretical foundation for the trend dissemination and

[1]College of Science, George Mason University, Fairfax VA, USA
[2]Department of Computer Science, Ludwig-Maximilians-Universität München, Germany
[3]Department of Computing Science, University of Alberta, Edmonton AB, Canada

archetypes. Additionally, the experimental evaluation was exclusively implemented and conducted by the author.

Part III: Uncertain Enriched Geo-Spatial Data.

Chapter 9: Approximate Probabilistic Nearest Neighbor Queries – based on [58], the contents of this chapter were developed for submission to the Springer journal *GeoInformatica* "SSTD 2015 Best Paper Special Issue" and accepted with minor revision as of August 2016. The author contributed large parts in conceptualization and ideas, in particular regarding the extensions from the paper version, i.e., the guaranteed Voronoi cells and domination criteria of kNN-queries for k greater than one. In addition, experimental evaluation was mostly implemented and conducted by the author.

Chapter 10: Approximate Probabilistic Representative Pattern Mining – contents of this chapter were published in [211]. Large parts of conceptualization as well as the entirety of experimental evaluation was contributed by the author.

Chapter 11: Approximate Probabilistic Representative Spatial Query Processing – extending from Chapter 10, this work applies the concepts presented in [211] to spatial queries. Ideas and concepts of this chapter were largely contributed by the author. Additionally, the entirety of experimental evaluation was prepared, implemented and conducted by the author.

Part II

Deterministic Enriched Geo-Spatial Data

This part of the thesis will give several applications of handling geo-spatial data enriched with various types of deterministic data. It is subdivided into the following chapters:

- Chapter 5 surveys the field of enhancements to touristic routing in road networks by enriching them with data from various sources. Main objective is finding a "best path" as judged by users, which might depend on qualitative features, for instance the scenery or the touristic attractiveness of a path. Machines are unable to quantify such "soft" properties. Crowdsourced data provides with a means to record user choices and opinions using, for example, mobile apps or social networks. The presented techniques utilize heterogeneous sources of spatial, spatio-temporal and textual crowdsourced data as a proxy to capture qualitative preferences of users in movement. This allows for aligning routing approaches with human cognition.

  This chapter explores (i) the process of qualitative information extraction from uncertain crowdsourced data sets based on probabilistic frameworks and (ii) the enrichment of road networks with the extracted information by adjusting its properties and by introducing a meta-network. It is based on research published in [172, 173, 171, 98] and was submitted in an extended version as an invited contribution to *GeoInformatica Journal* "SSTD 2015 Best Paper Special Issue".

- Enrichment with probabilistic models and traffic data is discussed in Chapter 6. We generalize the problem of suggesting routes with a high probability to find parking spots and introduce a probabilistic formalization to model the availability of resources at certain locations. Our probabilistic model considers short term observations (e.g., vacant parking spots) as well as long term observations (e.g., average occupancy time) to adapt to the level of information currently available. In contrast to previous models, we allow resources to reappear after a probabilistically modeled amount of time (e.g., a car leaves a spot). Based on this model, we propose the so-called probabilistic resource route query with reappearance. In order to compute feasible solutions to this query in interactive time, we propose two greedy approaches. Furthermore, we examine backtracking for computing exact solutions and extend the proposed method into a significantly more efficient branch and bound algorithm.

  Contents of this chapter were published in the 18th International Conference on Extending Database Technology as [97].

- Chapter 7 contains future research directions that were submitted to and accepted by the 11th International Workshop on Spatial and Spatiotemporal Data Mining (*SSTDM*) held at *IEEE ICDM* in 2016.

  In this work, we investigate the dissemination of trends over space and time. For this purpose, we employ a four-step framework. In the first step, we employ existing solutions to mine a large number of trends. Second, for each trend we create a spatio-temporal dissemination model, which describes the motion of this trend over space and time. To model this dissemination, we employ a (flow-source, flow-destination, time, trend) tensor. In the third step, we cluster these trend-tensors, to identify

groups of archetype trends. For each archetype, we aggregate all tensors of the same archetype, and employ a tensor factorization approach to describe this archetype by its latent features. As the fourth step, we propose an algorithm which can classify the trend-archetype of a new trend, in order to predict the future dissemination of this trend.

Chapter 5

Trip Planning on Touristic Data

5.1 Introduction

Crowdsourced data has benefited many scientific disciplines by providing a wealth of new data. Technological progress, especially smartphones and GPS receivers, has greatly facilitated contribution to the plethora of available information. Nowadays, a large share of crowdsourced data (often also: user-generated content/information) contains spatial information, often referred to as volunteered geographic information (VGI). The term, however, is very generic and refers to various different types of content. VGI may refer to geo-tags which have been explicitly or implicitly added to a tweet, picture or status update. Also, a check-in at a registered location or a review for a restaurant's menu in a social network can be considered VGI. Other examples include the shared record of a user's favorite cycling route or, in the broader sense, a textual description of a museum in a blog entry.

In this work, we explore how different user-generated data sources may be used to enrich road networks in order to better represent the human way of thinking and liking. It is our hypothesis that crowdsourced data reflects the "mind of the crowd", that it conveys semantic knowledge and that it expresses sentiment. Algorithms as used by navigation systems, however, rely on purely quantitative measures, i.e. "hard" metrics. We argue that routing has a great cognitive aspect which is ignored by plain turn-by-turn instructions derived from absolute measures. Using crowdsourced data as a proxy, we aim to bring routing algorithm in line with users' preferences and cognition. For example, under the assumption that Flickr users take photos of particularly appealing places, a routing algorithm which is "steered in the direction" of areas where photos are dense, generates paths which are likely to be more appealing. Therefore, our goal is to integrate the wealth of crowdsourced spatial data into road networks, such that existing routing algorithms can be applied to find routes that better reflect human perception.

The aforementioned diversity of spatial crowdsourced data constitutes the main challenge of this work. Some data sources are noisier than others, and some have greater depth of information than others. For example, check-ins at curated locations are more reliable than geo-tagged tweets which are dependent on the quality of the user's GPS signal and the

density of possible Points of Interest (POI) in their environment. A plain textual mention of a "Tibetian Yoga Studio", for instance, may be ambiguous and might not be mapped to a unique location. As textual sources are particularly noisy, we develop specific methods to handle their inherent ambiguity. Aside from uncertainty in geo-spatial locations, sentimental information expressed in crowdsourced data may also be highly uncertain: While reviews in location-based services often represent distinct and thorough opinions, tweets usually contain extremely condensed sentiment. In contrast, travel blog entries are typically more focused on the entirety of a trip. Because of this diversity, there is no absolute truth on what to extract from which source. We do not claim to extract all possible information, instead, we want to give a broad overview on how to generate knowledge from a variety of crowdsourced data sources which can in turn be used for routing applications.

Most data sources reviewed in this work provide information about particular geographical locations or specific POIs. Depending on the type of data, these POIs might represent different categories. For example, mentions of POIs in travel blogs predominantly cover sights while check-ins in location-based services happen mostly at restaurants, bars or clubs. This effect can be used to enhance the semantic information being extracted. Another phenomenon which we try to trace in the process of knowledge extraction are particular relations between a number of POIs. Especially for the application of routing, it makes sense to not only consider singular POIs but multiple POIs representing a particularly related set or a specific sequence of POIs. When considering the sequence of check-ins of one user during a single day, the sequence may indicate a recommendable itinerary. When considering POIs mentioned in travel blogs, it might make sense to recommend those which are perceived positively, or it might make sense to recommend those POIs together that have a strong spatial connectivity. In contrast, when considering spots where notably many photos are taken, each particular spot might represent a great lookout point on its own.

As an example, consider the routing scenario in Figure 5.1 which is set in the city of Paris, France. The continuous line represents the conventional shortest path from starting point "Gare du Nord" to the target at "Quai de la Rapée" while the dot dashed and dotted lines represent alternative paths computed in enriched networks as proposed in this work. The triangles in this example mark POIs as extracted from travel blogs, in this case mostly landmarks and sights. For instance, the dot dashed path on the bottom right passing recognizable locations such as "Place de la République", "Cirque d'hiver" and "la Bastille" reflects the knowledge of the data source, i.e., it satisfies the requirement of being touristically appealing. Compared to the conventional shortest path, it will yield greater value for travelers.

Having extracted the crowdsourced information, we proceed to enrich the underlying network. It is our goal to merge the inherent metrics of the network (like distance and travel time) with the semantic knowledge filtered from the data sources. Ideally, this leads to networks whose cost criteria reflect the user-generated spatial information, allowing for alternative routing algorithms which do not only take quantitative measures but also qualitative knowledge into account. We propose and investigate different methods of enrichment of the underlying road network, and, additionally, we introduce a meta-network

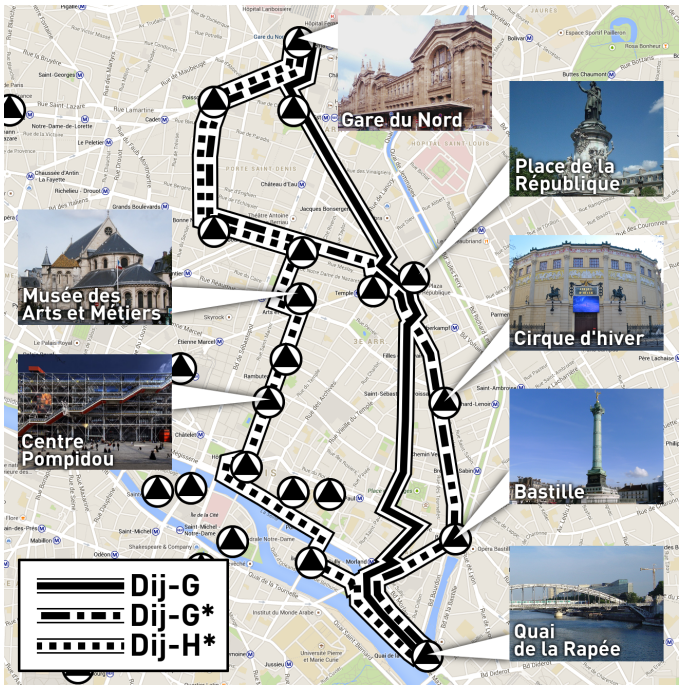


Figure 5.1: Shortest (continuous) and alternative paths (dot dashed and dotted) along POIs in Paris, France. This result is output of the methods presented in this work.

whose nodes correspond to POIs and whose edges correspond to shortest paths connecting them. Our evaluation is based on various real-world crowdsourced data sets. In the process of knowledge extraction different methods are applied, according to different points of view, as described above.

This work is divided into three parts. First, we give an extensive overview of different data sources, categorize them and mention advantages as well as possible drawbacks. We tackle the aspects of precision, reliability and information content. Second, we explore how crowdsourced knowledge can be extracted from different data sources. We present different approaches for singular, pairwise or sequential occurrences of POIs, and stress that diverse knowledge can be mined, depending on the approach. Third, we investigate how the knowledge may be integrated into the underlying road network in order to be implemented in routing algorithms. We propose to directly enrich the network or, alternatively, to build a meta-network which is built from the extracted knowledge. In our extensive experiments, we show that routing algorithms indeed benefit from incorporating crowdsourced knowledge. More precisely, at the cost of marginal increase in path length, we may generate paths which yield significantly higher value according to the data source which has been integrated. This work extends the research presented in [171], where we focused on spatial relations between pairwise occurrences of POIs. Incorporating ideas published in [98], we additionally present means for enrichment based on singular occurrences of POIs. Previously unregarded were sequences of POIs (triples or longer), which will be a novel extension in this work – theoretically as well as experimentally. In addi-

tion, we integrated further datasets. Finally, we restructured the chapter in order for this work to give a comprehensive survey of the means and methods for the enrichment of road networks with knowledge extracted from crowdsourced data.

The rest of this chapter is organized as follows: In Section 5.2 we summarize research from different areas which is related to this work. Section 5.3 classifies and distinguishes data types and mentions examples and sources. How and which particular knowledge may be extracted from these sources is detailed in Section 5.4. Section 5.5 introduces several methods for the enrichment of existing road networks with the knowledge extracted in the step before. The whole procedure, from data processing over knowledge extraction to network enrichment, is evaluated experimentally in Section 5.6. Section 5.7 concludes this work.

5.2 Related Work

In this section, we give an overview regarding the research relevant to our work which we divide into the following categories:

- 5.2.1 trajectories: mining semantic information and trajectory enrichment
- 5.2.2 qualitative routing
- 5.2.3 (crowdsourced) touristic trip planning
- 5.2.4 arc-orienteering paths

5.2.1 Semantic Mining and Enrichment of Trajectories

The discovery of semantic places through the analysis of raw trajectory data has been investigated thorougly over the course of the last years. The objective of this field of research is to analyze user trajectories and, in combination with POI databases, to extract semantically relevant places based on the spatio-temporal patterns (number of times POIs are visited and time spent there). The authors in [128, 196, 141] provide solutions for the semantic place recognition problem and categorize the extracted POIs into pre-defined types such as “home”, “work”, “education” and “shopping”. Moreover, the concept of “semantic behavior” has recently been introduced. This refers to the use of semantic abstractions of the raw mobility data, including not only geometric patterns but also knowledge extracted jointly from the mobility data as well as the underlying geographic and application domains in order to understand the actual behaviour of users in movement. Several approaches like [8, 142, 197, 198, 177, 199] have been introduced the last decade. The core contribution of these articles lies in the development of a semantic approach that progressively transforms the raw mobility data into semantic trajectories enriched with POIs, segmentations and annotations. Finally, a recent work, [60], extracts and transforms the aforementioned semantic information into a text description in the form of a diary. The difference between to what is presented in this work is that these approaches

do not intergrate the extracted semantic information into the road network. Instead, they combine trajectories with POI databases to extract semantic information on POIs and possibly enrich other trajectories (e.g., paths computed by an arbitrary algorithm) with semantic information. For instance, a sequence of timestamped geo-coordinates might be mapped to the semantic sequence: `home` → `work` → `kindergarden` → `supermarket` → `home` → `restaurant`.

5.2.2 Qualitative Routing

The term qualitative routing is not well-defined. We use it to describe two approaches to routing which do not solely rely on absolute measures and are therefore more "qualitative" rather than purely quantitative. First, the computation of routes which are easier to memorize, describe and follow. Second, the computation of routes which are particularly scenic, interesting or popular.

The first problem has been tackled from various angles and even research disciplines. Following a rather cognitive line of argument, the authors of [53] minimize the complexity of a route description. This is done by finding a trade-off between distance and weights which describe the complexity of the routing description at every intersection. Different cognitive models for the complexity can be employed. Later works explore the role of landmarks in route descriptions [54] and strategies on how to create compact route descriptions [150]. In [194], the authors explore the concept of route descriptions in detail by defining and evaluating agent models and deriving an agent-centric qualitative representation of the graph. A less cognitive and more spatial approach is chosen by the authors of [153]. They introduce cost criteria that allow for a trade-off beween distance and complexity based on network properties. It might be possible that the cognitive agent models could benefit from the extraction process presented in this work, however, this goes beyond the scope of the work. Our methods are based on crowdsourced knowledge such that ideally the crowd becomes its own agent, possibly making complex models obsolete.

The second problem, most prominently, has been examined in [146], which proposes a method for computing beautiful, quiet and happy paths, as the authors phrase it. In order to quantify these three qualities, the authors rely on explicit statements about the beauty of locations, obtained from a platform which asked users to specifically rate photos according to three categories. Obviously, this is a valid approach, however, it does not scale well. This is the reason why we propose to mine this kind of information from existing crowdsourced data, in order to avoid acquiring ratings in the manner of mechanical turks for locations on a global scale. Another way to generate interesting paths is to stitch previously recorded trajectories together, obtaining trajectories where every subtrajectory has previously been traveled by users and is thus "popular" following the definition in [33]. However, this only reflects a notion of common usage not taking into account, why a specific subtrajectory has been favored. For instance, when mining trajectories of commuters, fast paths are most likely to be chosen by most users, but when mining trajectories of runners, green paths will likely be preferred. Connecting parts from both sets, the obtained trajectories might fit into neither class.

5.2.3 Touristic Trip Planning

There are numerous variations of touristic trip planning problems but all are related to the original trip planning query (TPQ) [118]. The input of a TPQ are start and target locations in a weighted graph as well as a set of categories (of POIs). The output is the cost-optimal path from start to target visiting exactly one instance of each category. The NP-hard traveling salesman problem can easily be reduced to the TPQ by assuming that every POI belongs to its own category. Hence, efficient solutions to the TPQ or further constrained modifications are heuristic. However, in real-world scenarios the candidate sets of POIs may often be narrowed down drastically (e.g., by spatial pruning or additional constraints), allowing for a full enumeration of all solutions within seconds.

Early variations of the TPQ, [104, 32], allowed for a particular order of some of the categories and for further constraints regarding the entities of the categories. Further constrained modifications of the query often focus on the use case of touristic trip planning, ocasionally also referred to as itinerary planning, largely summarized in [65]. For example, the query objecive may be to maximize the subset of a set of predefined POIs which can be visited in a tour with a certain time-constraint [64]. Younger works in this area exploit crowdsourced data for POI categorization [30], for POI-popularity estimation [82], for POI-recommender systems [24], for the determination of opening hours or recommended visiting times [81], for deriving the average duration of stay at each POI [46] or for combinations of the above. In contrast to these works, we do not focus solely on the purpose of itinerary planning but follow a more general approach. We emphasize the process of knowledge extraction with regard to the diversity of the crowdsourced data, its quality and properties. Also, we investigate various different data types and sources and give insight on the aspects of related POIs and sequences of POIs. It is our belief that itinerary planning methods like the above could be refined using the contributions of this work. However, we choose not to adapt the rather rigid structure of the itinerary planning queries.

5.2.4 Arc-Orienteering Paths

Another relevant family of NP-hard problems are the orienteering problem (OP) and the arc-orienteering problem (AOP) and variants thereof. The input of both is a weighted graph as well as start and target locations and a cost budget, e.g., a maximal travel time. Additionally, there is a non-negative value assigned to the nodes of the graph of the OP, while for the AOP there is a value assigned to the edges of the graph (or arcs, hence the name). The output of both queries is the path from start to target which has the greatest accumulated value among all paths from start to target not exceeding the cost budget. The OP and numerous modifications have been thouroughly studied [186], especially in the field of Operations Research. Lately, the AOP has gained attention [176, 187], also in the database research community [127]. Contributions of this work have great relevance for OP and AOP, as the extracted knowledge can be conceived as a value associated with nodes or edges. Using these techniques, crowdsourced information may be integrated into existing solutions, adding another facette to the perception of “value” in OPs and AOPs.

5.3 Data Types and Processing

In order to enrich a road network, our approach combines data from vastly different sources of which each reflects a certain notion of value or popularity. Data may include geo-tagged multimedia data, check-in data as recorded by location-based (social) networks, GPS-trajectory data and even purely textual data. For each of these data sources, we propose one or more methods of knowledge extraction. We derive numerical values which can in turn be mapped onto the underlying road network and, thus, we enrich the road network with condensed crowdsourced information. As a first step, this section describes the data types and their sources as well as some preprocessing methods.

5.3.1 Spatially Enriched Data

By spatially enriched data we mean data with attached or inherent geo-tags, e.g., check-ins at pre-defined locations, geo-tagged tweets or geo-tagged images. The great benefit of basic spatial point data is its wide availability and large coverage. Most services relying on spatially enriched data provide APIs or at least exemplary data sets, such as Foursquare[1], Yelp[2], Twitter[3], Flickr[4] or the inoperative but oft-cited Gowalla. Check-ins at pre-defined locations, as recorded by location-based networks, provide precise information concerning the whereabouts of users (if not assuming deliberate misuse). This kind of reliability does not hold for all geo-tagged data, as the GPS sensors which typically provide the locations are never exact but, on the contrary, often imprecise. Other reasons for unrealibility may be data-dependent. Consider for example photos of a landmark like the Eiffel Tower. Because of its height, it is visible from far away, hence geo-locations may vary greatly. Therefore, it often makes sense to also consider dedicated meta-information such as Flickr tags or Twitter hashtags. How to incorporate this kind of information for probabilistic validation is explained in Section 5.4.1.

Of course, the spatial information is just an additional component to the actual content of the service, e.g., Twitter's tweets, Yelp's reviews or Flickr's photos. Depending on the content, different notions of information, value or popularity are reflected. For example, the number of Flickr photos in the vicinity of a particular POI may be interpreted as a measure for its appeal or, more generally, its popularity (as in [96, 171, 98, 127]). The assumption is that people tend to take photos at particularly appealing places, of scenic spots, of nice architecture. This might not be the case for all photo-sharing services (e.g., Snapchat), but whoever has scrolled through Flickr pictures will most likely agree to the assumption. Similarly, the number of check-ins in location-based networks like Foursquare and Yelp may also be considered a measure for trendiness or popularity (as in [24], [30], [81]). While Flickr photos tend to describe aesthetic appeal, accumulated check-ins rather reflect the popularity of restaurants, bars or clubs (according to the users of the particular

[1] www.foursquare.com
[2] www.yelp.com
[3] www.twitter.com
[4] www.flickr.com

service). This underlines the diversity of information provided by different data sources. A particularly ambiguous example are tweets which range from advertising over news and personal opinions to comical as well as serious content. Nevertheless, we choose not to disregard Twitter, as it is a rich source of crowdsourced information with millions tweets per day. How we cope with the ambiguities and extract knowledge from the data is also explained in Section 5.4.1.

5.3.2 Spatio-Temporal Data

We define spatio-temporal data as sequences of timestamped locations, possibly enriched with additional data, such as textual descriptions of a trip, of locations visited along the trip or meta-information like the vehicle used for the trip or the weather on that particular day. The prime example of spatio-temporal data are trajectories as collected by contributors to the open source map service OpenStreetMap[5], as contributed by users of shared mobility services such as Capital Bikeshare[6] or as recorded and uploaded by runners, cyclists or others to platforms like Endomondo[7]. However, any temporally ordered sequence of geo-locations can be classified spatio-temporal data, like consecutive check-ins of a particular user of a location-based network. When hand-ling trajectory data, usually the movement pattern is of interest, i.e., the actual path chosen by the user. Of course, this information is not available when mining consecutive check-ins. Although the actual path cannot be determined, a sequence of visited locations still might provide valuable information. For instance, if a significant number of users has visited the same locations within the time frame of a day (according to the timestamps), one may recommend visiting these locations as part of a day's trip (cf. [82], [24], [46]). Like in this case, spatio-temporal data may often be reduced to ordered sequences of spatial point data. Let us note that spatio-temporal data clearly suffers from the same measurement errors as spatial point data, but employing map-matching approaches (which often benefit from taking the timestamps into account) measurement errors are more easily rectified.

5.3.3 Textual Data

While spatio-temporal and spatially enriched data contain explicit spatial information, i.e., geo-locations, we now turn to implicitly spatial data. Pure textual narrative, for instance (non-geo-tagged) tweets, blog entries or other text corpora, often contains mentions of POIs. Obviously, a narrative is inherently noisy. Aside from the lack of geo-locations, the ambiguity of language and, more specifically, duplicate identifiers raise more difficulties. For instance, "Chinatown" is not a unique identifier, neither is "Gelateria Bella Italia" (it does not even relate locally), and a "Tibetian Yoga Studio" is most likely not in Tibet. Nevertheless, we want to stress the importance of textual data as it is a particularly rich source of crowdsourced information. Although authoring explicit spatial data has been facilitated

[5] www.openstreetmap.org
[6] www.capitalbikeshare.com
[7] www.endomondo.com

by technical progress, it sometimes still requires special applications and/or special knowledge, e.g., when contributing to OpenStreetMap. Hence, many users are more comfortable using narrative when generating content. Especially when evaluating visited places, users often generate narrative using qualitative adjectives such as "beautiful", "interesting" and "cool". Similarly with movement patterns, a lot of users describe their motion using toponyms (landmarks) and spatial relations ("near to", "next to", "close by", etc.) rather than using geo-coordinates. Hence, there is a largely unused abundance of crowdsourced knowledge in the form of blog entries or other narratives (freely) available on the internet. In the following, we describe how we mine toponyms and in turn geo-locations from travel blogs as an example for crowdsourced textual data. This is a prerequisite for the knowledge extraction methods presented in Sections 5.4.1 and 5.4.3.

In order to gather travel blog entries, we use standard web-crawling techniques and compile a database consisting of 250,000 blog entries from 20 different travel blogs [8] as presented in [172, 170]. Extracting qualitative information from text requires the detection of toponyms, i.e., placenames within the raw text. By geoparsing, candidate phrases containing references to POIs are identified. Let us note that geoparsing is a method well-proven in the field of Natural Language Processing, and we used the Natural Language Toolkit [126] in our implementation. Subsequently, we geocode these POIs, i.e., we map the POIs onto geo-locations. This is done using the geographical gazetteer database GeoNames[9] which contains over ten million POI names, their synonyms and their coordinates worldwide. Of the 500,000 POIs mined from the text corpus, we were able to geocode 480,000. For further details, we refer the reader to [169] and [170].

5.4 Knowledge Extraction

This section in detail describes the knowledge extraction process of qualitative information from crowdsourced data with increasing complexity. First, singular occurrences of POIs in the data are described, then the importance of relationships and sequences is stressed. Particular attention is paid to textual data due to its ambiguity and noise.

5.4.1 Single Occurrences

First, we describe the simplest approach to knowledge extraction from crowdsourced data sources. Accumulation of separate occurrences is a basic but effective method for quantifying spatial information resulting in simple numeric scores which reflect some notion of value or popularity. As mentioned before, the number of photos taken in the vicinity of a POI or the number of check-ins at particular locations have both been employed as a measure for popularity in research (see Section 5.3.1). When considering check-ins, the basic approach

[8] http://www.travelblog.com/
http://www.traveljournal.com/
http://www.travelpod.com/

[9] www.geonames.org

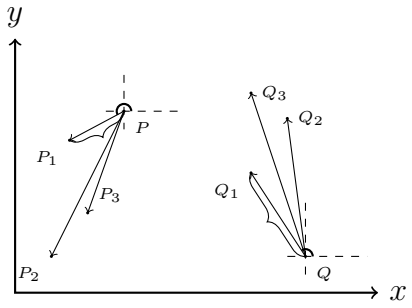


Figure 5.2: Illustration of spatial feature vectors between two POIs P and Q and their respective photos P_i and Q_i.

is to sum up the number of check-ins at each location and directly use this score as a measure (normalized by the maximum number of check-ins). A straightforward extension is to factor in the location-specific rating provided by the corresponding location-based social network, e.g., the ten-point-rating of Foursquare or the five-star-rating of Yelp.

In contrast, geo-tagged photos are not necessarily taken exactly at the POI they depict but in a more or less strict vicinity. The basic approach is to consult a POI database (like GeoNames or appropriately tagged OpenStreetMap nodes), to define a distance threshold ε and to sum up the number of all photos in the ε-range of each POI. However, depending on the threshold value, photos might be assigned to more than one POI. This can be avoided setting $\varepsilon := d_{\min}/2$, where $d_{\min}$ is the minimum distance between any pair of POIs in the considered network. Still, photos might be assigned to the wrong POI, as large landmarks or buildings on spacious esplanades, for instance, are visible from a greater distance than smaller ones. Better results may be achieved when additionally considering the Flickr tags (or similar features on other platforms). Using a gazetteer database such as GeoNames which provides synonyms (e.g., "Cologne Cathedral", "Kölner Dom", "Hohe Domkirche St. Petrus"), the text tags of all photos in a greater vicinity of a POI may be scanned for word matches and assigned to that particular POI. The number of photos assigned to a POI again gives an estimate for its popularity. Normalized by the maximum number of photos of one POI, this gives a score in the interval $[0, 1]$.

Not all photos are text-tagged properly. Therefore, we propose a probabilistic modeling approach for non- or insufficiently text-tagged photos. Consider the following scenario: For a specific POI, for example the "Eiffel Tower", let there exist n geo-tagged photos $\{p_1, \ldots, p_n\}$ which are also properly text-tagged, i.e., as "Eiffel Tower", "Tour Eiffel" or another valid synonym. We may use these to train a probabilistic model in order to classify the geo-tagged but not text-tagged photos in the vicinity, as proposed in [169]. From the POI's actual location (as stored in the database) and the photos' geo-tags, we may compute a two-dimensional spatial feature vector $v = (r, \phi)$ for each photo, containing the Euclidean distance and the orientation w.r.t. the counterclockwise rotation of the x-axis (centered at the actual location). These n feature vectors may be used to train a probabilistic model. In

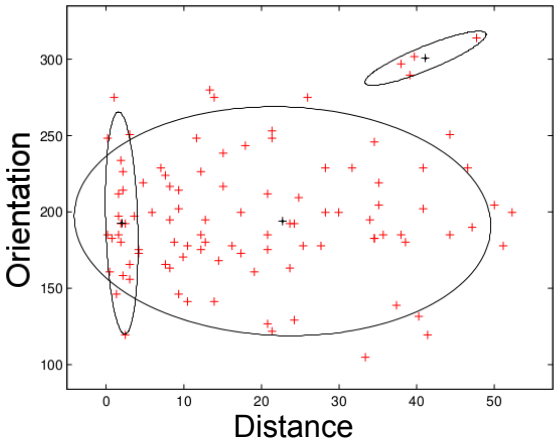


Figure 5.3: 3-component GMM trained on two-dimensional point data set visualized upon convergence of Expectation Maximization.

Figure 5.2 two POI locations P and Q are illustrated. Each POI is assigned three photos (for one of which distance and orientation are visualized).

We propose to train a Gaussian Mixture Model (GMM) which is a well-proven and extensively studied method for many supervised and unsupervised learning problems [21]. A GMM is a weighted sum of a fixed number M of Gaussian component densities

$$\mathbb{P}_\theta(v) = \sum_{i=1}^{M} w_i g(v; \mu_i, \Sigma_i)$$

where v is an l-dimensional vector, w_i are the mixture weights (summing to 1) and

$$g(v; \mu_i, \Sigma_i) = \frac{1}{\left((2\pi)^l \det(\Sigma_i)\right)^{1/2}} \exp\Big(-\frac{1}{2}(v-\mu_i)^T \Sigma_i (v-\mu_i)\Big)$$

is an l-variate Gaussian density function with mean vector $\mu_i \in \mathbb{R}^l$ and covariance matrix $\Sigma_i \in \mathbb{R}^{l \times l}$. The model is fully characterized by the weights, mean vectors and covariance matrices, collectively represented in $\theta = \{w_i, \mu_i, \Sigma_i\}, i = 1, \ldots, M$. In our case, $l = 2$, the dimensionality of the spatial feature vectors v. Figure 5.3 shows a 3-component GMM trained on two-dimensional point data.

For the parameter estimation of each Gaussian component, Expectation Maximization ([48]) is the state-of-the-art technique which updates the parameters of the components iteratively w.r.t. a given (feature) vector set until a convergence threshold is reached. Thus, using all geo-tagged and properly text-tagged photos of a particular POI POI, we obtain a probabilistic model $\mathbb{P}(\cdot \mid \theta)$ where θ is specific to POI. Applying the model to the spatial feature vector of a non-text-tagged photo, we obtain an probabilistic estimation of the photo's affiliation to that POI. If this probability is sufficiently high, i.e., exceeding

an application-dependent threshold, the photo may be assigned to the POI. Consider Figure 5.2, with a GMM trained on the given data without photo P_i, we could infer (with high probability) that P_i indeed depicts POI P and not POI Q.

Besides check-ins and photos, we propose another measure for popularity based on single occurrences of POIs in crowdsourced data. While photos tend to measure appeal and check-ins indicate popularity, we also exploit a combination of two crowdsourced data sets to infer sentiment according to Twitter. As described in Section 5.3.3, we have access to the names and geo-locations of touristically relevant POIs. Crawling Twitter's public feed for mentions of each POI's synonymous toponyms, it is possible to obtain a significant number of tweets in the area of interest. Finally, to each tweet we apply the sentiment analysis feature of the Natural Language Toolkit [126] and aggregate the output score for each POI separately. This gives us another notion of popularity in terms of a numeric score for each POI reflecting the sentiment of Twitter users in regard to that POI.

5.4.2 Pairwise Occurrences

So far, we have mentioned how knowledge can be extracted from single occurrences of POIs. It is our goal to propose methods of knowledge extraction for routing purposes. Thus, we now investigate how to consider sequences rather than separate POIs. By considering pairs of POIs, for example, it is possible to model connections between POIs. Depending on the nature of the pairings, different knowledge will be reflected in the connections. For example, if a significant number of users checked in at the same two locations, it might make sense to recommend the pair of locations rather than each location separately when planning an itinerary or a route. As mentioned in Section 5.4.1, check-ins at pre-defined locations are not subject to spatial variation as the set of locations where checking in is possible are curated. Therefore, it suffices to simply sum and normalize occurrences of pairs of check-ins in order to score their value when visited in combination. As before, additional information such as network-specific ratings may easily be taken into account. More importantly, when considering pairs of POIs visited in conjunction, spatial connectivity will become a factor. We explore this aspect in the following using purely textual data to show that even intricate data sets can be mined for knowledge about pairwise occurrences, such as spatially close pairs of POIs.

We make use of the text corpus described in Section 5.3.3. More precisely, we now consider toponyms (mapped to POIs) which are linked by spatial relations describing closeness, such as "nearby", "next to", "at", "in" or "in front of". We refer to these spatial relations as closeness relations. If a pair of POIs is mentioned significantly often linked by a closeness relation, one may deduce that the two POIs are in fact close to each other. In the following, we present a probabilistic approach to minings pairs of POIs which – according to the text corpus of travel blogs – stand in close spatial relation to each other. From the text corpus of 250,000 travel blog entries, we were able to mine 660,000 triplets of the form $(P_i, \textit{closeness relation}, P_j)$. Here and in the following, we denote POIs by P_k with varying index. A sample of POIs in London, UK, as well as New York City, US, and their respective closeness relations are visualized in Figures 5.4.

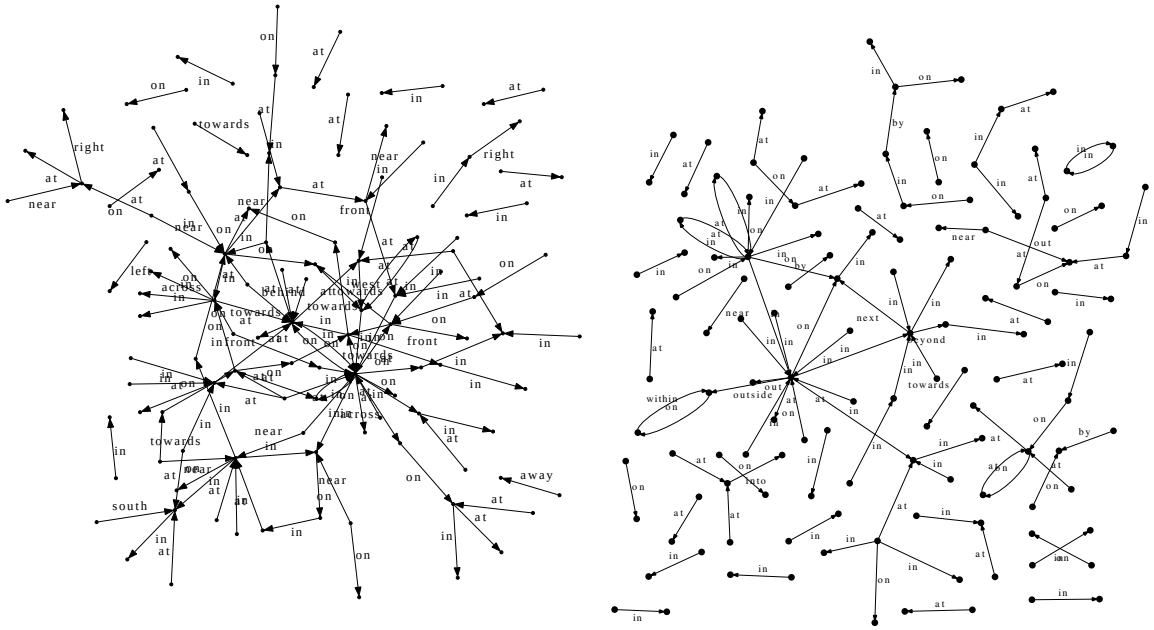


Figure 5.4: POIs (nodes) and their relations (edges) extracted from travel blogs. Visualized are two samples from London, UK, (left) and New York City, US.

As in Section 5.4.1 (and as presented in [171]), we rely on a probabilistic model to counter the ambiguity inherent in the data source. Similar to before, for each occurrence of a particular closeness relation, we create a two-dimensional spatial feature vector $v = (r, \phi)$ where r denotes the distance and ϕ denote the orientation. Now, however, distance and orientation are not relative to a fixed POI but relative to the two POIs which stand in relation to each other. For instance, assume $(P_i, \text{“next to”}, P_j)$ is a triplet mined from the text corpus, then v_{ij} describes Euclidean distance between P_i and P_j and the orientation as the counterclockwise rotation of the x-axis, centered at P_j, to P_i. Again, we obtain a set of two-dimensional spatial feature vectors V_{REL^k} for each of the closeness relations REL^k. These will in turn be used to train two-dimensional Gaussian Mixture Models, yielding a set of probabilistic models $\mathbb{P}(\cdot \mid \theta^k)$, one for each closeness relation $REL^k \in \{REL^1, \ldots, REL^m\}$, the set of all closeness relations we take into account. For a pair of POIs P_i and P_j with spatial feature vector v_{ij}, $\mathbb{P}_{\theta^k}(v_{ij})$ is the probability that P_i and P_j stand in spatial relation REL^k to each other. Based on this information, we now derive a closeness score for pairs of POIs by Bayesian inference.

For two distinct POIs, let REL_{ij} denote the set of all closeness relations existing between P_i and P_j. Note that REL^k denotes an abstract relation, while REL_{ij} denotes the set of occurrences of relations between a pair of POIs. Furthemore, let v_{ij} denote the spatial feature vector of P_i and P_j. In order to determine a numeric score describing the closeness of the POIs, we estimate the posterior probability of all closeness relations REL^k

and accumulate them. The posterior probability of relation REL^k is given by

$$\mathbb{P}(REL^k \mid v_{ij}) = \frac{\mathbb{P}(v_{ij} \mid \theta^k)\ \mathbb{P}(\theta^k)}{\sum_{l=1}^{m} \mathbb{P}(v_{ij} \mid \theta^l)\ \mathbb{P}(\theta^l)}$$

where $\mathbb{P}(\theta^k) = \mathbb{P}(REL^k)$ denotes the prior probability of relation REL^k given by the trained model represented by θ^k.

In a traditional classification problem the task would be to choose the closeness relation with the highest posterior and to assign the pair of POIs to this class. We, in contrast, consider each posterior probability $\mathbb{P}(REL^k \mid v_{ij})$ as a measure of confidence of the existence of REL^k between P_i and P_j. Stressing that all considered relations represent spatial closeness, we combine all posteriors into one measure which we call *closeness score* cs_{ij} of the pair of POIs P_i and P_j:

$$cs_{ij} = \frac{1}{m} \sum_{k=1}^{m} \frac{\mathbb{P}(REL^k \mid v_{ij})}{\max\{\mathbb{P}(REL^k \mid v_{ij}) \mid \forall\ i \neq j\}}$$

From pure textual data in the form of travel blogs, we first mined triplets of toponyms linked by spatial closeness relations. The toponyms were mapped onto POIs (i.e., their locations and synonyms). For each pair of related POIs, we computed a spatial feature vector, and the entirety of these vectors was used to train GMMs, one for each relation. Finally, we computed a closeness score from the posterior probabilities of each relation given spatial feature vectors. This score reflects a confidence in the notion of closeness as reflected by the spatial relations and by the underlying data. Hence, this method can be used to evaluate occurrences of pairs of POIs in textual data w.r.t. their closeness. In the following, when speaking of a pairwise occurrence (of POIs), we mean a mined pair of POIs with non-zero or sufficiently significant score (greater than a given threshold).

5.4.3 Sequential Occurrences

Extending the above introduced concept, we now consider sequential occurrences of POIs having more than two elements. By the same logic as above, general sequences of POIs might bear additional information compared to single or pairwise occurrences. In particular, itineraries might be refined incorporating triples, quadtruples or even longer sequences of POIs. In the use case of itinerary planning, combining two separate pairs of occurrences might yield unwanted categorical duplicates. Consider, for instance, one frequently visited pair of POIs where one POI is a restaurant and another frequently visited pair with another restaurant. In this case, concatenating the two pairs will guide the user to two restaurants, possibly within a short time-span. When mining longer sequences of POIs, we gain valuable information about combinations of POIs which the crowd found to be valid. For example, longer sequences might also contain two restaurants. However, when mining such a sequence frequently, it implies a validation by the crowd, e.g., one restaurant might be a great lunch spot, the other a nice dinner place. Accumulating the significant

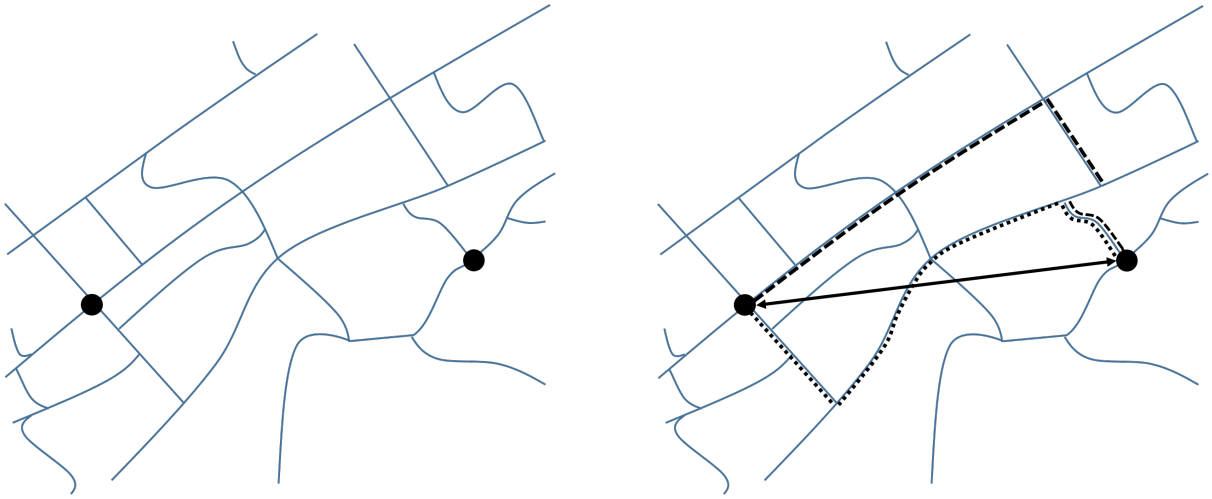
sequences, we obtain a measure for the relevance of a certain combination of POIs. Clearly, this also holds for similar sequences in other types of crowdsourced data, like, for instance, Flickr photos. Mining sequential photos of Flickr users to extract knowledge about their movement patterns is an established approach in the research community [46, 24].

From a theoretical point of view, the process of knowledge extraction is similar to that described above. Consider, for example, consecutive Foursquare check-ins by one user during a day. If the user checks in at two places, the combination forms a pair, if they check in at three places, a triple is formed. In this interpretation of the data, the two pairs contained in a triple need not necessarily form a pair themselves. Another way of looking at the data might be to imply that every relevant triple also generates two pairs. Besides consecutive Foursquare check-ins and Flickr pictures, one may also extract sequences of POIs from textual sources, for example from travel blog entries, following the approach introduced above. The challenge is now, how to enrich the network with the obtained scores. Consider the following example: Assume you have mined and quantified the score of two POI pairs (P_i, P_j) and (P_j, P_k) and the score of their concatenation (P_i, P_j, P_k). The score of the triple might surpass the score of the pairs. This might be because relevant triples are not considered to generate pairs or simply because triples are scored higher than pairs . The question on how to use the quantified knowledge in order to enrich the underlying road network is tackled in the next section. As before, when speaking of a sequential occurrence (of POIs), we mean a mined sequence of POIs with non-zero or sufficiently significant score (greater than a given threshold).

5.5 Enrichment

In this section, we present our approach to map the numerical score derived in Section 5.4 to the underlying road network. This section will show how the popularity scores presented in the previous section are made actionable for routing purposes. We explore two different approaches: For the first approach, presented in Section 5.5.1, we pursue the idea that a gain in score corresponds to a reduction in cost, i.e., gain and cost are assumed to be reciprocal. Conventional routing algorithms expand paths with promising edges, i.e., edges with low cost values. By reducing the cost of edges with non-zero score, conventional routing algorithms favor these edges, steering the exploration in the direction of areas with non-zero scores. For the second approach, presented in Section 5.5.2, we introduce a meta-network where nodes correspond to POIs and edges correspond to shortest paths between them. By requiring subpaths to follow the connections between POIs, this allows for a greater restriction of the result paths, a stronger binding to the network layer containing the POIs.

Both presented approaches utilize existing shortest path algorithms, including Dijkstra's algorithm [50], $A^\star$ [76], route skyline algorithms [160] and similar approaches, which we refer to as *conventional routing algorithms*.



(a) Part of a road network with two POIs (black circles).

(b) Illustration of two possible skyline paths connecting a pair of related POIs.

Figure 5.5: Example Road Network

5.5.1 Enriching the Road Network

We model any road network as a directed and weighted graph, based on OpenStreetMap[10] data, as illustrated in Figure 5.5(a). We denote the graph by $G = (V, E, c)$, where the vertices (or nodes) $v \in V$ correspond to crossroads, dead ends etc., and the edges $e \in E \subseteq V \times V$ represent streets connecting vertices. Furthermore, let $c : E \to \mathbb{R}_0^+$ denote the function which maps every edge onto its cost criterion. We introduce the following notations: $e_{uv} = (u, v)$ and $c_{uv} = c(e_{uv})$. If not stated otherwise, the cost criterion is distance. Other possible criteria are, for instance, travel time or energy consumption. If multiple cost criteria are used, they are denoted $c_1, \ldots, c_k$. Furthermore, let $\mathcal{P}$ denote the set of POIs. We assume every POI to be a node in the graph, i.e., $\mathcal{P} \subseteq V$. This is only a minor constraint as we can easily map each POI to the nearest node of the graph or introduce pseudo-nodes. Finally, a set of consecutive edges is referred to as path. Clearly, the notion of a cost criterion c extends naturally to any path p by defining $c(p)$ as the summed cost of its edges.

Single Occurrences

In Section 5.4.1 we describe the knowledge extraction process considering single occurrences of POIs in crowdsourced data. The output of this process is a normalized, node-associated score, i.e., POIs are assigned a numeric score in the range $[0, 1]$, describing some notion of value or popularity. 0 corresponds to POIs with no score, and 1 corresponds to the highest possible score. In order for this score to be used in conventional routing algorithms, it has

[10] www.openstreetmap.org

to be offset against the underlying cost criterion. Otherwise, the optimal path would be the solution to a traveling salesman problem visiting all POIs to acquire the maximum possible score. Hence, as mentioned before, we consider the score to be a reduction in cost. More precisely, in the following, we convert the node-associated score into an edge-associated cost.

Let $s(v)$ denote the score we associate with a node $v \in V$. Note that for $v \in V \setminus \mathcal{P}$: $s(v) = 0$, and for $P \in \mathcal{P} : s(P) \geq 0$. For each edge $(u, v) = e \in E$, we define the *score-discounted cost (sdc)* $sdc(e)$ as

$$sdc(e) := c(e) \cdot \phi^{\kappa(s(u)+s(v))} \tag{5.1}$$

where κ denotes a scaling parameter dependent on the data source and $\phi \in (0, 1)$ is a scaling factor for the influence of the respective score, similar to the approaches presented in [96, 98]. Intuitively, if e connects two vertices u and v with score 0, $sdc(e)$ equals $c(e)$. As the total score $s(u) + s(v)$ of nodes u and v increases, the score-discounted cost of e decreases exponentially. Thus, for an exceptionally large score of u and v, the adjusted cost $sdc(e)$ of edge e approaches zero. The parameters ϕ and κ control how quickly the score-discounted cost $sdc(e)$ converges to zero for an increasing aggregate score $s(u)+s(v)$ of the two vertices connected by e. For a larger parameter κ, the discount $sdc(e) - cost(e)$ increases more quickly, making the discounted edge e more attractive for conventional routing algorithms. Analogously, a smaller parameter ϕ yields the same effect. We use κ to normalize the popularity score across different data sources, ϕ in contrast is a global scaling parameter.

If a given road network graph G is enriched with a score function *sdc* reflecting value or popularity of single occurrences of POIs, we refer to $G^1 = (V, E, sdc)$ as the *enriched (road network) graph*. Since each edge in G^1 is given the score-discounted cost $sdc(e)$, the notion of optimal paths will change given the new cost measure. This way, we achieve our goal of incorporating the notion of crowdsourced popularity or value using traditional shortest path algorithms.

Pairwise Occurrences

Now, let us consider pairwise occurrences of POIs such as consecutive check-ins of a user of a location-based network. As mentioned before, the actual route the user chose between those check-ins is not recorded. Although there is a (spatial) connection between the check-in locations, it is unclear which part of the network should be enriched. Thus, we make the assumption that users generally prefer cost-optimal paths, i.e., paths which are optimal under the given cost criterion or criteria, such as the shortest path, fastest path or paths in the route skyline. We propose to discount all edges along these paths, such that routing algorithms will prefer sections of the network which have been favored by the crowd according to the underlying data set. To summarize and illustrate our approach to enrich the network with pairwise occurrences of POIs, reconsider Figure 5.5(b). The set of optimal routes with respect to multiple cost criteria is highlighted. Our approach increases the popularity and thereby decreases the cost of all edges located on any of the highlighted

routes connecting the POIs. This approach is described in more detail in the following. First, we describe our method for selecting the paths to be enriched, then we describing our approach to enrich this set of paths in the road network.

If the underlying road network uses one cost criterion only, usually distance or travel time, we simply enrich the cost-optimal path connecting the pair of POIs. In multicriteria road networks, e.g., additionally holding cost values for energy consumption or road tolls, there are several possibilities of enrichment. A straightforward approach is to discount the cost-optimal paths according to each criterion, i.e., enriching up to d paths with crowdsourced knowledge when the network has d criteria. However, in order to increase the density of enrichment, we propose to enrich the path skyline or linear path skyline [160, 167, 166]. The path skyline contains all paths which are optimal under some monotonic combination function of the cost criteria. Depending on the number of cost criteria and on the distance between start and target node, the number of paths in the conventional skyline quickly deteriorates. Thus, if the number of cost criteria is relatively high (more than three) and/or the average extent between query end points is relatively big (> 100 km), it is recommendable to use the linear path skyline approach to restrict the influence of the enrichment. Once we have selected the set of paths to enrich, we proceed as follows.

Let (P_i, P_j) denote a pair of POIs mined from the data. For instance, two frequent consecutive check-in locations or two POIs which are connected by spatial closeness relations. According to one of the definitions above, we compute the set of (linear) skyline paths, denoted by $S_{i,j}$. Although the paths contained in $S_{i,j}$ differ from one another, they often share edges. We discount the cost of an edge only once, even if it occurs in more than one skyline path. Let $E_{i,j} \subset E$ denote the set of all distinct edges which are part of at least one (linear) skyline path from P_i to P_j. Then we define the score-discounted cost of cost criterion c_i of an edge as

$$sdc_i(e) = c_i(e) \cdot \prod_{e \in E_{i,j}} (1 - \phi s_{ij})$$

As before, $\phi \in (0, 1)$ is a scaling factor. s_{ij} denotes the data set-specific score of the pair of POIs. For example, s_{ij} might be the number of consecutive check-ins at P_i and P_j normalized by the maximum number of check-ins at a pair of POIs. Or, $s_{ij} = cs_{ij}$ might be the closeness score defined in Section 5.4.2, an aggregate for spatial closeness relations extracted from text. Analogous to before, given a road network graph G, we define the enriched (road network) graph for a score s_{ij} reflecting connections between pairs of POIs as $\mathrm{G}^2 = (V, E, sdc_1, \ldots, sdc_k)$.

Discussion

All of the approaches of Section 5.5.1 modify the costs in the underlying road network. The cost of an edge is reduced if a relevant amount of qualitative information regarding that edge (or its immediate vicinity) has been found. By employing conventional routing algorithms which construct cost-optimal paths w.r.t. one or more cost criteria, cheaper edges will be favored over more expensive ones. More precisely, if edges e and f have the same cost but e has higher score or than f, then $sdc(e) < sdc(f)$. When mining single

occurrences of POIs from crowdsourced data, this adequately reflects the local influence of the score of each POI on the cost of the adjacent edges. When considering pairs of POIs the cost of those edges is reduced which are part of some cost-optimal path connecting the pair. This, however, does not "force" routing algorithms to compute a path which actually visits both these POIs.

5.5.2 Creating a Meta-Network

To ensure that in such cases both POIs are indeed visited, we now introduce a meta-network. This meta-network also allows to consider sequential occurrences of POIs mined from the data. We refer to this meta-network as *POI graph* because the nodes of the graph correspond to POIs and the edges to paths connecting these POIs in the underlying road network.

Enrichment of Pairwise Occurrences

In the following, we build a POI-graph G^2_{POI} using pairwise occurrences. Each POI mentioned in any occurrence (or in significantly many) forms a vertex in G^2_{POI}. For any pairwise occurrence between two POIs P_i and P_j, a corresponding edge e_{ij}, i.e., a shortcut, is added to G^2_{POI}. This edge holds a reference to the cost-optimal path (and its accumulated cost) connecting the pair. If the underlying road network is a multicriteria network, it is possible to precompute the cost-optimal path for each criterion. Figure 5.5(b) depicts an example: Here, a new edge is added as a shortcut between the two depicted POIs. The shortcut edge is not present in the underlying road network shown in Figure 5.5(a).

When issuing a query, the user inputs start and target nodes. Of course, these are not necessarily POIs, i.e., part of G^2_{POI}. Thus, we propose using both graphs, G and G^2_{POI}, in combination with a slight modification of Dijkstra's algorithm, as pseudo-coded in Algorithm 1. The idea is to find entry and exit POIs which are close to the start and target node, respectively. Subsequently, two paths are computed in G: the cost-optimal path from s to the entry POI and the cost-optimal path from the exit POI to t. Between entry and exit POI, routing is executed in G^2_{POI}, i.e., always following cost-optimal paths between pairs of POIs which are related according the crowdsourced information.

Enrichment of Sequential Occurrences

So far, we only discussed the enrichment of single and pairwise occurrences. The reason is that in order to enrich the network with crowdsourced knowledge from sequential occurrences of POIs, a meta-network is needed. Assume that in one scenario only the two POI pairs (P_i, P_j) and (P_j, P_k) were mined from the data, and in another scenario the triple (P_i, P_j, P_k) was mined from the data. Thus, in the former case, users frequently travel from P_i to P_j, and from P_j to P_k, but users do not take the full journey from P_i to P_k (via P_j). In the latter case, the full journey was mined as a frequent sequence of POIs, and thus, P_j might be a simple stopover in-between two popular POIs P_i and P_k. By only

Algorithm 1: Using the meta-network for path computation

Input: G, G^2_{POI}, start s, target t
Output: Path connecting s and t

begin
  P_{entry} = closest POI to s in G^2_{POI}
  P_{exit} = closest POI to t in G^2_{POI}
  $(s, \ldots, P_{entry})$ = cost-optimal path in G from s to P_{entry}
  $(P_{exit}, \ldots, t)$ = cost-optimal path in G from P_{exit} to t
  $(P_{entry}, \ldots, P_{exit})$ = cost-optimal path in G^2_{POI} from P_{entry} to P_{exit}
  $(P_{entry}, \ldots, P_{exit})'$ = mapping of $(P_{entry}, \ldots, P_{exit})$ onto the road network G
  return *concatenation of* $(s, \ldots, P_{entry}), (P_{entry}, \ldots, P_{exit})', (P_{exit}, \ldots, t)$
end

enriching pairs of POIs, the distinction between these two scenarios is not possible. The information that visiting all three POIs in sequence is popular according to the crowd, is lost. Therefore, we extend the concept of the POI graph, first to triples of occurrences, then to arbitrary sequences.

We denote the extension of a POI graph G^2_{POI} to triples of POIs as G^3_{POI}. Before, POIs which occurred in sufficiently many pairs according to the data formed a node. In G^3_{POI}, each POI forms a node which occurs in sufficiently many pairs or sufficiently many triples with greater score than its two consecutive pairs. Therefore, we only introduce edges for triples which bear greater score when visited in sequence than visiting its two pairs separately, i.e., $s(P_i, P_j, P_k) > s(P_i, P_j) + s(P_j, P_k)$. As before, each pair of POIs mined from the data is connected by an edge, also called *direct edge* for distinction. Additionally, for any triple (for which holds $s(P_i, P_j, P_k) > s(P_i, P_j) + s(P_j, P_k)$), we introduce a *indirect edge (or shortcut)* in G^3_{POI} from P_i to P_k holding a reference to the concatenated cost-optimal path from P_i to P_j and from P_j to P_k. This is illustrated in Figure 5.6 where the doubled lines represent shortcuts visiting triples of POIs and the single lines represent paths between POIs. We refer to the middle node of a shortcut, as seen in Figure 5.6a) as an *intermediate node*. In the following, we denote the direct edge from P_i to P_j by e_{ij} and the indirect edge (P_i, P_j, P_k) by e_{ijk}.

The idea of introducing shortcuts to model particularly valuable sequences of POIs is appealing. However, conventional routing algorithms may generate result paths with cycles. This is due to two reasons. First, intermediate nodes are not explicitly visited by a routing algorithm and can therefore not be flagged appropriately. Second, in order to promote the usage of sequences of POIs, we discount shortcuts which may lead to violations of the triangle inequality. e.g., $c_{ijk} < c_{ij} + c_{jk}$. Before we decide which cycles to avoid and how, we introduce definitions to distinguish different types of cycles.

Definition 1 *In a directed graph, a cycle is a path (oriented and consecutive set of edges) where no node is visited twice except for the start/end node. We distinguish between cycles*

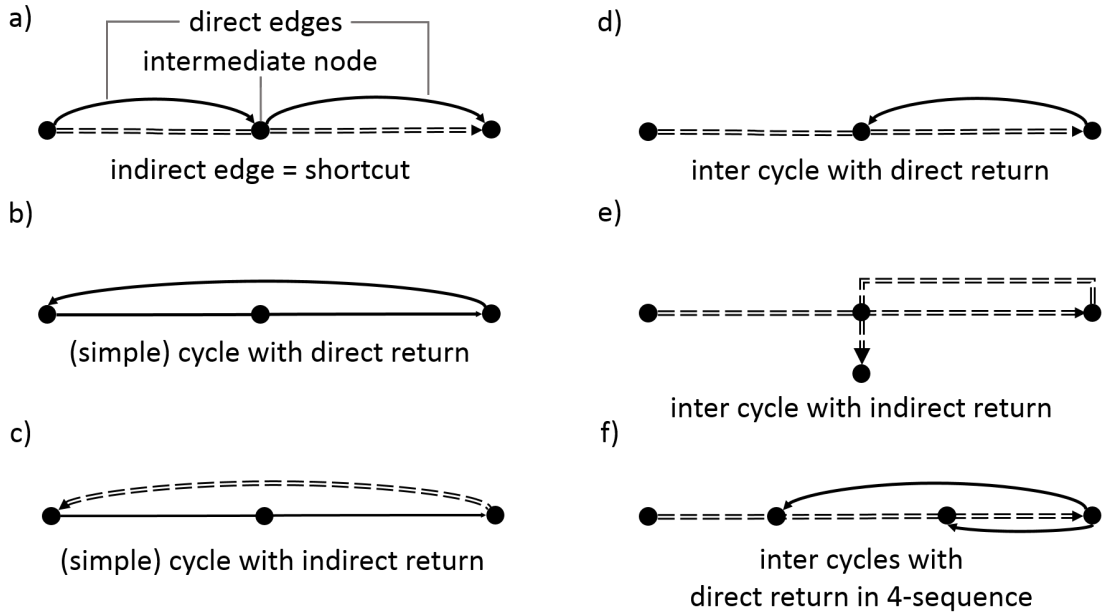


Figure 5.6: Illustration of the introduced terminology (a)), of cycles with direct and indirect return (b) and c)) and of inter cycles with direct and indirect return (d), e) and f)).

where the start/end node is a conventional node or an intermediate node (of an indirect edge). We refer to the former as (simple) cycles *and to the latter as* inter (intermediate) cycles*. Furthermore, we distinguish between cycles where the last edge, i.e., the "returning" edge, is a direct edge and where it is an indirect edge. We refer to these cycles as having* direct return *or* indirect return*, respectively.*

The possible occurrences of these cycles are depicted in Figure 5.6: Figures 5.6 b) and c) show simple cycles with direct an indirect return, respectively. Figures 5.6 d), e) and f) show inter cycles, where the cycle is closed at a node which is not explicitly visited by the routing algorithm, as it is "hidden" by a shortcut.

In the following, we examine which type of cycle may be part of a result path generated by a cost-minimizing routing algorithm and how this affects the result. We first consider simple cycles which require no handling at all, stated by the following lemma.

Lemma 1 *Simple cycles cannot occur in result paths.*

Proof 1 *This lemma is a direct consequence of the Dijkstra property: When visited, every node is reached through the cost-optimal path. Formally, the cost of any path* $p = (\dots, e_{i,j}, \dots, e_{i,k}, \dots)$ *will never be less than that of* $p' = (\dots, e_{i,k}, \dots)$ *as all edge costs are non-negative, i.e.,* $c_{i,k} \leq c_{i,j} + \dots + c_{i,k}$.

Next, consider inter cycles with direct and indirect return, as illustrated in Figures 5.6 d), e) and f). Let us first consider inter cycles with indirect return. Clearly, such cycles imply a detour. Yet, this detour may be compensated by the increased popularity incurred by a valuable sequence of POIs. The gain of visiting the sequence (P_i, P_j, P_k) may justify revisiting P_j, reflected in a sufficiently significant score-discounted cost. However, this is only valid for inter cycles with indirect return, not for those with direct return. This is because inter cycles with direct return offer no additional gain for revisiting a POI. Figuratively speaking, if the gain of the sequence has been collected, returning to a previously visited POI bears no gain. Hence, we want to ensure that inter cycles with direct return cannot occur when employing a routing algorithm on a POI graph with triples $\mathrm{G}^3_{\mathrm{POI}}$. If $\mathrm{G}^3_{\mathrm{POI}}$ fulfills a specific requirement, we can prove this property. The left side of Figure 5.7 exemplifies the occurrence and illustrates notation.

Lemma 2 *Let G^3_{POI} be a POI graph for pairs and triples of POIs. If for every indirect edge (P_i, P_j, P_k) holds:*

$$\delta_{ijk} < c_{jk} + c_{kj} \qquad (\star)$$

then no result path has inter cycles with direct return. δ denotes the additional discount of the POI triple over the concatenation of the two pairs, i.e., $\delta_{ijk} := c_{ij} + c_{jk} - c_{ijk}$.

Proof 2 *We first prove the statement for cycles of the form $(P_i, P_j, P_k), (P_k, P_j)$. Assume, there was an inter cycle with direct return:*

$$c_{ijk} + c_{kj} \leq c_{ij}$$

By the property $(\star)$ we have:

$$\begin{aligned} c_{ij} &= c_{ij} + c_{jk} + c_{kj} - c_{jk} - c_{kj} \\ &\underset{(\star)}{<} c_{ij} + c_{jk} + c_{kj} - \delta \\ &= c_{ijk} + c_{kj} \end{aligned}$$

which is a contradiction to the assumption. Hence, there cannot exist an inter cycle with direct return in a result path. For any longer cycles $(P_i, P_j, P_k), \ldots, (P_k, P_j)$ the statement holds because edge-costs are non-negative.

Note that if one of the POI pairs is not connected, the property is fulfilled trivially. If (P_i, P_j) are not connected, the property is always fulfilled, if (P_k, P_j) are not connected, then no direct return to an intermediate node is possible. It should be noted that in the data set for our experiments with triples of POIs, the property was rarely violated, and enforcing it bears minor computational overhead.

In the following, we extend the lemma to sequences of POIs. The problem with sequences of n POIs is the possible direct return to any of its $n-2$ intermediate node. For a sequence of four POIs this is illustrated on the right of Figure 5.7. Lemma 2 forbids the direct return to the last intermediate node, i.e., the $n-1$-st POI. In order to also forbid

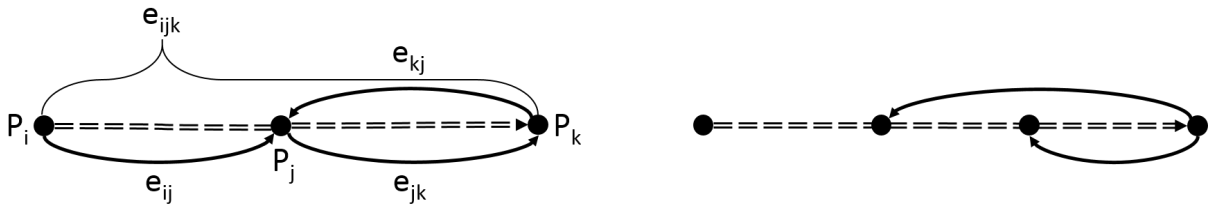


Figure 5.7: Illustration of an inter cycle with direct return in a triple of POIs (left). Visualization of both possible inter cycles with direct return in a 4-sequence of POIs.

the other direct returns, we have to formulate similar conditions for all other intermediate nodes. With increasing length of sequences, however, it becomes less likely that cases occur where these conditions have to be enforced. Also, the conditions can be checked when constructing the POI graph, i.e., only once during a preprocessing phase of graph extraction.

For this purpose, we introduce the following new notation. Sequences of POIs will be indexed by numbers instead of letters. Additionally, by $\delta_{r,s},\ r > s$, we denote the difference in cost between the s-prefix sequence of an r-sequence. For instance, for a sequence of four POIs (P_1, P_2, P_3, P_4), $\delta_{4,2} = c_{12} - c_{1234}$ or $\delta_{4,3} = c_{123} - c_{1234}$. Generally, $\delta_{r,s} := c_{1,\dots,s} - c_{1,\dots,r}$. Note that these values are not necessarily positive. If they are, this implies a high discount of the full sequence compared to the prefix sequence. Obviously, since all edges, direct or indirect, have a non-negative cost, the cost of visiting additional POIs increases monotonically – despite any discount. Hence, the greater the interval between s and r, the less likely the value is positive. Bearing this in mind, we now extend the above statement inductively. This means, for any POI graph with sequences of POIs up to length r, we assume the statement for sequences up to length $r-1$ holds.

Lemma 3 *Let G^r_{POI} be a POI graph for sequences of POIs up to length r. If for every indirect edge $(P_1, \dots, P_r)$ the following statments hold*

$$\begin{aligned}
\delta_{r,r-1} &< c_{r,r-1} \\
\delta_{r,r-2} &< c_{r,r-2} \\
&\vdots \\
\delta_{r,3} &< c_{r,3} \\
\delta_{r,2} &< c_{r,2}
\end{aligned}$$

then no result path has inter cycles with direct return.

Proof 3 *As before, it suffices to prove the statement for cycles of the form*

$$(P_1, \dots, P_s, \dots, P_r), (P_r, P_s)$$

Figure 5.8: Visualization of the eleven hotspot centers in Paris, France.

which directly follows from the above:

$$c_{1,\dots,r} + c_{r,s} > c_{1,\dots,s}$$

Any inter cycle with direct return to POI P_s has greater cost than the shortcut of length s, $e_{1,\dots,s}$. Like before, the cycle cannot be part of a result path as it has greater cost than the cycle-free counterpart. The argument holds analoguously for longer cycles.

In the following, we assume POI graphs for triples and longer sequences to fulfill the above properties. Therefore, it is ensured that we do not produce result paths with cycles (with direct return). For routing purposes, we may use Algorithm 1, replacing G^2_{POI} by G^3_{POI} or G^r_{POI} dependent on the POI graph at hand. For a given user query, i.e., start and target nodes, close entry and exit POIs are retrieved. From start to entry POI and from exit POI to target, the cost-optimal paths in the underlying road network are computed. Between entry and exit POIs, routing is executed in G^3_{POI} or G^r_{POI}, i.e., hopping along pairs, triples or longer sequences of POIs. Finally, the three subpaths are concatenated yielding a path from start to target.

This concludes the section on enrichment of road networks with knowledge mined from crowdsourced data. In the next section, we investigate the whole pipeline, from data processing over knowledge extraction to network enrichment, experimentally. We examine different data types, different data sets and different means of enrichment.

5.6 Experimental Evaluation

In this section, we investigate the effect and impact of the network enrichment with crowd-sourced data. We compare different data sources as described in Section 5.3, different means of extracting knowledge as described in Section 5.4 as well as different methods for enrichment as described in Section 5.5. To be able to draw comparisons between the different setups, we locate all our experiments in the city of Paris, France, which has high data density for all our sources. For road network data, we use OpenStreetMap. The road network extracted from the raw data has about 1M nodes and around 1.8M edges. All language processing was implemented in Python, the modeling of pairwise occurrences was implemented in Matlab. The tasks of network enrichment and path computation were conducted in the Java-based framework MARiO [68] on an Intel(R) Core(TM) i7-3770 CPU at 3.40GHz and 32 GB RAM running Linux (64 bit).

First, in Section 5.6.1, we explore the varying effects when using different data sources. For this, shortest paths in networks enriched with different data mined from single and pairwise occurrences are examined. More precisely, we compare shortest paths in different G^1 and G^2 graphs with each other. Second, in Section 5.6.2, we examine the value of routing in a meta-network compared to routing in an enriched road network graph. For this, paths generated in an enriched network G^2 are compared to those generated within a POI graph G^2_{POI}. Third, in Section 5.6.3, we substantiate the value of sequences of POIs over pairs of POIs. For this, paths generated in a POI graph for pairs G^2_{POI} are compared to those in a POI graph for triples G^3_{POI}. The roadmap for our experiments in summarized in Table 5.1.

For all experiments, we rely on three data sources from which we extract different notions of popularity. First, a Flickr data set, provided by authors of [133], consisting of 14M photos worldwide and over 40K photos in the metropolitan area of Paris, France. Second, a Foursquare data set, provided by the authors of [200, 201], consisting of 33M check-ins by over 250K users at more than 3.5M venues worldwide, thereof 7.6K venues in the area of Paris. Third, an extract from the aforementioned textual data set extracted from travel blogs. This extract contains 200 significant POIs and 2K occurrences of closeness relations (see Section 5.4.2) in the area of Paris. Using Twitter's public feed, we obtain around 200K tweets regarding these POIs. Which kind of knowledge is extracted and how the network is enriched with this knowledge, is described in the respective sections. Table 5.2 gives an overview which type of (POI) graph is derived from the different sources. Note that this selection is not exhaustive, for instance, one could use consecutive photos of Flickr users to derive scenic spots which are spatially connected.

Evidently, the density of the data sets varies considerably. Therefore we restrict ourself to single, pairwise and triple occurrences of POIs and omit longer sequences. In order to obtain meaningful results, we empirically determine eleven hotspot regions in Paris where all data sets are particularly dense. A hotspot is a circular region with a 500 meter radius centered such that it contains significantly many data points of all sources. The hotspot centers are visualized in Figure 5.8. Covered by the entirety of the eleven hotspots are over 5K nodes of the underlying road network. For a query, two random nodes are drawn

from this set as start and target. The results of each setting are grouped in two distance brackets corresponding to the Euclidean distance between start and target, 0 to 4 and 4 to 8 kilometers. For each setting, 6K runs are executed.

In the following, we compute paths in different graphs and compare them w.r.t. the respective scores they attain. We define three scores which are related to the scores used for enrichment in Section 5.5.1 but not identical. For a given path p, the absolute Flickr score S_{FLR} corresponds to the summed number of Flickr photos within a 40 meter radius of the course of p. Analogously, the absolute Foursquare score S_{FSQ} is the total number of check-ins at POIs within a 40 meter of the course of p. Finally, the absolute textual sentiment score S_{TXT+SA} is the total number positive mentions of all POIs within a 40 meter radius of the route of p. Again, Table 5.1 gives an overview of the measures used in the experiments.

	Data Sources Sect. 5.6.1		Networks Sect. 5.6.2		Sequences Sect. 5.6.3
	Single	Pair	Foursquare	Text.Closeness	
Graphs	G^1(FLR) G^1(FSQ) G^1(TXT+SA)	G^2(FSQ) G^2(TXT+CR)	G^2(FSQ) G^2_{POI}(FSQ)	G^2(TXT+CR) G^2_{POI}(TXT+CR)	G^2_{POI}(FSQ) G^3_{POI}(FSQ)
Measures	PL S_{FLR} S_{FSQ}	PL S_{TXT+SA} S_{FSQ}	PL S_{FSQ}	PL S_{TXT+SA}	PL S_{FSQ}

Table 5.1: Overview of the experiments conducted, the graphs used and the measures employed. PL stands for path length. Lengths and score values are always relative to those of the conventional shortest path (in G).

Graph type Num. of occurrences	G^1 enriched netw. single	G^2 enriched netw. pairs	G^2_{POI} meta-netw. pairs	G^3_{POI} meta-netw. triples
Flickr (FLR)	✓	✗	✗	✗
Foursquare (FSQ)	✓	✓	✓	✓
Travel Blogs (TXT)	✗	✓	✓	✗

Table 5.2: This table shows which type of graph and POI graph can be derived from the different sources used in this work.

5.6.1 Comparing Data Sources

In this part of our experimental evaluation, we illuminate the effects of enrichment with different data sources. Throughout this chapter, we focus on data sources which reflect

some notion of crowdsourced popularity with differing connotations. Enriching the road network with this kind of information, we aim to generate a graph and paths therein which better reflect the qualitative "mind of the crowd". Finding measures for evaluation of this enrichment, however, is not straightforward. Because, if there existed an absolute measure for popularity, quality or value, this work would be obsolete. Thus, we have to rely on the measure provided by the enrichment with one data source in order to evaluate the enrichment with other data sources. For instance, consider a single-criterion road network enriched with two notions of popularity, e.g., aesthetic appeal, as for example provided by the crowdsourced knowledge of Flickr photos, and (nightlife or culinary) trendiness, as for example provided by the crowdsourced knowledge of Foursquare check-ins. Given start and target, we may now compute three different cost-optimal paths. The cost-optimal path w.r.t. the network cost criterion and the two cost-optimal paths w.r.t. the score-discounted cost functions as described in Section 5.5.1. In lack of an objective measure for the qualitative information of our enriched network, we evaluate each path w.r.t. the score provided by the other data source. More precisely, for the optimal path reflecting aesthetic appeal, we compute the score reflecting trendiness and vice versa. We hope to observe an increase of popularity scores for any path computed within an enriched network, compared across different enrichment sources and, particulary, compared to the conventional cost-optimal path w.r.t. network criterion. We compare paths computed in enriched graphs G^1 and G^2.

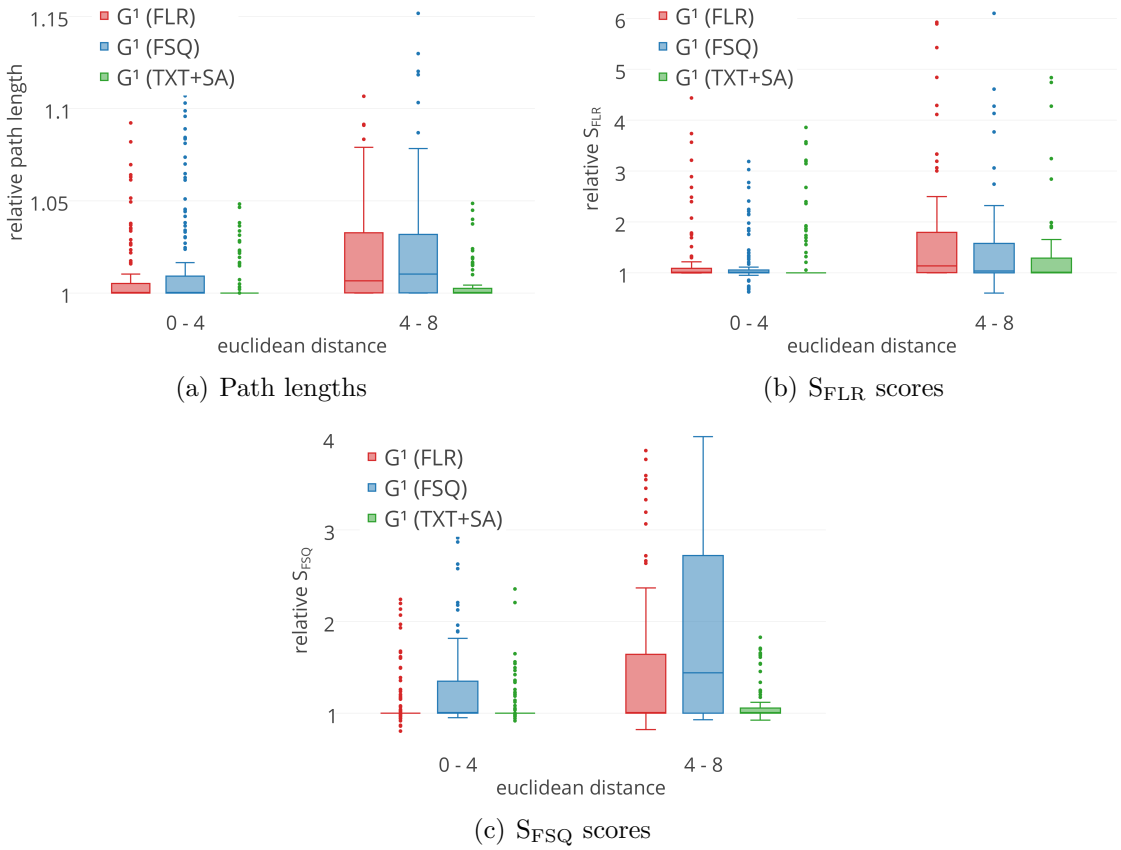
Single Occurrences

For single occurrences of POIs we rely on all three data sources: Flickr, Foursquare and the travel blog entries. From these data sources we derive three score-discounted cost functions and thus three enriched networks, as detailed in Section 5.4.1. We denote the enriched networks by G^1(FLR),G^1(FSQ),G^1(TXT+SA), respectively.

For each experimental setting, we choose one of the scores as evaluation measure. For each run of a setting we choose start and target as described above. For every start and target, we compute the cost-optimal path in all four networks using Dijkstra's algorithm, and for each of the four results, we compute the chosen score function. We group the resulting scores according to the Euclidean distance between start and target in two distance brackets.

The results are shown in Figure 5.9(a) for distance, in Figure 5.9(b) for Flickr score S_{FLR} and in Figure 5.9(c) for Foursquare score S_{FSQ}. The results for the textual sentiment score show a similar behavior and are therefore omitted here. All results are relative to the conventional shortest path in G. The conventional shortest path serves as a baseline for distance as well as for any score function. This is because the conventional shortest path reflects the score inherent in the query itself, the score attained "by chance".

Figure 5.9(a) displays how far the cost-optimal paths in enriched networks stray from the shortest path. Evidently, the increase in path length is marginal, rarely above 5%. The increase in path length is, of course, expected to provide an increase in score. We evaluate the paths generated in each graph w.r.t. to each of the scores. Considering Figure 5.9(b),



(a) Path lengths

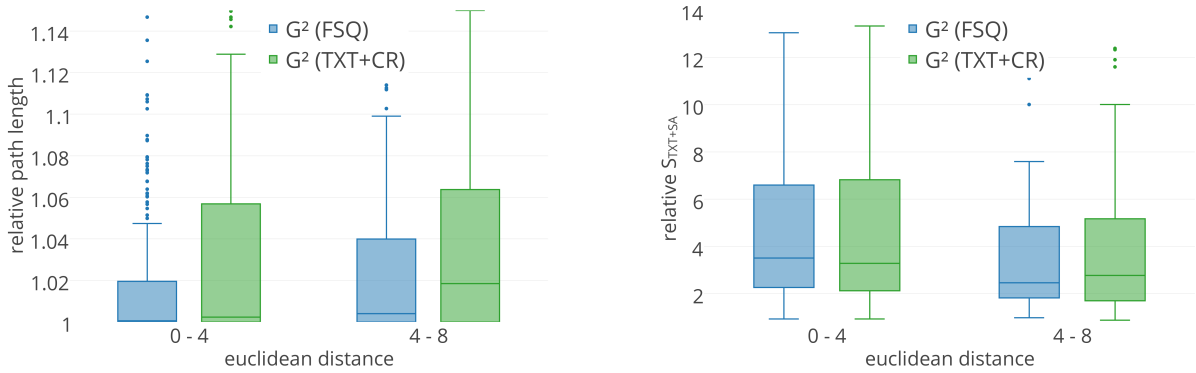
(b) S_{FLR} scores

(c) S_{FSQ} scores

Figure 5.9: Path lenghts and scores of optimal paths in graphs G^1(FLR), G^1(FSQ) and G^1(TXT+SA) relative to path lengths or scores of shortest paths (in G).

unsurprisingly, the highest S_{FLR} is achieved by the optimal path in G^1(FLR), analogously for S_{FSQ} and G^1(FSQ) (see Figure 5.9(c)). Overall, the paths in the enriched networks score around 10% to 20% higher than the shortest path. For longer distances, considerable increase in scores is attained, while for the smaller distance bracket, 0 to 4 kilometers, the increase in scores is merely marginal. This emphasizes the importance of networks enriched with binary or sequential occurences of POIs. As we will see later, scores are boosted when relying on non-single occurrences and POI graphs. Nevertheless, the results shown here can be understood as a proof of concept for our work: Optimal paths in networks which are enriched with crowdsourced data, indeed gain higher scores than simple shortest paths[11]. Also, optimal paths in a network enriched from one data source gain higher scores w.r.t. this source's score than paths in networks enriched with other sources. Furthermore, we observe a certain correlation between the different score values. An increase in score

[11] We note that this effect is independent of the underlying cost criterion. The same holds for other hard metrics like travel time or energy consumption. However, for reasons of brevity, we omit this evaluation here.



(a) Path lengths

(b) S_{FSQ} scores

Figure 5.10: Path lengths and scores of optimal paths in enriched networks G^2(FSQ) and G^2(TXT+CR) relative to score of shortest path (in G).

S_{FLR} implies an increase score S_{FSQ} and vice versa. The score S_{TXT+SA}, however, stays largely unaffected by the increase of both other scores. This could imply that the notion of popularity induced by Flickr and Foursquare data sets are more dual to each other, while the popularity induced by the sentimentally analyzed travel blog mentions is rather orthogonal.

Occasionally, it is possible for optimal paths in enriched networks to undercut the score of the conventional shortest path. This is because the score of enrichment and the score of evaluation differ. For the enrichment, the density of data around nodes within a data source-specific radius is considered. For our experimental evaluation we chose a fixed radius of 40 meters, independent of the data source. Therefore, it is possible that the radius around the course of the shortest path "accidentally" attains a higher score.

Pairwise Occurrences

For pairwise occurrences of POIs we rely on two data sources: Foursquare and the travel blog entries. The knowledge extraction phase follows Section 5.4.2, for the enrichment we follow Section 5.5.1. Specifically, for each Foursquare user, we extract the pairs of consecutive check-ins, resulting in just under 28K pairwise occurrences. For the textual blog entries, we build a probabilistic model and compute the closeness scores based on the spatial closeness relations such as "next to" and "nearby", yielding around 2K significant pairwise occurrences. As detailed before, we may use this knowledge to either enrich the underlying road network directly or to construct a meta-network, the POI graph. For now, we focus on the comparison of different data sources. How the different network types perform, is evaluated in Section 5.6.2. For reasons of clarity, we therefore restrict us in this part to the enriched networks G^2(FSQ), G^2(TXT+CR), the meta-networks G^2_{POI}(FSQ), G^2_{POI}(TXT+CR) are examined later. Dijkstra's algorithm is used to compute cost-optimal paths in the enriched networks.

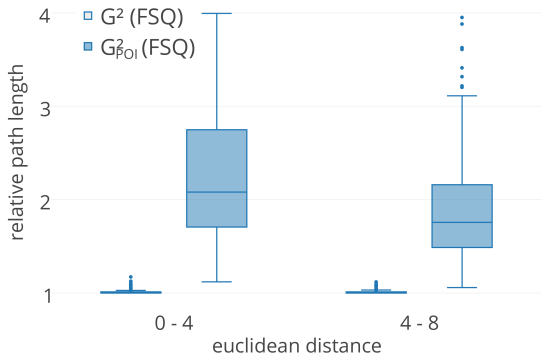
Figures 5.10(a) and 5.10(b) show the results for networks enriched with pairwise occurrences of POIs, evaluated w.r.t. path lengths and scores. All results are relative to the conventional shortest paths in the road network G. Regarding path length, we observe only marginal increase, similar to the networks enriched with single occurrences. However, the increase in scores (here: S_{TXT+SA}) is significantly higher than before, around double to triple the score of the shortest path. Interestingly, the accumulated textual sentiment score S_{TXT+SA} of the paths computed in the Foursquare graph G^2(FSQ) are on a par with those in the textual closeness graph G^2(TXT+CR). This can be attributed to the discrepancy between the score S_{TXT+SA} and the enrichment of G^2(TXT+CR). While S_{TXT+SA} scores positive textual mentions of POIs, the edges of G^2(TXT+CR) are score-discounted and therefore favored if they are part of shortest paths between "close" POIs. The similar score values therefore imply that the textual sentiment score, the notion of POI popularity mined from Tweets, is equally reflected in the enrichment with consecutive Foursquare check-ins and POI pairs which are close according to the crowd. From an application standpoint this can mean that S_{TXT+SA} is insufficiently selective, i.e., it does not qualify well for demarcation. On the other hand, the effect might also be considered a benefit: The knowledge extracted from two different data sources is reflected in one score derived from another data source. Of course, further tests are required to substantiate either claim.

Overall, we may state that different data sources produce different paths. Which knowledge is mined from which data source is application-dependent, correlations occur, and implications are subject to expert supervision. We find that already for networks enriched with single occurrences of POIs, there is an increase in scores at the cost of only marginal increase in length. As we will see, this effect can be amplified considerably when employing networks enriched with pairwise occurrences and, even stronger, when employing POI graphs.

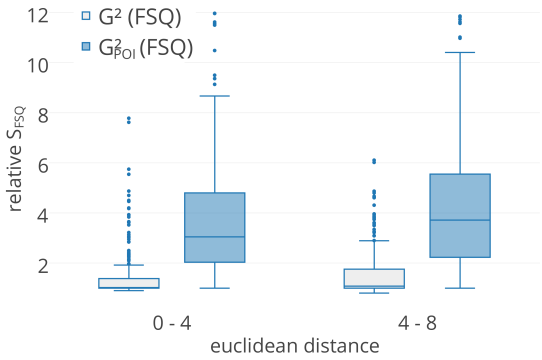
5.6.2 Comparing Enriched Networks and Meta-Networks

Having compared different data sources, we now compare the different network types. For this set of experiments, we use networks enriched with pairwise occurrences, i.e., G^2 versus G^2_{POI}. We rely on the same data sources as before for pairwise occurrences, consecutive Foursquare check-ins and textual POI mentions that stand in closeness relation to each other. From these sources, four graphs are derived, two enriched networks, G^2(FSQ), G^2(TXT+CR) and two meta-networks G^2_{POI}(FSQ), G^2_{POI}(TXT+CR).

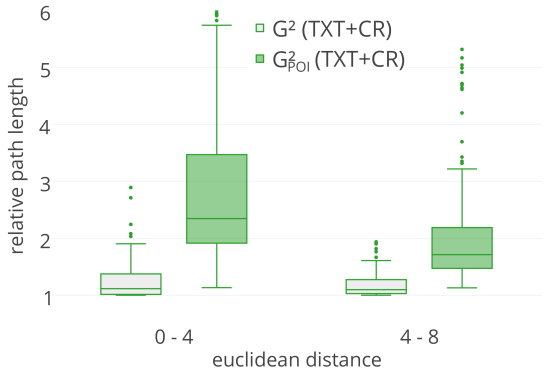
Figures 5.11(a) and 5.11(b) show the results for the Foursquare graphs G^2(FSQ) and G^2_{POI}(FSQ) relative to path lengths and S_{FSQ} scores of the shortest path in G. Figures 5.11(c) and 5.11(d) show the results for the textual closeness graphs G^2(TXT+CR), G^2_{POI}(TXT+CR) relative to path lengths and S_{TXT+SA} of the shortest path. Note that when comparing data sources in Section 5.6.1, we evaluated paths generated in an enriched network (e.g., G^1(FLR)) with the scores induced by other data sources (e.g., S_{FSQ}, S_{TXT+SA}). Now, we compare the paths generated in an enriched network to those in a meta-network. Therefore, we evaluate the paths in one networks (e.g., G^2(FSQ), G^2_{POI}(FSQ)) with the score induced by the same data source (e.g., S_{FSQ}).



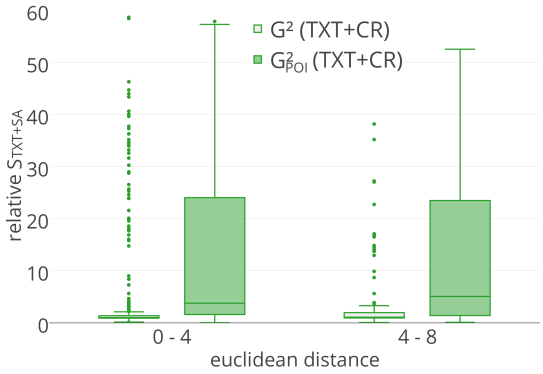
(a) Path lengths in G^2(FSQ) and G^2_{POI}(FSQ)



(b) S_{FSQ} scores in G^2(FSQ) and G^2_{POI}(FSQ)



(c) Path lenghts in G^2(TXT+CR) and G^2_{POI}(TXT+CR)

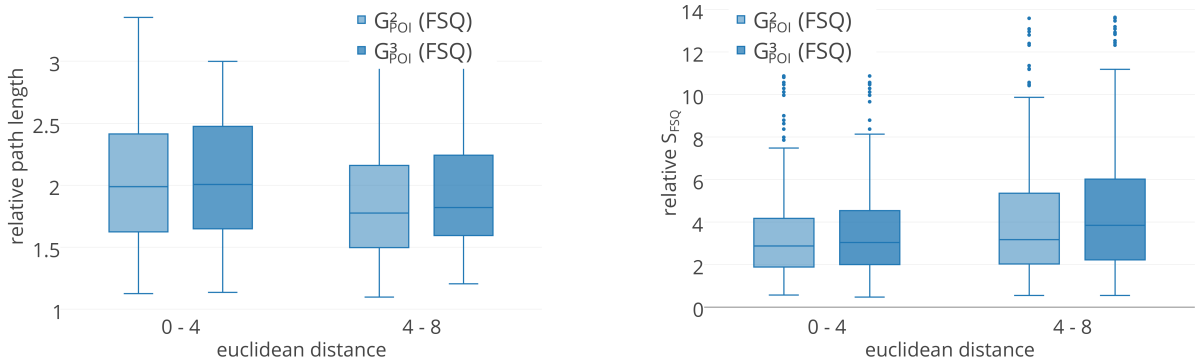


(d) S_{TXT+SA} scores in G^2(TXT+CR) and G^2_{POI}(TXT+CR)

Figure 5.11: Path lengths and scores of optimal paths in Foursquare graphs (top) and textual closeness graphs (bottom) relative to shortest path (in G).

In terms of path length, both enriched networks, G^2(FSQ) and G^2(TXT+CR), create results only slightly longer than the shortest path. Of course, when routing in a POI graph, the paths increase considerably in length. This is due to path computation, which (recalling Algorithm 1) for the most part follows paths between POI pairs and therefore enforces stronger binding to parts of the network which are scored higher. Hence, compared to the shortest path, we observe roughly doubled path lengths in G^2_{POI}(FSQ) and G^2_{POI}(TXT+CR). Interestingly, the increase in length is greater when start and target are closer to each other. This can be attributed to the density of the POI graphs which, compared to the road network, is low. Therefore, the detours which have to be made due to the structure of the POI graph carry less weight as the overall distance grows.

For the marginal increase in length using enriched networks G^2(FSQ) and G^2(TXT+CR), results attain considerably higher scores (see Figures 5.11(b) and 5.11(d)). While path lengths are almost the shortest path distance in G^2(FSQ), S_{FSQ} is around 30% higher than that of the shortest path. For G^2(TXT+CR), even more so. At a length increase



(a) Path lengths (b) S_{FSQ} scores

Figure 5.12: Path lengths and scores of optimal paths in graphs G^2_{POI}(FSQ) and G^3_{POI}(FSQ) relative to shortest path (in G).

of 20%, paths attain roughly double the S_{TXT+SA} score. For both data sources, the POI graphs intensify the effect. By doubling path length, three (G^2_{POI}(FSQ)) to five times (G^2_{POI}(TXT+CR)) the popularity scores of the shortest paths are attained. Of course, this effect is dependent on parametrization and data density, but the results substantiate the claim that routing in POI graphs serves the purpose of increasing data-specific scores. For any application where straying further from the shortest path is unproblematic, meta-networks may be employed. This may hold, for example, for tourist route recommendation systems where path length is a minor criterion. Also, in the case of sportive routing, e.g., for cyclists and runners, path length might be a limiting factor, but attractiveness of the path itself is probably the most important factor. When only minor increase in length is tolerated, then enriched networks yield the better trade-off. For negligible additional path length, considerable gain in score is achieved. This may be of beneficial for car navigation systems. For drivers, travel time is usually the highest ranked criterion. However, some drivers might be willing to accept a minor detour for a more scenic trip, i.e., knowledge that may be extracted from crowdsourced data like Flickr photos, for instance.

5.6.3 Comparing Pairs and Triples

For our final setting, we examine the benefit of enriching meta-networks with sequential POI knowledge. We rely on consecutive Foursquare check-ins for this experiment, i.e. the POI graphs G^2_{POI}(FSQ) and G^3_{POI}(FSQ). As mentioned before, we were able extract just under 28K pairwise occurrences, i.e., consecutive check-ins pair by the same user. Extending the sequence of consecutive check-ins to triples, we were able to mine 14K triple occurrences. Combining pairs and triples of POIs, we may build a POI graph G^3_{POI}(FSQ) which, in addition to the links between POI pairs, holds shortcuts for relevant occurrences of POI triples. As mentioned before, Algorithm 1 may equally be applied to POI graphs

for sequences. The structure of the graph, however, can lead to cycles during path computation. In Section 5.5.2 we detailed which kinds of cycles in result paths are problematic. We explained which are to be avoided (indirect cycles with direct return) and how to ensure this by asserting that a property related to the cost of edges holds (recall Lemma 2). Building $G^3_{POI}(FSQ)$, we encountered violations of this property in a negligible number of triples, < 100. It can therefore be stated that ensuring the correctness of POI graphs for triples is not at all a limiting constraint.

Figure 5.12(a) shows the path lengths, Figure 5.12(b) shows the Foursquare scores S_{FSQ}. Additionally using triples of POIs has hardly any increasing effect on path lengths. This can be attributed to presence of links between pairs as well as shortcuts in $G^3_{POI}(FSQ)$. If a shortcut is not beneficial, i.e., its trade-off between length and score is suboptimal, the routing algorithm may often also choose the path corresponding to one of the pairs. Thus, the POI graph for triples offers additional knowledge without degrading the knowledge extracted from pairwise occurrences. The increase in scores of paths generated in $G^3_{POI}(FSQ)$ compared to those in $G^2_{POI}(FSQ)$ is not as strong as between $G^2_{POI}(FSQ)$ and $G^2(FSQ)$. Nevertheless, at the cost of virtually no increase in path length, we are able to attain 15% to 30% higher scores. This emphasizes the value of POI sequences for routing purposes. Seeing as the overhead in programming is low, it can be recommended to integrate this kind of knowledge if it may be extracted from the corresponding data source.

5.7 Conclusions

In this work we presented new approaches to computing knowledge-enriched paths within road networks. We incorporated novel methods to extract spatial relations between pairs of Points of Interest such as “near” or “close by” from crowdsourced textual data, namely travel blogs. We quantified the extracted relations using probabilistic models to handle the inherent uncertainty of user-generated content. Based on these models, we proposed a new cost function to enrich real world road networks, based on Djikstra and skyline path computation. The new cost function reflects the closeness aspect according to the crowd. In contrast to existing approaches, we did not enrich previously computed paths with semantical information, but the entire network. Continuingly, two routing algorithms were presented taking this closeness aspect into account. Finally, we evaluated our ideas on two real world road network datasets, i.e., Paris, France, and New York City, USA. We used metadata from geotagged Flickr photos as a ground truth to support our initial goal of providing more popular paths. All our approaches performed very well by providing slightly longer paths but with significantly higher values of popularity.

For future work, we are researching alternative methods for aggregating all categories of spatial relations. Furthermore, we would like to investigate ways to suggest the popular path descriptions to the user based on the Points of Interest they will encounter underway.

Chapter 6

Parking Queries on Road Network Data

6.1 Introduction

With increasing gas prices, escalating greenhouse gas emissions and heavy traffic congestion in metropolitan areas, optimizing traffic is of great ecological and social importance. While the basic routing task of finding a path from start to target is a well-explored research area, there are other routing tasks common in everyday life which have drawn less attention so far. An example are trip planning queries (TPQ) which are specified by a start location, a target location and a number of resource types which have to be visited along the trip. For instance, the user might provide the resource types "ATM", "gas station", and "department store". The result of such a query is the shortest path from start to target visiting exactly one instance of each resource type. There are several variants to this kind of problem which will be reviewed in Section 6.2.

A problem relevant to everyday life is guiding a user through a road network to enable them to find a resource for which the availability at certain locations is uncertain. There are several examples for this task. For instance, consider parking spots: although it might be known that certain streets allow parking, it is generally not known whether there will be any vacant spots upon arrival. Another example are drivers of electric cars looking for public charging stations. In some developing countries hospitals with emergency capacities are scarce. Thus, ambulances often visit more than one hospital before being able to hospitalize their patient. In all of these cases, it holds that if the resource is not available upon arrival, the search must be continued to other resource locations until an available resource is found. Hence, guiding a user to the closest resource does not yield a satisfactory solution. Instead, in order to yield a sufficiently large probability of success, a route visiting several resource locations is required. A general problem of finding such routes is that there are two contrary characteristics describing the quality of a route. The first quality measure is the overall likelihood of finding an available resource when following the route. The second quality measure is the cost, e.g., the expected travel time or distance, until the

respective resource is found. These two measures are complementary, because continuing the search to another resource location will always increase the success probability but also – without exception – the cost. Thus, it is not possible to minimize the cost while maximizing the probability of success.

In order to compute the success probability of a route, it is necessary to employ information about the availability of resources at all known resource locations. In this chapter, we assume a system which collects observations on the availability (and conversely the consumption) of resources over time. These so-called long term observations are then used to compute probability distributions modeling general resource availability. Furthermore, the system provides current information about resource status at query time. We refer to this kind of information as short term observations. For example, long term observations correspond to the average time a parking spot remains vacant or occupied. Short term observations, on the other hand, provide information about currently vacant spots. Depending on the scenario the amount of short term observations might be limited, e.g., only a limited number of parking spots are equipped with occupancy sensors while the rest is detected and reported by other roaming cars. At first glance, short term observations might seem sufficient for a successful search. However, knowing that a resource is available at the moment, does not mean that it still will be upon arrival. Thus, as time progresses, the influence of short term observations on the probability of finding a resource decays. Long term observations address this problem in three ways. First, if there is no short term observation available for a resource, the expected vacancy time of a resource can compensate for the lack of current information. Furthermore, long term observations can be used to predict the probability that a currently available resource will still be vacant upon arrival. And, finally, long term observations can serve as an estimate for the expected occupancy time, i.e., the expected time until a consumed resource becomes available again. For example, in the parking spot scenario this corresponds to the expected parking duration.

In this chapter, we present a statistical model describing a road network comprising resource locations of a specific resource type. Our model incorporates both types of information described above, i.e., long term as well as short term observations. From a formal point of view, our model describes each resource location as a continuous-time Markov chain with two states, `available` and `consumed`. Based on this model, we introduce the following query: For a given query location, compute a route for which the probability of finding an available resource exceeds a given probability threshold (e.g., 90 %) while minimizing the (expected) cost, e.g., travel time or distance. A route may be extended infinitely, and each extension adds to the success probability but also to its cost. Thus, in order to optimize one measure, we have to bound the other. More precisely, the best route may be the one with the least cost among all routes exceeding the probability threshold. Or, converesely, the best route may be the one with the highest success probability among all routes not exceeding a cost threshold.

Since our model allows for reapparance of resources, the search space of possible solutions is unlimited. However, in many applications the time frame of finding a suitable answer is rather small as users only tolerate limited answering time. Therefore, we investigate two greedy search heuristics which promise admissible results in efficient time. To

allow the computation of optimal results, we examine a recursive backtracking approach to avoid exhaustive search. Furthermore, we propose a lower bound for the remaining increase in cost used in a highly efficient branch and bound solution providing optimal results. We evaluate our approach within real world road networks on the application of finding parking spots and on the application of finding charging stations for electric vehicles. To conclude, the contributions of this work are as follows:

- A novel probabilistc model describing the availability of resources in road networks. We model resources as continuous-time Markov chains which are parametrized by long term as well as short term observations and allow to model the reapparance of previously consumed resources.
- A new type of query, the Probabilistic Resource Route Query with Reappearance (PRRQR).
- An approximate solution to the PRRQR employing two different search heuristics, as well as optimal solutions based on backtracking and branch and bound.

The rest of the chapter is organized as follows: Section 6.2 summarizes related work about similar types of queries. In Section 6.3, our probabilistic model is described, followed by the formal definition of the PRRQR. Section 6.5 describes the heuristics, bounds and query algorithms introduced to process PRRQRs. The results of our experimental evaluation are presented in Section 6.6. The chapter is concluded with a summary and an outlook for future work in Section 6.7.

6.2 Related Work

In this section, we survey existing work on similar tasks. First, we will give a review of basic query types related to the one introduced in this work. Then, we address works which model the existence of resources in a probabilistic way. All of the following query types have the same meta-task, namely, guiding a user to a certain resource. In all of these scenarios, a database which stores the resources and their respective locations is assumed. Although this task may also be carried out in Euclidean space, we restrict ourselves to road networks, as most of the applications are traffic-related.

The simplest type of query guiding a user to the next available resource is the nearest neighbor query (NNQ). In this setting, the user specifies a location – typically his current location – as well as some type of resource. The result of the NNQ is the optimal path (fastest, shortest, etc.) to the closest location providing the resource. As NNQ are a well-explored research area, we will not go into detail on their solutions. An extension of this query are trip planning queries (TPQ) [118] (also referred to as route planning queries [31] or route search queries [101]). In this problem setting, the user specifies a selection of different resources, e.g., “ATM”, “restaurant”, “florist”, “cinema”. Additionally to his start location the user may specify a target location. The result of a TPQ is the optimal

path from start to target visiting at least one instance of each resource. Computing TPQs is NP-complete because in the case that each specified resource type occurs exactly once the TPQ degenerates to the Traveling Salesman Problem (TSP).

In another variation of the problem, the order in which resources are visited may be constrained, as described in [163] or [102]. For instance, if planning a date, the order of resources might be restricted by the constraints that the ATM has to be visited first and the florist should be visited before the restaurant and the cinema. However, since the task usually maintains an NP-hard subproblem, solutions to any of these problems typically employ heuristics ([31, 163]).

In the settings presented so far, the existence of a resource at its location is considered to be guaranteed. In many real world applications, however, a resource will only be available with a certain probability. If no table or seats have been reserved, not all locations of type "restaurant" or "cinema" might have the resource available. The same holds for the resource type "florist" if looking for specific flowers. In all of these cases, the requested resource is available with a certain probability (and consumed with the converse probability), i.e., prior to arrival it is not known with certainty whether the resource is available or consumed. At first glance, these kinds of uncertainty may seem congruent. However, there are significant differences which require specific modeling. We distinguish the following types of uncertainty.

Assume a surfer is looking for good waves and is considering different beaches. At every beach, the waves might be sufficient with a certain probability. If available, the resource "waves" cannot be consumed by the presence of other surfers. Thus, we refer to the probability of finding waves as *static (resource) uncertainty*. This uncertainty is independent from the time of arrival and the presence of competitors.

Now, consider a cinema where seats are a limited resource. If no seats have been reserved, the probability of finding seats for a particular show decays as screening time approaches. Since in this case a limited quantity of the resource is consumed over time, we refer to this type of uncertainty as *time-decaying (resource) uncertainty*. Contrastingly, while all tables at a certain restaurant might be occupied now, there might be one available later the same evening. In this case, the quantity of the resource might decay (or more generally: change) over time but regenerate at a later point in time. This is a significant difference to the other scenarios where revisiting resources does not yield any benefit. However, in this scenario, although the resource might have been consumed upon arrival, it might make sense to revisit after an adequate waiting time. We refer to this kind of uncertainty as *time-dependent (resource) uncertainty with reappearance*.

To the best of our knowledge, there is no previous work supporting short term observations or taking time-dependent uncertainty with reappearance into account. Therefore, we shall now review works which incorporate static as well as time-decaying resource uncertainty. While we provide an abstract problem definition (cf. Section 6.3) and applications based upon this definition, some works focus on their application and adapt their problem definition to the respective use case. Nevertheless, these works can – with some restrictions – in many cases be extended to incorporate general resources.

The authors of [101], for example, compute paths which guide the user along certain

resources. In their follow-up work, [102], this problem is extended by ordering constraints (as in some of the examples above). Both papers present algorithms based on greedy search heuristics as well as on heuristics which minimize the expected distance until search success. In both papers, resources may have assigned success probabilities. However, these probabilities reflect static uncertainty, i.e., non-time-dependent and non-reappearing. The same holds for [51] and its follow-up work [103] where probabilistic k-route queries are introduced and examined. Here, the authors introduce a confidence value which corresponds to the existence probability of a resource at each location, also reflecting static uncertainty. Employing different heuristics, the presented algorithms find approximate solutions by clustering resource locations that maximize the expected success.

In contrast, the authors of [47] take time-dependency into account and assume a linear decay for the vacancy of parking spots. Although this model is aimed at a specific application, it may be generalized to abstract resources. Note, however, that this model does not allow reappearance of resources which is a significant shortcoming, especially in this application. This is because a typical strategy looking for a parking spot is roaming the target area until someone vacates a spot, i.e., a resource reappears. The authors propose two approaches to maximizing the probability of finding a parking spot. The first approach finds the optimal result, however, this is done employing full enumeration on the time-varying TSP. Due to the brute force nature of this algorithm, query processing quickly becomes infeasible with increasing number of resource locations. The second approach is an algorithm that clusters resource locations before solving a TSP on the clusters. Subsequently, the optimal solution within each cluster is searched. Based on a heuristic, this algorithm yields an approximation of the optimal result, while providing a considerable speed-up.

There are various methods, focusing on the application of taxi pickups as well as ridesharing, such as [132, 156, 129, 67]. In [132], the task is equivalent to solving a classic TSP, i.e., ordering different but fixed pickup locations such that the total distance is minimized. The authors rely on a genetic algorithm approach to solve this problem. In [156],[67] and [129] the task is more complicated. Here, the task is first of all to assign cabs to a set of currently available customers. After this assignment is done, a route for picking up and dropping off the customers has to be found. Thus, only the last part of the query is related to our work. Furthermore, none of the works considers uncertainty w.r.t. customer availability.

6.3 Problem Setting

In this section, we formalize our problem setting. First, we define the graph which represents the underlying road network. Then, we introduce the probabilistic model which describes resource availability and consumption.

6.3.1 Road Network Graph

For a given road network, we let $G = (V, E)$ denote the corresponding graph, i.e., the vertices (or nodes) $v \in V$ correspond to crossings, dead ends, etc., and the edges $e \in E \subseteq V \times V$ represent directed road segments connecting the vertices. We refer to this graph as *Road Network Graph*. Furthermore, let $c : E \to \mathbb{R}_0^+$ denote the function which maps every edge onto its respective cost, e.g., travel time or distance. If the employed cost function is not travel time, we additionally assume the travel time to be known and given by a function $t : E \to \mathbb{R}_0^+$ (as resource availability is dependent on the time of arrival). By *route*, we mean a consecutive set of edges (possibly with cycles), i.e., $r = (e_1, \ldots, e_n)$ where all e_i are taken from the corresponding set of edges and for all $1 \leq j \leq n-1 : e_i = (u, v) \Rightarrow e_{i+1} = (v, w)$. The cost of a route $r = (e_1, \ldots, e_n)$ is defined as the accumulated cost of its edges, i.e., $c(r) = \sum_{i=1}^{n} c(e_i)$. By *path*, we mean a cycle-free route.

6.3.2 Probabilistic Model

In the following, we introduce our probabilistic model. As mentioned before, at every resource location the respective resource may either be `available` or `consumed`. However, prior to arrival at the location, it is not known which of the two is the case. Our probabilistic model has to be able to reflect all three kinds of uncertainty: static, time-decaying, and time-dependent uncertainty with reappearance. While the former two kinds of uncertainty are rather straightforward, the latter requires more work and a novel approach.

The static uncertainty of a resource is easily expressed by a random variable X which takes the values 0 or 1, representing the states `available` and `consumed`, respectively, where the probabilities of X are $\mathbb{P}(X = 0) = p$ and $\mathbb{P}(X = 1) = 1 - p$ for some $p \in [0, 1]$. For illustration, recall the example of a surfer looking for waves at a beach. Independent of the time of their arrival there will be waves with probability p.

In the case of time-decaying uncertainty, we propose modeling the probability that a resource X is available at time $t > 0$ as $e^{-\lambda t}$ for some $\lambda > 0$. Consequently, at time $t = 0$, the resource is available with probability 1, but it decreases as time progresses and asymptotically approaches 0. Or, in other words, the probability that X is consumed is the cumulative distribution function of an λ-exponentially distributed random variable. This coincides with the intuition of modeling waiting times as exponentially distributed random processes. For illustration, recall the example of buying tickets to the movies, where the probability of available seats is 1 at $t = 0$ but decreases as screening time approaches.

Now, let us turn to the case of time-dependent uncertainty with reappearance which is the core of this work, as it is the only concept that can model the use cases of our experiments (parking spots, charging stations). As before, resource availability has two states, but now, there may occur multiple state transitions at arbitrary points in time. Therefore, we propose modeling each resource as a stochastic process. The most common type of stochastic processes are Markov chains which model the transition probabilities within a system with a given number of states. When a system transitions from one state into another, the future state is only dependent on the present state. This property is

central to Markov chains and referred to as memorylessness or Markov property. Markov chains can either assume discrete or – as in our case – continuous time. In a discrete model, there exist equal time steps, and for each step, the probability of transitioning into another state can be computed. In a continuous-time model, the sojourn time in each state, i.e., the time until the next state transition, is perceived as a random variable itself. The notion of memorylessness extends naturally to the case of continuous time.

Thus, we model time-dependent resource availability with reappearance at each resource location as a continuous-time Markov chain (CTMC). More precisely, each resource location r^i is now represented by a family of random variables $\{X_t^i, t \geq 0\}$ with values in the state set $\{0, 1\}$. Note that there exists a one-to-one relationship between each resource location, and its resource modeled by the respective CTMC. Thus, we denote the CTMC of each resource location r^i by X^i and denote the set of all CTMCs by $\mathcal{X}$, where $|\mathcal{X}| = |\mathcal{R}|$. We may use the term *resource* for the geolocation associated with a resource as well as for the corresponding CTMC. When not clear from the context, we will state explicitly which of the two is referred to. Also, we assume the resources, more specifically their CTMCs, to be mutually independent. The independence assumption keeps the model general and applicable even if the available observations are limited.

Besides its state space, a CTMC is (under the reasonable assumption of time-homogeneity) defined by the family of transition matrices $\{P_t, t \geq 0\}$ and the (infinitesimal) generator matrix Q. Using the Kolmogorow equations, each may be computed from the other by solving the first order differential equation $P'(t) = P(t)Q$. For more mathematical details, we refer the reader to [138]. We omit the explicit calculations here and restrict ourselves to explaining the connection between $P(t), Q$ and the states of a resource.

In our case, Q is a 2×2-matrix. Its diagonal entries reflect the parameters of the random variables modeling the sojourn time of each state while the non-diagonal entries reflect the rate of transition into another state. Q has the following form:

$$Q = \begin{pmatrix} -\lambda & \lambda \\ \mu & -\mu \end{pmatrix}$$

The family of transition matrices $P(t)$ for $t \geq 0$ is defined as follows:

$$P(t) = \begin{pmatrix} \frac{\mu}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t} & \frac{\lambda}{\lambda+\mu} - \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t} \\ \frac{\mu}{\lambda+\mu} - \frac{\mu}{\lambda+\mu} e^{-(\lambda+\mu)t} & \frac{\lambda}{\lambda+\mu} + \frac{\mu}{\lambda+\mu} e^{-(\lambda+\mu)t} \end{pmatrix} \tag{6.1}$$

For each resource, the sojourn times of its states `available` and `consumed` are modeled as exponentially distributed random variables with parameters λ and μ, respectively. This, again, coincides with the convention of modeling waiting times as exponentially distributed. The expected value of an exponential distributed random variable $\exp(\phi)$ is $1/\phi$. Thus, the expected sojourn time in the state of availability is $1/\lambda$, and the expected sojourn time in the state of consumption is $1/\mu$. One application on which we evaluate our model in Section 6.6 is finding vacant parking spots. In this use case, $1/\lambda$ would be the expected

vacancy time, analoguously, $1/\mu$ would be the time until an occupied spot becomes vacant again. Of course, these parameters may be different for each resource location if enough observations for an individual estimation are available. Therefore, it is possible for each resource to have distinctly parametrized Q and $P(t)$.

Let us shortly explain how the assumed observations are used for parameter estimation. As mentioned before, our model allows to incorporate short term as well as long term observations. The latter are used for parameter estimation in the following way: Consider a resource X, which has an unknown expected sojourn time in the state `available` but is assumed to be exponentially distributed with parameter λ. Given a number of observations $x = (x_1, \ldots, x_r)$, i.e., exemplary measurements of the time span during which X stays available, we can easily estimate λ using the maximum likelihood estimator. The according likelihood function is given by:

$$L(\lambda) = \prod_{i=1}^{r} \lambda \exp(\lambda x_i) = \lambda^r \exp(-\lambda r \bar{x})$$

where $\bar{x} = 1/r \sum x_i$ denotes the mean of all measurements. Differentiating the logarithmized likelihood function yields the maximum likelihood estimator $\hat{\lambda} = 1/\bar{x}$, which is simply the inverse of the mean value. The parameter μ may of course be estimated analoguously.

Now, let us review some properties of the transition matrices $\{P(t), t \geq 0\}$ central to the model. Given a resource X and its sojourn time parameters λ and μ, we can compute $P(t)$ as in Equation 6.1. If we also have an initial probability distribution based on short term observations of the states of X at time $t_0 = 0$ (denoted as π_0), we can compute the according probability distribution after an arbitrary point in time $t \geq 0$ (denoted as π_t) as follows:

$$\pi_t = \pi_0 P(t)$$

Note that $P(0) = I$ is the identity matrix (cf. Equation 6.1) which means that if no time has passed, the probability distribution of t_0 is still active. For example, if there is an observation at t_0 of X being in state `consumed`, then $\pi_0 = (0, 1)$. The consumption of resource X is certain – but only at this particular point in time. As time progresses, it becomes more likely that the state changes. Therefore, the original probability distribution π_0 (given by the observation) changes. Note that this reflects the notion of reappearance of previously consumed resources. This is expressed in the (exponentially) decaying influence of the second summands in every entry of $P(t)$ (cf. Equation 6.1). Eventually, the original observation becomes obsolete. This can be seen from the convergence of $P(t)$ as $t \to \infty$:

$$\lim_{t \to \infty} P(t) = \begin{pmatrix} \frac{\mu}{\lambda+\mu} & \frac{\lambda}{\lambda+\mu} \\ \frac{\mu}{\lambda+\mu} & \frac{\lambda}{\lambda+\mu} \end{pmatrix}$$

Asymptotically both rows of $P(t)$ are equal. This implies the initial observation has no more influence on the probability distribution of X as $t \to \infty$. For example, whether the initial observation was `consumed`, i.e., $\pi_0 = (0, 1)$, or `available`, i.e., $\pi_0 = (1, 0)$ is without

significance. In any case, $\lim_{t\to\infty} \pi_t$ is the so-called stationary distribution as introduced below.

In our case, a resource X is a finite-state Markov chain where all states communicate and thus X has a unique stationary distribution π [138]. By definition: $\pi P(t) = \pi, \forall t \geq 0$. Solving this system of equations, we get:

$$\pi = (\pi_1, \pi_2) = \begin{pmatrix} \frac{\mu}{\lambda+\mu} & \frac{\lambda}{\lambda+\mu} \end{pmatrix}$$

This is equal to the rows of $\lim_{t\to\infty} P(t)$ which supports the intuition of t having no more influence on the probability distribution. For example, if no observation for X is available, the only unbiased assumption is the stationary distribution since it assumes the respective share of both states w.r.t. λ and μ.

Let us use all of the above in an example related to one of our applications: Let X be a CTMC modeling the vacancy of a parking spot at a certain location. We make the following assumptions on the model: If `available` (meaning vacant), we expect X to be occupied within 5 minutes. This means, the sojourn time of state `available` is a 1/5-exponentially distributed random variable. If X is occupied, we expect the occupant to leave within 20 minutes. Hence, the sojourn time of state `consumed` (meaning occupied) is a 1/20-exponentially distributed random variable. As mentioned before, $P(0) = I$. Let us investigate how $P(t)$ changes as time (non-infinitely) progresses. For instance, after 1 and after 3.5 minutes:

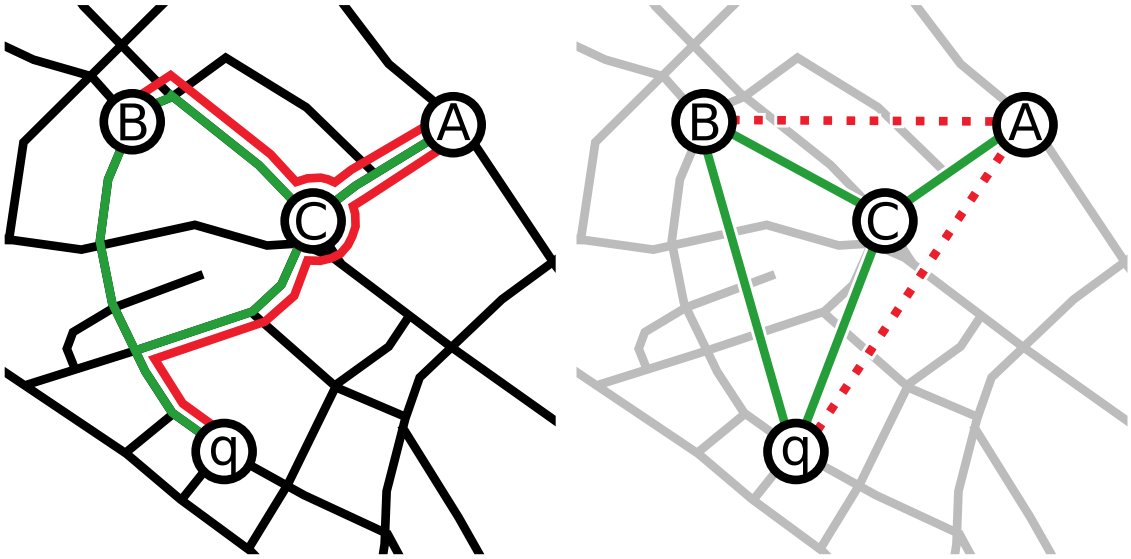
$$P(1) \approx \begin{pmatrix} 0.8 & 0.2 \\ 0.05 & 0.95 \end{pmatrix} \quad P(3.5) \approx \begin{pmatrix} 0.52 & 0.48 \\ 0.12 & 0.88 \end{pmatrix}$$

Assume that at $t_0 = 0$ X has been observed as `available`, i.e., $\mathbb{P}(X_0) = (1, 0)$. Then at $t = 1$ the spot will still be `available` with probability 0.8. After 3.5 minutes this probability will have decreased to 0.52. Now, consider a different scenario where at $t = 0$ X has been observed as `consumed`, then after 1 minute it is `available` with probability 0.05. After 3.5 minutes this probability will have increased to 0.12.

To conclude, we now have a probabilistic model on our hands which is capable of describing all three kinds of resource uncertainty introduced in Section 6.2. The core contribution of this model is its ability to reflect resource reappearance. Also, it allows efficient resource-specific parametrization by incorporating long term and short term observations.

6.4 Query Definition and Result Set

Now that we have defined the probabilistic model, we turn to our query and its result. Both are best described using an alternative graph, referred to as *resource graph* $\hat{G}$. Therefore, in this section, we will first define the resource graph. Subsequently, we introduce two measures which are then used to define the Probabilistic Resource Route Query with Reappearance (PRRQR) and its result.



(a) Optimal direct paths between resources are marked green, fastest paths with intermediate resources are marked red.

(b) Edges of the resource graph (marked continuously green) and excluded transitive connections (marked dashed red).

Figure 6.1: Illustration of a query node q and resources A, B, C in a road network graph (a) and the respective resource graph (b).

6.4.1 Resource Graph

We assume that for a road network graph and a query node q a set of suitable resources $\mathcal{X}(q) = \{X^1, \ldots, X^N\}$ is given. In the majority of applications, only a reasonable subset of resources might qualify. For example, parking spots should be within walking distance of the driver's destination. In this case, the query node would be the driver's position when he reaches the vicinity of his destination. An according range query to a database would then retrieve suitable parking opportunities on which a PRRQR would be executed. For every resource X^i, we assume distribution parameters λ^i, μ^i and an initial distribution π_0^i as given.

The set of vertices of the resource graph is defined as $\hat{V} := \{q\} \cup \mathcal{X}$, where $q \in V$ denotes the query node. The edges of the resource graph, $\hat{E}$, represent cost-optimal paths between resource locations in the underlying road network. Thus, each route in the resource graph can be expanded into a corresponding route in the transportation network (cf. Figure 6.1(a)). Let us note that even in cases where cost does not refer to the travel time, we will also compute the travel time of cost-optimal path because it is needed to for estimating the success probability. Although it is possible to compute the cost-optimal path between each pair of resource locations, we will require $\hat{E}$ to contain only a minimal set of edges by removing transitive connections. A transitive connection is a path within the road network that contains at least one intermediate resource location. For example,

Algorithm 2: Computation of $\hat{E}$

Input: Query setting $\mathcal{X}(q), q$
Output: Edges $\hat{E}$ of resource graph $\hat{G}$

```
1  begin
2      Ê ← ∅
3      foreach X ∈ 𝒳(q) do
4          Compute fastest path p from q to X
5          if no intermediate resource on p then
6              Ê ← Ê ∪ p
7          end
8          Compute multi-target Dijkstra from X to all X' ∈ 𝒳 \ X in G
9          foreach fastest path p from X to X' do
10             if no intermediate resources on p then
11                 Ê ← Ê ∪ p
12             end
13         end
14     end
15 end
```

if along the cost-optimal path from X^A to X^B resource location X^C is encountered, then X^A and X^B would not be connected in the resource graph (cf. Figure 6.1(b)). However, $\hat{G}$ would contain edges connecting X^A and X^C as well as X^C and X^B. We explicitly exclude transitive connections from the resource graph for two reasons. First, transitive links do not allow computing the success probability correctly because the intermediate resource locations are not considered. Second, the existence of transitive connections leads to the inefficient traversal of identical subpaths.

We also compute the cost-optimal paths from the query node q to all resource locations. But as there is no gain in returning to the query node, paths ending at q are excluded from the computation. Algorithm 2 describes the computation of $\hat{E}$ which is illustrated in Figures 6.1. Note that in order to compute $\hat{E}$, we need to compute all cost-optimal paths within the road network graph and then prune the transitive connections. It is not possible to avoid the computation of transitive connections directly.

The cost function of the road network graph G naturally extends to $\hat{G}$. Since every edge in $\hat{G}$ corresponds to an cost-optimal path in G, the cost function $\hat{c} : \hat{E} \to \mathcal{R}_0^+$ maps an edge of $\hat{E}$ to the accumulated costs of the respective path in G. Analogously, $\hat{t} : \hat{E} \to \mathcal{R}_0^+$ maps an edge of $\hat{E}$ to the accumulated time of the respective cost-optimal path in G.

Combining the above, we define the resource graph as $\hat{G} = \hat{G}_{(q,\mathcal{X})} := (\hat{V}, \hat{E}, \hat{c}, \hat{t})$. Note that $\hat{G}$ holds all the query-relevant information since it contains the query node q as well as the resource locations $\mathcal{X}(q)$. Therefore, speaking of a query setting, we mean q and $\mathcal{X}(q)$ as well as the according resource graph $\hat{G}$.

6.4.2 Resource Routes and PRRQR

Relying on $\hat{G}$, we may now define the possible solutions to our query. For a given query setting $q, \mathcal{X}(q)$ and the according resource graph $\hat{G}$, a *resource route* is a route r in $\hat{G}$, starting at query node $q = X^0$. Note that by construction of the resource graph a resource route only follows optimal paths between resources. We describe a resource route as a set of edges (in $\hat{G}$), i.e., $r = (e_{r_1}, \ldots, e_{r_n})$, where $e_{r_i} \in \hat{E}$ for all $1 \leq i \leq n$. Since in our context the order of resources is of particular interest, we introduce a specific notation for it. Recall that every edge in $\hat{E}$ connects two resources unless it starts at the query node. Hence, along a resource route r with n edges we encounter n resources. Note that these resources are not necessarily distinct, as r may contain cycles. Let $X_r = (X^{r_1}, \ldots, X^{r_n})$ denote the n resources along r. To introduce a measure for the success probability of a complete route, we start by defining the success probability of a resource route.

Definition 2 *For a given resource route r along resources $(X^{r_1}, \ldots, X^{r_n})$, let t_i denote the* time of arrival *at X^{r_i}, i.e., $t_0 < t_1 < \cdots < t_n$, and let c_i denote the accumulated cost up to resource X^{r_i}. Furthermore, let $\{P(X^{r_i}, t), t \geq 0\}$ denote the transition matrices of X^{r_i} (dependent on the parameters of X^{r_i}) and let $\pi_0(X^{r_i})$ denote the initial distribution of the states of X^{r_i} (dependent on the availability of short term observations regarding X^{r_i}). Then the probability distribution of X^{r_i} at t_i is defined as:*

$$\big(\mathbb{P}(X^{r_i}_{t_i} = 0), \mathbb{P}(X^{r_i}_{t_i} = 1)\big) := \pi_{t_i}(X^{r_i}) = \pi_0(X^{r_i})P(t_i)$$

Hence, the success probability *of X^{r_i} at arrival time t_i is the probability that resource X^{r_i} is in state* `available` *at time t_i, i.e., $\mathbb{P}(X^{r_i}_{t_i} = 0)$.*

Note that in accordance with the probabilistic model as presented in Section 6.3, we denote state `available` of a resource X by $X = 0$ and the state `consumed` by $X = 1$. Based on this definition, we are able to define the success probability for a complete resource route:

Definition 3 *Let $q, \mathcal{X}(q), \hat{G}$ be a query setting and r be a resource route in $\hat{G}$. The* Success Probability *of r, denoted by $\mathbb{P}_S(r)$, is defined as the probability of the complementary event of not finding any available resource along r:*

$$\mathbb{P}_S(r) := 1 - \left(\prod_{i=1}^{n} \mathbb{P}(X^{r_i}_{t_i} = 1)\right)$$

Given the success probability, we now need to find a second measure which measures the effort of finding an available resource, i.e., the expected cost of a resource route:

Definition 4 *Let $q, \mathcal{X}(q), \hat{G}$ be a query setting and r be a resource route in $\hat{G}$. The* Expected Cost *of r, denoted by $\mathbb{E}_c(r)$, is defined as:*

$$\mathbb{E}_c(r) := \sum_{i=1}^{n} \left(c_i \cdot \mathbb{P}(X^{r_i}_{t_i} = 0) \cdot \prod_{j=1}^{i-1} \mathbb{P}(X^{r_j}_{t_j} = 1) \right)$$

Relying on both measures, we can now define the *Probabilistic Resource Route Query with Reappearance (PRRQR)*:

Definition 5 *Let q be a query node, $\mathcal{X}(q)$ the set of corresponding resources, $\hat{G}$ the according resource graph. Furthermore, let $0 \ll \rho < 1$ denote a probability threshold. The result of a PRRQR with threshold ρ, denoted by $PRRQR(\rho)$, is the resource route in $\hat{G}$ with minimal expected cost among all resource routes which exceed the probability threshold ρ.*

$$PRRQR(\rho) = \arg\min\{\mathbb{E}_c(r) \mid \mathbb{P}_S(r) \geq \rho\}$$

Note that there exists a straightforward variation of the PRRQR. Instead of thresholding the success probability, one could bound the maximal cost. Given a cost threshold $\tau > 0$, the query result is the resource route maximizing the success probability while not exceeding the cost bound τ. In this case, it is usually more reasonable to employ τ as a strict bound on the maximal cost instead as a bounding for the expected cost. For example, consider driving an electric vehicle with a limited remaining range. Bounding the expected distance misses the point, only bounding the actual driven distance (while maxmimizing the success probability) will provide a suitable route. This variation of the query is also examined our experiments. However, in the following, we focus on the first (and more sophisticated) case to keep the description compact.

6.5 Query Processing

In this section, we present algorithms for processing PRRQRs. First, we propose two heuristics which are employed in a greedy search that computes approximations of the optimal results. Afterwards, we will present two methods computing optimal solutions, relying on backtracking as well as on branch and bound. In the following, let $0 < \rho < 1$ be a probability threshold, and let $\hat{G}$ be the resource graph according to a given query setting $q, \mathcal{X}(q)$.

Let us note some aspects central to the PRRQR before going into the details of the algorithms. Given $\hat{G}$ and the query position q, then all resource routes start at q by definition. Hence, the set of possible solutions may be conceived as a search tree rooted at q where each branch is a sequence of resource locations. For a probability-constrained PRRQR with $\rho = 1$, i.e., a PRRQR requiring certainty of finding a resource, this search tree is infinite. The effect is caused by the time-dependent decay inherent in our model. Thus, a certain observation of availability of a particular resource will no longer be certain at the time of arrival. As a result, the success probability of a resource route can only asymptotically converge against 1. Even when $\rho < 1$, the search space is generally very large. This is because considering resource reappearance adds considerably to the complexity of the task. Similar to the Traveling Salesman Problem (TSP), there is no local optimality w.r.t. the resource subsequences that can be exploited. Hence, it is not possible to tell whether a resource route r can be extended into an optimal solution based on its current $\mathbb{P}_S(r)$ and $\mathbb{E}_c(r)$. Furthermore, it is easy to see that all permutations of the set of

resources can be found in the search tree. Thus, from a theoretic point of view the problem is NP-complete. Only by setting the threshold $\rho < 1$, the search space becomes finite.

One precomputational step which all algorithms have in common is the computation of the resource graph $\hat{G}$ and its edges which constitute cost-optimal paths between resource locations in the underlying road network graph. The pseudocode for this operation is given in Algorithm 2. The set of edges $\hat{E}$ is realized as an adjacency matrix A of dimension $N + 1 \times N$, where $|\mathcal{X}(q)| = N$ is the number of suitable resources of the respective query setting. For notational reasons, we denote the query node q by X^0. The entries $a_{ij}, 0 \leq i \leq N, 1 \leq j \leq N$ of A are defined as:

$$a_{ij} = \begin{cases} c(p(X^i, X^j)) & \nexists X^k \in p(X^i, X^j), i, j, k \text{ pairw. inequal} \\ \infty & else \end{cases}$$

where $p(X^i, X^j)$ denotes the cost-optimal path from X^i to X^j within the underlying road network graph. A holds the cost of all cost-optimal paths, both pairwise between resources as well as from q to any resource, if no intermediate resources are located along this path. A, however, does not hold any information about paths from any resource to q, because the query node is not a resource and thus does not yield any gain toward the query goal. In case cost does not refer to travel time, we also compute the travel times of the cost-optimal paths. Recall the example of a query setting and its resource graph, as illustrated in Figure 6.1.

The according adjacency matrix of this scenario is given by:

	A	B	C
q	∞	$t(p(q, B))$	$t(p(q, C))$
A	∞	∞	$t(p(A, C))$
B	∞	∞	$t(p(B, C))$
C	$t(p(C, A))$	$t(p(C, B))$	∞

As mentioned before, depending on the application, the subset of suitable resource locations may be query-dependent. However, as the total set of resource locations is known prior to the query, it is possible to precompute the above adjacency matrix. If at query time a selection of suitable resources is required, the non-relevant rows and columns may simply be ignored. In the following, we consider an appropriate adjacency matrix as given. This assumption is not to the disadvantage of any of the proposed algorithms, as they all rely on the resource graph $\hat{G}$ and its edges modeled by the adjancency matrix.

6.5.1 Heuristic Solutions

In order to cope with the complexity of the PRRQR, we propose two search heuristics employed in greedy algorithms. Both heuristics aim at exceeding the given probability threshold by extending a (partial) resource route r by the "best" next resource location. The first heuristic greedily chooses the resource location which yields the best success

probability upon arrival, while the second heuristic greedily chooses the resource location which yields the best tradeoff between success probability upon arrival and cost to reach the location from the present one. Formally, we propose to evaluate a possible extension of resource route r along resources $(X^{r_1}, \ldots, X^{r_{i-1}})$ by one of the resource locations $\{X^1, \ldots, X^N\}$ according to the following heuristics:

(1) Extend r by X^{r_i} such that

$$X^{r_i} = \arg\max\{\mathbb{P}(X_{t_j}^{r_j} = 0) \mid 1 \leq j \leq N\}$$

(2) Extend r by X^{r_i} such that

$$X^{r_i} = \arg\max\{\mathbb{P}(X_{t_j}^{r_j} = 0)/\hat{c}(e_{r_j}) \mid 1 \leq j \leq N\}$$

where $\hat{c}(e_{r_j})$ denotes the cost for traveling from resource location $X^{r_{j-1}}$ to X^{r_j} along the an cost-optimal path.

In conclusion, our greedy approaches **G1** and **G2** proceed as follows: For a given query setting $q, \mathcal{X}(q)$ and a probability threshold $0 < \rho < 1$ all cost-optimal paths from q to all resource locations adjacent to q w.r.t. the adjacency matrix are computed. Then, **G1** and **G2** choose the most promising extension according to the extension strategies (1) and (2), respectively. If the success probability of the obtained resource route does not exceed the probability threshold ρ, the procedure is repeated for all resource locations adjacent to the current one w.r.t. the adjacency matrix. As soon as the success probability exceeds ρ, we have found a viable solution.

6.5.2 Optimal Results

The greedy approach described above aims at the computation of reasonable resource routes in efficient time. However, in some applications, quality is more important than efficiency. For these cases, we propose two different approaches which guarantee optimal results. We present a backtracking algorithm and a further accelerated branch and bound approach.

Optimal Results through Backtracking

The backtracking approach, denoted by **BT**, starts at query node q and gradually expands resource routes as long as they qualify as result candidates. A resource route disqualifies as a result candidate if it exceeds the expected cost of the currently best resource route. During the expansion, **BT** explores the search tree (rooted at q) in depth-first order. Note that this search tree is generally infinite. Consequently, it is of even greater importance to exclude resource routes from expansion early on in the algorithm. Therefore, we conduct a prior initialization step equal to an execution of the greedy **G2** algorithm. This generates a valid resource route in efficient time, its expected cost may be used as a first bound. We

Algorithm 3: Expansion step of **BT**

```
1  expandRecursive(Resource route r,
2  resource X)
   Data: Upper bound for E_c, M_c
   Output: Optimal resource route r̂
3  begin
4      if E_c(r) > M_c then
5          return ∅
6      end
7      if P_S(r) > ρ then
8          return r
9      end
10     initialize variable holding current best route r̂ = ∅
11     foreach resource X adjacent to last resource of r do
12         r' ← expand(r, X)
13         if r̂ = ∅ ∨ E_c(r') < M_c then
14             r̂ = r'
15             M_c ← E_c(r̂)
16         end
17     end
18     return r̂
19 end
```

omit the initialization step here (since it is equal to the description of **G2** above). Instead, we only give the recursive procedure as presented in Algorithm 3.

The procedure `expandRecursive` is initially called with a trivial resource route only consisting of query node q and an unspecified resource (which is reset to an adjacent resource during the first run). The expected cost of the result generated by **G2** is held in a global variable M_c as an initial upper bound for the expected cost. While traversing the search tree M_c will be tightened by finding better solutions. In line 3, candidates which do not qualify as results are excluded, while in line 6 possible result are generated (i.e., resource routes exceeding the probability threshold). The actual search tree traversal is realized recursively in lines 10, 11. If an expansion r' of a resource route r is better than the current best route along this subtree $\hat{r}$ (w.r.t. the expected cost), then $\hat{r}$ is updated to r' and M_c is updated to $\mathbb{E}_c(r')$ (lines 13,14). Thus, by sequential traversal of the search tree, **BT** returns the optimal result upon termination. However, due to the exponential number of branches and the technically infinite length of the branches runtime is prone to degenerate. Therefore, we propose another algorithm which computes optimal results in significantly less time.

Optimal Results through Branch and Bound

Like the backtracking algorithm **BT**, this branch and bound approach, denoted by **BB**, relies on an upper bound for the expected cost (M_c) which is tightened as the algorithm progresses. Additionally, **BB** incorporates a forward estimation for the expected cost a route minimally needs to exceed the probability threshold ρ. The forward estimation is a lower bound for the expected cost w.r.t. a resource route and ρ. Consequently, if this lower bound exceeds the upper bound for the expected cost M_c, r can be excluded from further expansion, i.e. the respective subtree can be pruned.

Algorithmically, **BB** is similar to **BT**, except for the mentioned forward estimation. This forward estimation is incorporated into Algorithm 3 as an if($m_c < M_c$))-statement spanning from line 11 through line 17, where m_c is the output of procedure `forwardEstimation`, as presented in Algorithm 4. Before each possible expansion of a resource route r, `forwardEstimation` is called with r and the probability threshold ρ. In lines 3-7 the parameters are set which are subsequently used to compute the lower bound for the expected travel time. t_{now} is the absolute travel time of input resource route r. t_{min} is the fastest travel time between any two resources. Thus, t_{opt} is the minimal possible arrival time at the next resource w.r.t. the absolute travel time of r. p_{opt} and m_c are initialized with $\mathbb{P}_S(r)$ and $\mathbb{E}_c(r)$, respectively. Both values are updated in the while loop (lines 8-13) until $p_{\text{opt}} \geq \rho$, i.e. until the optimal success probability exceeds the threshold. Now, let us investigate the operations in the while loop. First, p_{max} is defined as the maximal probability among all resources at the minimal possible arrival time t_{opt}. Note that we only allow observations to be incorporated into the model until query time. Therefore, p_{max} is monotonically decreasing in t_{opt}, and it converges against the minimal value of all stationary distributions in state `consumed`. m_c is extended by a new summand reflecting a hypothetical and optimal journey to the resource with maximal probability and minimal cost. Consequently, the success probability bound is updated to the probability of the complementary event of not finding any available resource along this optimal journey. By this strategy, in every iteration of the while loop, a journey to an optimal next resource causing minimal cost is simulated. Thus, the maximal success probability is aggregated while assuming minimal cost. We prove this in the following lemmas. We introduce the following terminology: For a given resource route r we refer to any iteration of the while loop of Algorithm 4 as an *optimal extension*. This coincides with the above described intuition.

Lemma 4 *In any optimal extension the gain of the updated values $m'_c \leftarrow m_c$ and $p'_{opt} \leftarrow p_{opt}$ yield the best possible trade-off between expected cost and success probability.*

More specifically, let r be a resource route with $\mathbb{E}_c(r) = m_c$, $\mathbb{P}_S(r) = p_{opt}$ and travel time t_{opt}. Then the following statement holds: For any possible extension r' of r to another resource:

$$\frac{m'_c - m_c}{p'_{opt} - p_{opt}} < \frac{\mathbb{E}_c(r') - \mathbb{E}_c(r)}{\mathbb{P}_S(r') - \mathbb{P}_S(r)}$$

Algorithm 4: Forward Estimation of **BB**

```
1  forwardEstimation(Resource route r, current E_c bound M_c)
   Output: Upper bound for the success probability of any extension of
           r until it exceeds M_c
2  begin
3      t_now ← arrival time at last resource of r
4      t_min ← min_{e∈Ê} t̂(e)
5      t_opt ← t_now + t_min
6      p_opt ← P_S(r)
7      m_c ← E_c(r)
8      while m_c < M_c do
9          p_max ← max_{X∈X(q)} P(X_{t_opt} = 0)
10         m_c ← m_c + (t_opt · p_max(1 − p_opt))
11         p_opt ← 1 − ((1 − p_opt)(1 − p_max))
12         t_opt ← t_opt + t_min
13     end
14     return p_opt
15 end
```

Proof 4 *In order to prove this lemma, we need to formulate the success probability of a resource route r differently:*

$$\begin{aligned}\mathbb{P}_S(r) &= 1 - \prod_{i=1}^{n} \mathbb{P}(X_{t_i}^{r_i} = 1) \\ &= \sum_{i=1}^{n} \left(\mathbb{P}(X_{t_i}^{r_i} = 0) \cdot \prod_{j=1}^{i-1} \mathbb{P}(X_{t_j}^{r_j} = 1) \right)\end{aligned}$$

Note that the equality indeed holds. This is because the event of finding at least one available resource can be described by the complementary event of not finding any resource in state available. Equally, it can be described as the union of events that resource X^{r_i} is available but all other resources thus far were consumed.
Now, for a given resource route r, let r' denote an arbitrary extension of r by another resource X with respective arrival time t. By the alternative definition of $\mathbb{P}_S$, we have $\mathbb{P}_S(r') = \mathbb{P}_S(r) + t\mathbb{P}(X_t = 0) \cdot (1 - \mathbb{P}_S(r))$. Recall that $\mathbb{E}_c(r) = m_c$ and $\mathbb{P}_S(r) = p_{opt}$.

Now, we show our claim:

$$\frac{m'_c - m_c}{\mathbb{E}_c(r') - m_c} < \frac{p'_{opt} - p_{opt}}{\mathbb{P}_S(r') - p_{opt}}$$

$$\frac{c'_{opt} p'_{opt}(1 - \mathbb{P}_S(r))}{c(r)\mathbb{P}(X_t = 0)(1 - \mathbb{P}_S(r))} < \frac{p'_{opt}(1 - \mathbb{P}_S(r)}{\mathbb{P}(X_t = 0)(1 - \mathbb{P}_S(r)}$$

By definition, $t'_{opt} \leftarrow t_{opt} + t_{min}$, where t_{min} denotes the minimal travel time in $\hat{E}$. Consequently, $t'_{opt} < t$, therefore, the inequality holds which proves the claim.

Lemma 5 *Let r be a resource route. The forward estimation of the expected cost as computed by Algorithm 4 is indeed a lower bound.*

Proof 5 *This follows from the following properties:*

(i) The number of optimal extensions needed until r exceeds the probability threshold is at most the number of actual extensions needed.

(ii) Every optimal extension of r yields a better trade-off than an actual extension.

(iii) No sequence of actual extensions of r can exceed the probability threshold while yielding a lower expected cost than the sequence of optimal extensions chosen by Algorithm 4.

(i) follows directly from the definition of $p_{opt} \leftarrow 1 - \big((1-p_{opt})(1-p_{max})\big)$. In every optimal extension, p_{opt} is increased by the maximally possible value. Therefore, no other sequence of extensions can yield a faster increase. (ii) is the statement of Lemma 4. Finally, (iii) follows from both, (i) and (ii).

Note that all of the above is easily applied to the case where instead of minimizing the expected cost w.r.t. a probability threshold we maximize the probability w.r.t. an absolute cost bound (as introduced at the end of Section 6.4.2). For example, consider the backtracking expansion Algorithm 3. Instead of dismissing (storing) a route r if $\mathbb{E}_c(r) > M_c$ ("$<$" holds), in the complementary scenario, a route r is dismissed (stored), if $c(r) > M_c$ ("$<$" holds). Similarly for the forward estimation presented in Algorithm 4. Again, the expected cost $\mathbb{E}_c(r)$ is to be replaced with the absolute cost $c(r)$. While the absolute cost bound is not exceeded, optimal path extensions are simulated, adding maximal success probability to the path. When the cost bound is exceeded and the maximal current best success probability is not surpassed, the search tree can be pruned. If, on the other hand, the success probability is surpassed by the optimal path extension, the path (and its subtree in the search tree) qualifies as a candidate. As for the theoretic arguments, they apply analoguously, therefore we omit an adapted version due to space limitations.

Concludingly, we have presented four algorithms for solving the proposed PRRQR in this section. **G1** and **G2** follow a greedy heuristic to produce approximate results, while **BT** and **BB** produce exact results. **BB** is an extension of **BT** which makes use of a lower bound forward estimation of the expected cost. In the above lemmas, we have shown correctness of the proposed bound.

6.6 Experimental Evaluation

We evaluate our model and our algorithms on settings in real world road networks extracted from OpenStreetMap[1] (OSM) using the MARiO framework [68]. All experiments were conducted on a desktop computer equipped with an Intel Core i7-3770 CPU and 32 GB RAM, running Java 1.64 (64-Bit) on Linux 3.13 x86_64. Different algorithms are always tested on the same randomly generated scenario before comparing results. Runtime evaluations are based on Java's nanotime clock and performed for each algorithm individually excluding preliminary steps like graph population and building of the adjacency matrix. Computation of the latter takes around 250 milliseconds, for standard settings in the *Parking* and *Charging* scenarios, respectively. Note that all cost-optimal paths were computed using Dijkstra's algorithm. Choosing a different routing algorithm or employing a speed-up technique would yield the same benefit for all compared approaches. Modifying the path computation algorithm is an easy task, however, on a city scale (which the applications require) this would hardly yield any computational benefit. We present experiments for two realistic applications:

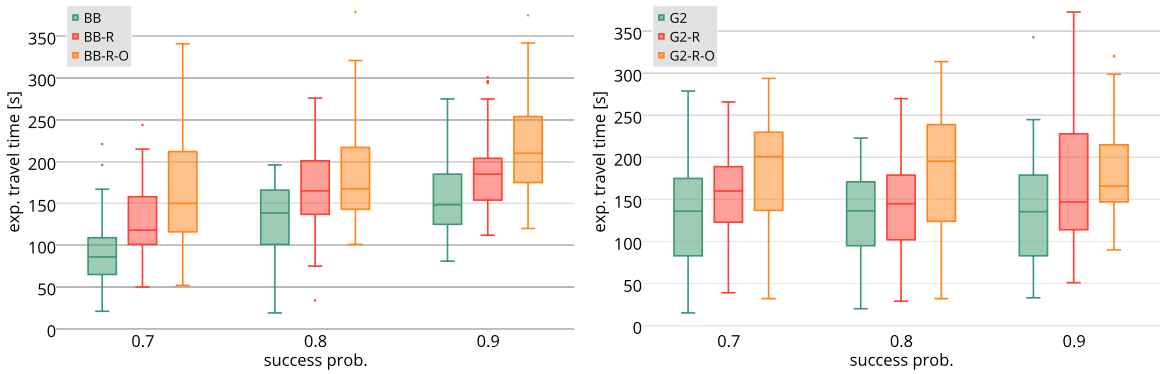
- *Parking* scenario (located in Bamberg, Germany): Given a probability threshold, we provide a route along parking spots which surpasses the threshold and minimizes the expected travel time. This scenario is based on ground truth extracted from OSM metadata.
- *Charging* scenario (located in Brussels, Belgium): Given a query position and a range limit (as used by electric vehicles), we provide a route along charging stations not exceeding the range limit and maximizing the success probability.

Note that these scenarios are complementary w.r.t. the criterion which is bounded and the criterion which is to be optimized, as explained in Section 6.4.1. Besides, while *Charging* relies on an hard numeric bound (distance), *Parking* relies on the more sophisticated expected value bound. Therefore we choose *Parking* as our main scenario. We will not present all charts for both scenarios, however noting that corresponding charts show the same behavior.

6.6.1 Parking Scenario

We generated the following test cases on the road network of the city of Bamberg, Germany, containing approximately 10.000 nodes and 20.000 edges as well as nearly exhaustive metadata regarding parking spots. For every test case, a target node is randomly drawn from all road network nodes of degree ≥ 1 within a three kilometer radius from the city center. Then, an isochrone of 800 meters walking distance is computed around the target. Let N be the number of resources (according to the ground truth) within the isochrone. In our experiments, resources are rather dense, i.e., $25 \leq N \leq 100$. Subsequently, the query

[1] http://www.openstreetmap.org/

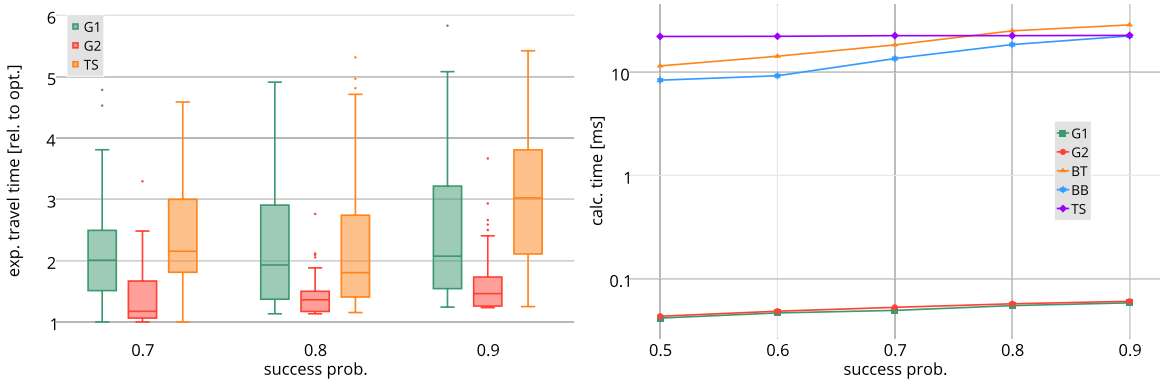


(a) **BB**-related algorithms (b) **G2**-related algorithms

Figure 6.2: Illustration of the influence of model complexity on the quality of results (*Parking* scenario.

node q is randomly drawn from all nodes within the isochrone. This corresponds to the use case where we expect the user to trigger the query when they are in the vicinity of their target. The average and maximal distance from q to a resource are by construction 800 and 1600 meters, respectively. Finally, $M \leq N$ observations of resource availability are randomly distributed among the resource locations, and the respective sojourn times in the states `available` and `consumed` are set. For reasons of clarity, in our experimental settings the sojourn times are set to the same configurations for all resources. We assume the expected time a spot stays vacant (`available`) to be 3 minutes and the expected time a spot stays occupied (`consumed`) to be 90 minutes. Note that the resources could easily be parametrized separately to model differently volatile resources. In this scenario, a probability threshold is given, and as a cost function we use travel time as formalized in Definition 4. The optimal resource route is the one with the least expected travel time among all resource routes with a success probability exceeding the threshold.

First, we want to evaluate how much the additional information held by our probabilistic model improves result quality. Recall, that our model supports reappearance and incorporates short term observations, two properties that distinguish this work from others. In order to prove that the gain in result quality outweighs the gain in model complexity, we trim our algorithms **BB** (branch and bound) and **G2** (greedy approach with probability per cost heuristic) to partially ignore the information provided by the underlying model. In a first step, we disable the possibility of resource reappearance, we denote these approaches by **BB-R** and **G2-R**, respectively. This means, **BB-R** and **G2-R** proceed like their respective counterparts but do not revisit resources which have previously been observed as `consumed`. This corresponds to a simpler probabilistic model without the feature of resource reappearance. In a second step, we additionally disable short term observations. We denote these approaches by **BB-R-O** and **G2-R-O**. As before, they proceed like **BB-R** and **G2-R**, respectively, but additionally ignore any short term observations.



(a) Expected travel time relative to optimal solution

(b) Calculation time

Figure 6.3: Illustration of quality as well as efficiency of all algorithms in the *Parking* scenario.

Hence, the variations emulate an even simpler model which only allows static uncertainty, as used in [51], for example.

The results for the **BB**-related and the **G2**-related algorithms are shown in Figures 6.2(a) and 6.2(b), respectively. Both figures depict the same settings. It is obvious that requiring a greater probability threshold results in resource routes with longer expected travel time. Therefore, the overall increase in expected travel time is consequential. Both figures clearly show that the algorithms which rely on greater information, i.e., use a more complex model, yield better results. Figure 6.2(a) visualizes the results of the branch and bound approaches which are optimal w.r.t. to the information available. As claimed, **BB** on average outperforms **BB-R**, its counterpart which does not allow reappearance by at least 20 percent. **BB-R**, in turn, outperforms its counterpart which does not incorporate short term observations, **BB-R-O**. This supports the previously made claim that resource reappearance and short term observations do indeed improve the quality of results. From Figure 6.2(b) we observe, that simpler algorithms also benefit from the additional information contained in the model. Comparing the two figures, **BB**-algorithms of course yield better results than **G2**-algorithms and do so with significantly less variance than the greedy approaches. This is because the heuristics rely on chance in the form of beneficial problem settings in order to generate near-optimal results.

Next, let us investigate the performance of the algorithms presented. As mentioned before, there exists no work which is fully comparable. However, as the PRRQR is related to the TSP and clustering is commonly used to approximate the TSP (as in [47]), we use this concept to implement an approximative comparison partner denoted by **TS**. It is important to mention that **TS** does not support resource reappearance, because otherwise the heuristic would not visit sufficiently many distinct resources to achieve a comparable success probability. Before we present the results, let us explain how **TS** proceeds. In a

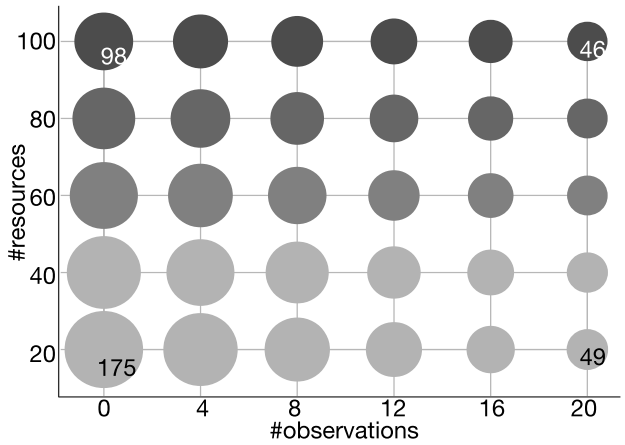
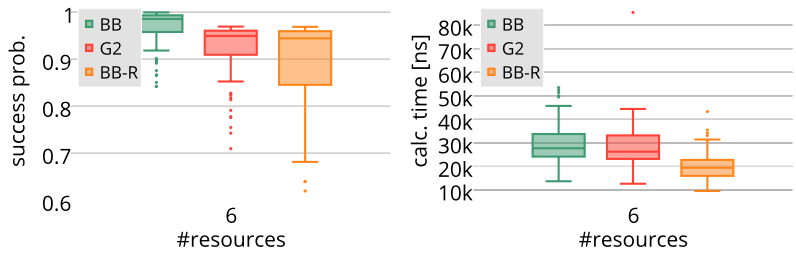


Figure 6.4: Influence of the number of resources and the number of observations on the expected travel time, i.e., result quality (for a probability threshold of 0.7)

first step, **TS** conducts a k-medoid clustering on the set of all resources, where experimentally $k = 6$ has proven adequate. Subsequently, a TSP on the cluster medoids (starting at the query node) is solved. Then follows the actual resource route computation. It starts at the query node and computes the cost-optimal path to the first medoid. In the respective cluster, a greedy depth-first search (starting at the medoid) is conducted, returning an approximation of the cluster-internal cost-optimal path. From the last resource of the cluster we compute the cost-optimal path to the next medoid. This procedure is continued until the resource route exceeds the given probability threshold. **TS** serves as an algorithmic competitor based on a simpler probabilistic model but with a solid heuristic that has proven efficient when solving TSP-related problems. Note that the cost-optimal paths between all resources are precomputed in order to make the comparison to our algorithms – which use the precomputed adjacency matrix – fair.

We compare **TS** to all algorithms introduced in Section 6.5, i.e., the two greedy approaches **G1** and **G2** as well as the exact solutions **BT** and **BB**. Figure 6.3(a) shows the quality of the results produced by the approximative algorithms, i.e., **G1**, **G2**, and **TS**. Their respective expected travel times are given relative to the optimal results. The higher the probability threshold, i.e., the more complex the task, the greater the discrepancy between optimal results and approximation. Although **G2** relies on the rather simple probability-to-cost ratio heuristic, it significantly outperforms its comparison partners. While **G2** yields near-optimal results in the easier settings, the optimal solutions in the most elaborate scenario (probability threshold 0.9) undercut its expected travel times on average by about 30 percent. This gain in quality, however, comes at the price of calculation time, as depicted in Figure 6.3(b). This illustration shows the averaged runtimes of all algorithms when increasing the required probability threshold. The greedy approaches generate results in almost interactive time, while **BB**, **BT**, and **TS** are around two to three orders of magnitude slower. However, it is important to note that two orders of magnitude



(a) Success probability (b) Calculation time

Figure 6.5: Illustration of quality as well as efficiency of selected algorithms in the *Charging* scenario.

only correspond to around 100 ms of calculation time. Comparing the exact algorithms, we observe that **BB** outperforms **BT** which can be attributed to the forward estimation. The competitive approach **TS** performs in constant time of about 150 milliseconds (for the same number of resources), however generating the worst results.

Finally, we want to explore how volatile the results are w.r.t. the model parameters. We restrict ourselves to the optimal solution provided by **BB**, seeing as the quality ratio of optimal to approximative solutions has been explored above. Figure 6.4 depicts the influence of the number of parking spots relative to the number of short term observations of vacant parking spots. Thus, each circle in the plot corresponds to a pair of parameter values, and the diameter of each circle represents the average expected travel time of this scenario in seconds, as do the numbers in the corner circles. The result shows the expected behavior that with an increasing number of parking spots, expected travel time decreases. Furthermore, for any given scenario, it can be seen that the increased amount of short term observations also reduces the expected time until a vacant spot is found. Similarly expectable behavior is observed when varying the sojourn time parameters $1/\lambda$ and $1/\mu$, therefore further charts are omitted.

6.6.2 Charging Scenario

For *Charging* we generated test cases on the road network of the city of Brussels, Belgium, containing approximately 30.000 nodes and 67.000 edges. For every test case a query node is randomly drawn from all road network nodes of degree ≥ 1 within a 6 kilometer radius of the city center. Then, an isochrone of 6 kilometers is computed around the query node, wherein 6 resource locations are randomly drawn. We have evaluated other numbers of resources but the results do not reveal additional information and are therefore omitted here. Compared to *Parking*, where nearly every street holds at least one resource, this scenario models resource scarcity. Again, if N denotes the number of resources (6 in our experiments), then $M \leq N$ observations of resource availability are randomly distributed among these resource locations. The expected time a charging station remains vacant

(`available`) is set to 30 min, and the expected time it remains occupied (`consumed`) is set to 50 minutes. As before, every charging station may be parametrized individually, however we pass on it for reasons of clarity and lack of ground truth. In this scenario an absolute distance bound of 6 kilometers is giving, emulating the remaining range of an electric vehicle with low battery. Note that in contrast to *Parking*, this bound is strict and cannot be exceeded. Every algorithm computes a route with an absolute distance of 6 kilometers, the optimal resource route is the one with maximal success probability.

In a first setting, we compare the result quality of our exact algorithm **BB**, our greedy solution **G2** and **BB-R**, the branch and bound variation which does not incorporate resource reappearance. Additionally to the scarcity of resources, the remaining range (6 kilometers) is only double the average distance from query node to the next resource (3 kilometers, as resources are distributed uniformly within the isochrone). Due to these tightened constraints, superiority of the optimal results generated by **BB** becomes more apparent. In almost three out of four runs, **BB** yields a success probability of over 95 percent, outperforming **G2** significantly. While the greedy heuristic worked well before, it is now easily lead down a considerably less beneficial branch of the search tree. Nonetheless, **G2** still produces slightly better results than **BB-R**. Again, this advocates our model which supports resource reappearance. Even an approximative approach on our model yields better results than an exact algorithms on a less sophisticated model due to lack of information. Of course, a simpler model needs less intricate function evaluations. In our case, however, the difference is merely a matter of microseconds, as depicted in Figure 6.5(b).

Concludingly, we have empiricially proven the benefit of our probabilistic model. It improves the quality of results by incorporating richer information, especially for complex but also for simpler tasks while not causing any significant computational overhead. On the contrary, our greedy approaches deliver competitive results in near-interactive time while our branch and bound approach yields optimal solutions in efficient time.

6.7 Conclusions

In this chapter, we investigate probabilistic route queries in road networks where the user is guided along a set of resources in order to maximize the probability of encountering an available resource. We aim to find a route with minimal expected cost among all routes exceeding a given probability threshold. We propose a novel framework in which resources are modeled as continuous-time Markov chains with two states, `available` and `consumed`. In contrast to similar problems, our framework allows for consumed resources to reappear and take short term as well as long term observations into account. The introduced query, referred to as PRRQR, is theoretically NP-complete and has an unlimited search space.

To solve this problem, we propose approximative as well as optimal solutions. We employ two different search heuristics in a greedy algorithm to achieve a trade-off between accuracy and calculation time. Furthermore, solutions using backtracking and a branch and bound approach provide optimal solutions in competitive time. We demonstrate the

superiority of our model as well as the efficiency and effectiveness of our algorithms on two realistic applications. The first is the search of a vacant parking spot, and the second is the search for a vacant charging station for electric vehicles.

For future work, we want to turn to settings considering other types of observations like competing drivers looking for the same type of resource. Furthermore, we want to investigate the influence of edge costs which might change during the search.

Chapter 7

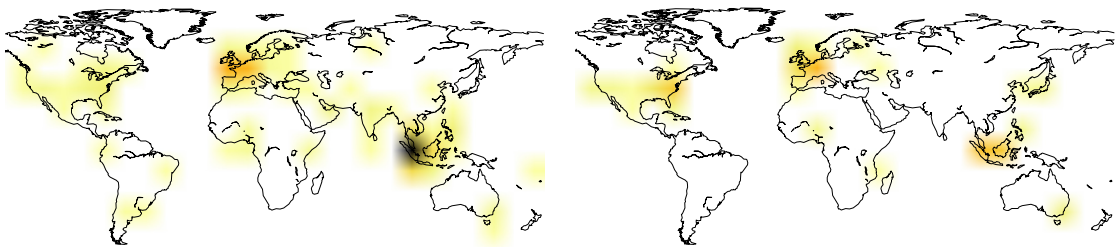
Trend Dissemination Mining on Social Media Data

7.1 Introduction

Social media such as Twitter or other microblogging platforms are a popular source for live textual data, often associated with geographic information. Such data may describe an event, an experience or a point of interest that is relevant to a user. More generally speaking, such microblogs describe events, objects and persons that are on the mind of a user. The prediction of trends has a plethora of economic applications in targeted marketing and investment banking, by knowing what people will have on their mind tomorrow. In this work, we do not predict new trends. However, we predict the flow of existing trends over the globe. For instance, trends related to *fashion* might often arise in France, then move over to the rest of Europe within a few days, then start to affect North America within weeks, and then flow to Australia within weeks and months. In contrast, technological trends might often be initiated in Japan and South Korea, then flow to North America, and only then flow to Europe.

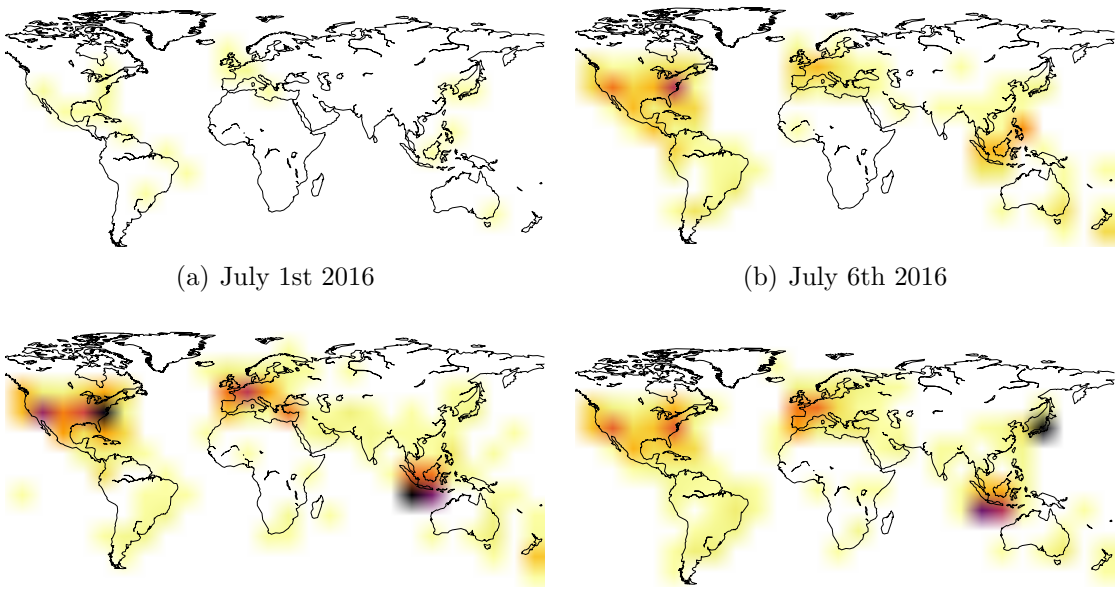
As an example of such a trend, Figure 7.1 shows the location of tweets issued in July of 2014 corresponding to the lost Malaysian Airlines flight "MH17". The trend shows initial strong bursts in Malaysia as well as in the Netherlands, from where the missing flight originated, as seen in Figure 7.1(a). From there, the trend quickly spread all across the world – two days later, the rest of Europe as well as North America are just as involved in the trend. This can be seen in Figure 7.1(b).

A more recent trend development can be seen in Figure 7.2, where the location of tweets containing the string "Pokémon" is shown for several days. Beginning with the first of July, 2016, Figure 7.2(a) exhibits a globally low interest in this topic, indicating no trend at that time. As the free-to-play game "PokémonGo!" was released for cell phones in the United States, Figure 7.2(b) shows a highly significant burst of tweets on this topic on July the 6th, originating in the US alone. One week later, on July 13th, the trend has moved to Europe as the game was released in several countries there. This can be seen



(a) July 17th 2014 (b) July 19th 2014

Figure 7.1: Distribution of trend “MH17”



(a) July 1st 2016 (b) July 6th 2016

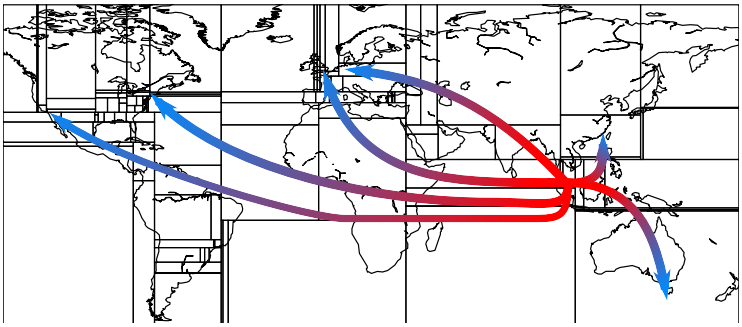
(c) July 13th 2016 (d) July 22nd 2016

Figure 7.2: Distribution of trend “PokémonGo!”

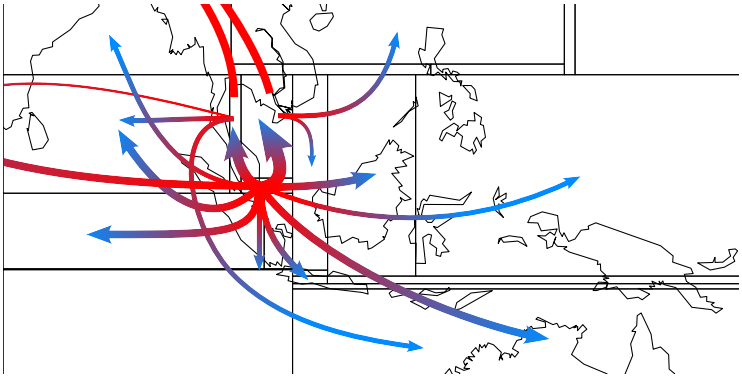
in Figure 7.2(c). Asia follows, mainly with the Japan release on July 22nd, with a high activity regarding the topic as shown in Figure 7.2(d).

Intuitively, different types of trends are expected to show different distributions. While a few trends spread to a global scale within hours due to dissemination through news networks, other trends may be more local, spread slower, might be originating from specific regions, or might disseminate to specific regions only.

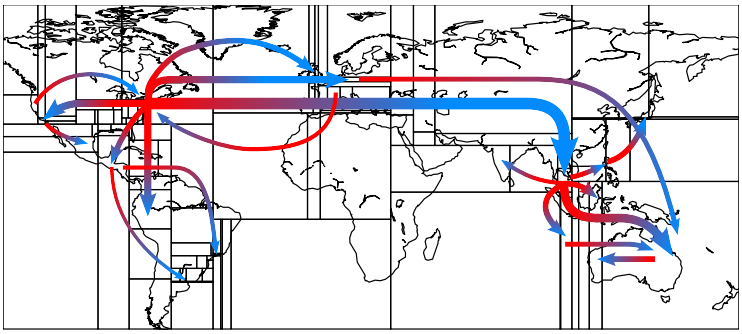
In this work we model and mine such dissemination of trends over space and time. That is, we observe the flow of trends, specified by source and target regions, over time. Figure 7.3 exemplifies such flows for the two examples given before, namely “MH17” and “PokémonGo!”. The arrows on the map indicate a flow in activity from source (red) to target (blue). For the sake of readability, the representation has been kept coarse and omits certain regional interdependencies. Geographical regions are referenced by their position



(a) “MH17” - world map



(b) “MH17” - detail map - south-east Asian



(c) “PokémonGo!” - world map

Figure 7.3: Spatio-Temporal Trend Dissemination

in our index (drawn in black outlines), and thickness of arrows indicates strength of the dependence. Figures 7.3(a) and 7.3(b) exhibit trend dissemination of the trend “MH17” in a full world view and one of the south-east Asian region alone, respectively. As can be seen very clearly, the trend originates from Malaysia and spreads over the world from there, partially using other regions as intermediate hops. In contrast, Figure 7.3(c) uses the same representation for the trend “PokémonGo!” on a world-wide scale and while there is a general main direction from the US east coast, several rules in the opposite direction indicate a more diverse dissemination pattern. Curiously, once again, south-east Asia is a strong hub for this trend, resulting from a local burst on this topic from Indonesia.

But rather than looking at a few, hand-selected, trends as shown in these figures, we use existing trend mining solutions to automatically extract the disseminations of a large number of past trends. Each trend yields a spatio-temporal trend-tensor, containing for each discrete time interval, and each spatial region the number of corresponding tweets. As our first contribution, we postulate and verify the hypothesis that trends follow different archetypes, which differ strongly in terms of their dissemination patterns. Using a clustering approach, we identify these archetypes trends. For new trends, this result can be used to quickly classify a new trend as an archetype trend, to more effectively predict its future dissemination, allowing to predict where a trend will move to in the near future.

To model the dissemination of trends in space and time, this chapter is organized as follows. The next section, Section 7.4 gives an overview over the state-of-the-art of modelling trends in space and time. Section 7.2, formally defines a trend, and introduces our notion and data structures to define the spatio-temporal motion of a trend. Section 7.3.4 presents our technical concept for modeling the dissemination of a trend. This concept is experimentally evaluated in Section 7.5, and the chapter is concluded in Section 7.6.

7.2 Preliminaries

This section will define terms and notations used throughout this work, and formally defines the problems tackled in the following. In this work we consider spatio-temporal text data, that is text data annotated with a geo-location and a timestamp, such as obtained from Twitter.

Definition 6 (Spatio-Temporal Text Database) *A* spatio-temporal text database $\mathcal{DB}$ *is a collection of triples* (s, t, c)*, where* s *is a point in space,* t *is a point in time, and* c *is a textual content.*

A concept that we adopt from the literature is the concept of a *trend* as introduced in [159].

Definition 7 (Trend) *A trend* $\tau_{K,t}$ *is a set of keywords* K *that appear significantly more often starting at a time* t*.*

A more formal definition, which introduces the requirements of a set of terms to be considered as significant, will be given in Section 7.3.1. The set of spatio-temporal text objects which support trend $\tau_{K,T}$, is denoted as

$$\mathcal{DB}_{\tau_{K,T}} = \{(s, t, c) \in \mathcal{DB} | c \in K \wedge t \in T\}.$$

Definition 8 (Spatio-Temporal Occurrence) *Let* $\tau_{K,T}$ *be a trend. Let* $\mathcal{S} = \{S_1, ..., S_{|S|}\}$ *be a partitioning of space into spatial regions, and let* $\mathcal{T}$ *be a partitioning of time into equi-sized time intervals denoted as epochs. Further, let* $T := t \cap \mathcal{T} = \{T_1, ..., T_{|T|}\}$ *be the set of epochs overlapping the trending time* T*. Then*

$$Occ_{\tau_{K,T},S} = |\{(s, t, c) \in \mathcal{DB} | s \in S \wedge t \in T \wedge c \in K\}|.$$

is the number of occurrences of trend $\tau_{K,T}$ *at region* S*.*

The aim of this work is to find the dissemination of trends, that is, pairs of spatial locations (S_1, S_2) such that any trend that appears in region S_1 is significantly more likely to appear in S_2 in the next epoch.

To describe the motion of a trend (K, t) in space and time, each trend is described by a time-space matrix, describing for each spatial region and each epoch $t \in T$ the number of tweets of the trend.

Definition 9 (Trend Count Matrix) *The trend count matrix $D(\tau_{K,T}) \subseteq R^{|S|} \times R^{|T|}$ contains all occurrences of trend $\tau_{K,T}$ over space and time, and is defined as follows:*

$$D(\tau_{K,T})_{i,j} = Occ(\tau_{K,T_i}, S_j)$$

In this work, the main task is to analyze and mine multiple trend count matrices as defined in Definition 9, in order to identify groups of similar trends, groups of similar spatial regions, and to find common spatio-temporal dissemination of trends. These problems are formally defined as follows.

Definition 10 (Trend Clusters) *Let $\mathcal{DB}$ be a spatio-temporal text database, let $\mathcal{DB}_\tau$ be a set of trends mined from $\mathcal{DB}$, and let $D(\tau \in \mathcal{DB}\tau)$ denote the trend count matrix of each trend. A trend cluster $C \subseteq \mathcal{DB}_\tau$ is a set of trends that exhibit mutually similar trend count matrices.*

Given a set of trends, the main challenge is to find association rules of the form "Any trend observed in region A today, is likely to appear in region B tomorrow". This kind of spatio-temporal trend dissemination is defined as follows.

Definition 11 (Spatio-Temporal Trend Dissemination Rule) *Let $\mathcal{DB}_\tau$ be a set of trends and their corresponding trend count matrices $D(\tau \in \mathcal{DB}\tau)$. For two spatial regions S_s and S_t, a spatio-temporal trend dissemination rule $S_s \rightarrow S_t$ implies that a large trend count at source region S_s at any time t indicates a large trend count at target region S_t at time $t + 1$, formally:*

$$(S_s \rightarrow S_t) \leftrightarrow \forall i, \forall \tau \in \mathcal{DB}_\tau : D(\tau)_{i,s} \rightarrow D(\tau)_{i+1,t},$$

where $D(\tau)_{i,s} \rightarrow D(\tau)_{i+1,t}$ denotes that a large value in $D(\tau)_{i,s}$ implies a large value in $D(\tau)_{i+1,t}$

Finally, Definition 11 allows us to define the problem of spatio-temporal trend dissemination rule mining.

Definition 12 *Let $\mathcal{DB}_\tau$ be a set of trends and their corresponding trend count matrices $D(\tau \in \mathcal{DB}\tau)$. The problem of spatio-temporal trend dissemination rule mining is to find all pairs of spatial regions (S_s, S_t) such that $(S_s \rightarrow S_t)$ holds.*

7.3 Spatio-Temporal Trend Dissemination Rule Mining

This section describes our approach at mining spatio-temporal trend dissemination rules. As a first step, we need to acquire past trends, to mine dissemination rules from. For this purpose, we apply existing textual trend mining solutions proposed in the recent past, which are briefly sketched in Section 7.3.1 for self-containment. Next, as a second step used for preprocessing, we employ a space composition scheme in Section 7.3.2 to ensure having a similar number of tweets in each spatial region using a k-d tree. As a third step, we model the flow of trends over space and time in Section 7.3.3. Therefore, we transform a *trend count matrix*, as defined in Definition 9, into a *trend flow tensor*, which describes the flow from any source region to any target region at any point in time for any trend. Consequently, constructing a *trend flow tensor* for each trend that we mined in the first step, yields a four-mode Space × Space × Time × Trends tensor, which will be fed to our fourth step, the mining step. In the mining step proposed in Section 7.3.4, we employ a tensor factorization approach to discover latent features of trends, latent features of trend-source-regions and latent features of trend-target-regions. These latent features allow us to cluster trends into sets of trends which disseminate similarly over space and time. Then, for each cluster of similar trends, we obtain trend flows from the *reconstructed trend flow tensor*.

7.3.1 Traditional Trend Mining

We use SigniTrend [159] to establish our trend baseline. SigniTrend uses Count-min data structures [42] for approximate counting and tracks the average and standard deviation of term and term pair frequencies. In order to estimate the average $EWMA$ and the variance $EWMVar$ for a frequency x on a data stream, they can rely on earlier work by Welford [191] and West [193] on incremental mean and variance. The update equations given by Finch [61] for the exponentially weighted variants allow these values to be efficiently maintained on a data stream:

$$\Delta \leftarrow x - EWMA$$
$$EWMA \leftarrow EWMA + \alpha \cdot \Delta \tag{7.1}$$
$$EWMVar \leftarrow (1 - \alpha) \cdot (EWMVar + \alpha \cdot \Delta^2) \tag{7.2}$$

The learning rate α can be set using the half-life time $t_{1/2}$; a parameter a domain expert will be able to choose easily based on his experience and needs:

$$\alpha_{\textit{half-life}} = 1 - \exp\left(\log\left(\tfrac{1}{2}\right) / t_{1/2}\right)$$

To capture interesting relationships among trends (such as `"Facebook"` bought `"WhatsApp"` or Edward `"Snowden"` traveled to `"Moscow"`) SigniTrend also tracks word pairs. A single term is thereby modeled as a co-occurrence with itself. Given a word pair (w, l) where w

and l are single word tokens, SigniTrend uses a classic model from statistics to measure the significance: Let $f_t(w,l)$ be the relative frequency of this pair of tokens within the documents $D_t = \{d_1, \ldots, d_n\}$ at time t, i.e.

$$f_t(w,l) := \frac{|\{w \in d \wedge l \in d \mid d \in D_t\}|}{|D_t|}$$

then they use the series of previous values $f_1, \ldots, f_{t-1}$ to compute an estimated value and a standard deviation. To facilitate aging of the data and to avoid having to store all previous values, they employ the exponentially weighted moving average ($EWMA[f(w,l)]$) and moving standard deviation ($EWMVar[f(w,l)]$). With these estimates, the z-score of the frequency is computed as follow:

$$z_t(w,l) := \frac{f_t(w,l) - \max\{EWMA[f(w,l)], \beta\}}{\sqrt{EWMVar[f(w,l)]} + \beta} \tag{7.3}$$

The term β is motivated by the assumption that there might have been $\beta \cdot |D|$ documents that contained the term, but which have not been observed due to incomplete data. With this Laplace-style smoothing we prevent instability for rare observations of pairs (w,l). For Twitter, the suggested value for this term is $\beta = 10/|D|$: intuitively we consider 10 occurrences to be a by chance observation. This also adjusts for the fact that we do not have access to the full Twitter data.

Terms and pairs with corresponding z-scores (see Equation (7.3)) larger than a given threshold τ are considered as trends. For our experiments we chose $\tau = 3$.

7.3.2 Space Decomposition Scheme

To fit a flow model between spatial regions, we need to minimize the bias that results from having a non-uniform distribution of tweets on earth. We remedy this problem by partitioning the geo-space in a way that minimizes the difference of tweets between spatial regions. For this purpose, we insert the geo-locations of all tweets in our database into a k-d tree, having a maximum node capacity of 1000. Thus, every leaf node of this k-d tree is guaranteed to have between 500 and 1000 two-dimensional points. Each of this leaf node is then used as a spatial region in the remainder of the work. The decomposition that we obtained this way is exemplarily depicted in Figure 7.4. Note that this tree is constructed upon a typical, yet static, set of tweets.

7.3.3 Trend Flow Modeling

In this section we describe our approach of obtaining a trend flow from raw trend s. Thus, for a given trend, we consider all N occurrences of this trend at some time t and all M occurrences at the next time $t+1$. All the regions having the trend at time t can be considered as sources of the trend, and all regions having the trend at time $t+1$ can

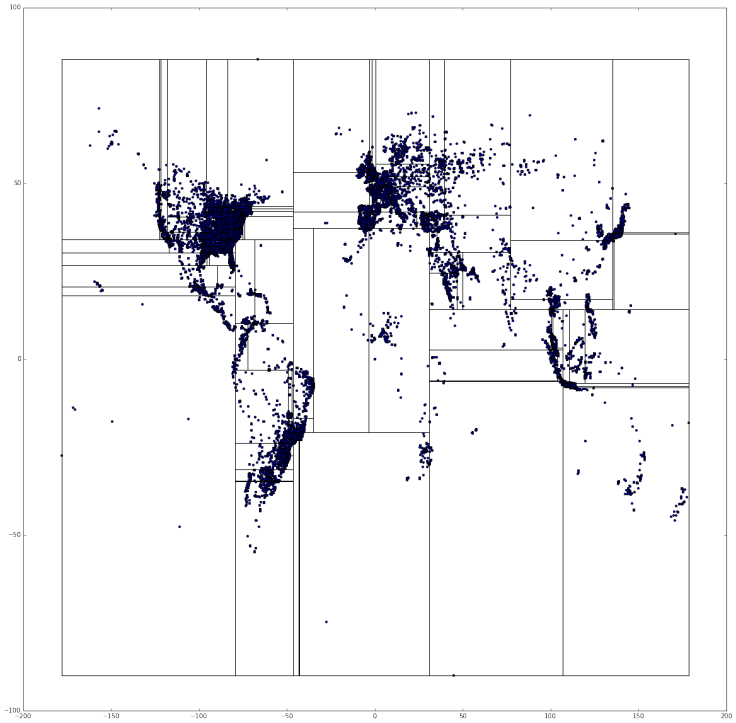


Figure 7.4: k-d tree based space decomposition

$T_i \rightarrow T_{i+1}$

$$\begin{pmatrix} 2 \\ 0 \\ 0 \\ 1 \end{pmatrix} \xrightarrow[S_s \rightarrow S_t]{flow} \begin{pmatrix} 3 \\ 1 \\ 0 \\ 5 \end{pmatrix} \xRightarrow[F(\tau_{K,T})]{tensor} \begin{pmatrix} \frac{6}{9} & \frac{2}{9} & 0 & \frac{10}{9} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{3}{9} & \frac{1}{9} & 0 & \frac{5}{9} \end{pmatrix}$$

$T_{i+1} \rightarrow T_{i+2}$

$$\begin{pmatrix} 3 \\ 1 \\ 0 \\ 5 \end{pmatrix} \xrightarrow[S_s \rightarrow S_t]{flow} \begin{pmatrix} 2 \\ 1 \\ 3 \\ 4 \end{pmatrix} \xRightarrow[F(\tau_{K,T})]{tensor} \begin{pmatrix} \frac{6}{10} & \frac{3}{10} & \frac{9}{10} & \frac{12}{10} \\ \frac{2}{10} & \frac{1}{10} & \frac{3}{10} & \frac{4}{10} \\ 0 & 0 & 0 & 0 \\ 1 & \frac{5}{10} & \frac{15}{10} & 2 \end{pmatrix}$$

Figure 7.5: Trend Flow Modelling

be considered as targets of the trend. Yet, we do not know any more specifically, which source region has affected which target region and to what degree, since we do not know through which channels and medias the trend was disseminated. Thus, due to lack of better knowledge, we assume that all sources affect all target uniformly. This flow model is formalized as follows

Definition 13 (Spatio-Temporal Trend Flow Model) *Let $\tau_{K,T}$ be a trend having a set of keywords K and having a time interval $T = \{T_1, ..., T_{|T|}\}$ which covers $|T|$ epochs. Let $\mathcal{S} = \{S_1, ..., S_{|\mathcal{S}|}\}$ be a space composition into $|\mathcal{S}|$ spatial regions. Furthermore, let $D(\tau_{K,T})_{i,j} = Occ(\tau_{K,T_i}, S_j)$ be the trend matrix of $\tau_{K,T}$. We define the* trend flow model *$F(\tau_{K,T})$ of trend $\tau_{K,T}$ as a $\mathcal{S} \times \mathcal{S} \times \{T_1, ..., T_{|T-1|}\}$ tensor, such that*

$$F(\tau_{K,T})_{i,j,k} = \frac{Occ(\tau_{K,T_k}, S_i) \cdot Occ(\tau_{K,T_{k+1}}, S_i)}{\sum_{S_n \in \mathcal{S}} Occ(\tau_{K,T_{k+1}}, S_n)}$$

Intuitively, an entry $F(\tau_{K,T})_{i,j,k}$ of tensor $F(\tau_{K,T})$ corresponds to the absolute flow of occurrences from region S_i to region S_j from time T_k to time T_{k+1}.

Example 1 *To illustrate the construction of tensor $F(\tau_{K,T})$, consider an example depicted in Figure 7.5. Here, the occurrences matrix of a tensor of a trend is shown for four spatial regions. At the first point of time t_i, the trend appears twice in the first region and once in the fourth region, yielding the vector $(2, 0, 0, 1)^T$. The second t_{i+1} and third point of time t_{i+2}, the distribution of occurrences is $(3, 1, 0, 5)^T$ and $(2, 1, 3, 4)^T$, respectively, yielding the trend matrix shown in Figure 7.5. Transitioning from the first epoch t_i to the second epoch t_{i+1}, the occurrences change from $(2, 0, 0, 1)^T$ to $(3, 1, 0, 5)^T$. The first spatial location S_1, having an initial value of two tweets, is thus a source of the trend. Since we cannot observe the latent means of dissemination of a trend (through the internet, via TV, radio, word-of-mouth, etc.), we estimate that region S_1 disseminates its trend to all other regions having this trend. Since a fraction $\frac{3}{9}$ of all tweets at time t_{i+1} are observed in region S_1, we estimate a trend-from of $\frac{2 \cdot 3}{9}$ from region S_1 to itself. In contrast, only one trending tweet is observed at location S_2 at time t_{i+1}, of which we contribute a flow of $\frac{2 \cdot 1}{9}$ to S_2. Similarly, a flow of $\frac{1 \cdot 5}{9}$ is contributed from S_4 to S_4.*

It is notable that each time-slice of tensor $F(\tau_{K,T})$ is a rank-1 matrix, as all lines are multiples of each other. This redundancy is desirable, as it evenly distributes the flow from all source regions to all target regions, and this redundancy will be removed in a later tensor factorization step. For each trend $\tau_{K,T}$ we obtain a three-mode tensor as described in Definition 13. Concatenating these tensors for each trend $\tau \in \mathcal{DB}_\tau$ yields a four-mode tensor $\mathcal{F}(\mathcal{DB})$ which is passed into the trend flow mining step described in the following.

7.3.4 Trend Flow Mining

We propose to decompose tensor $\mathcal{F}(\mathcal{DB}) \in \mathbb{R}^{I_1 \times ... \times I_N}$ using a CANDECOMP/PARAFAC tensor decomposition [27], [75] using k latent features, where k is a parameter of our algorithm. A CP factorization decomposes a tensor into a sum of component rank-one-tensors, i.e.

$$\mathcal{F}(\mathcal{DB}) \approx \sum_{r=1}^{k} u_r^1 \circ \cdots \circ u_r^N$$

where $u^n \in \mathbb{R}^{I_n}$ for $n = 1, \ldots, N$. Hence, as illustrated in Figure 7.6, this factorization decomposes our four-mode $\mathcal{S} \times \mathcal{S} \times T \times \mathcal{DB}_\tau$ tensor into four sets of vectors:

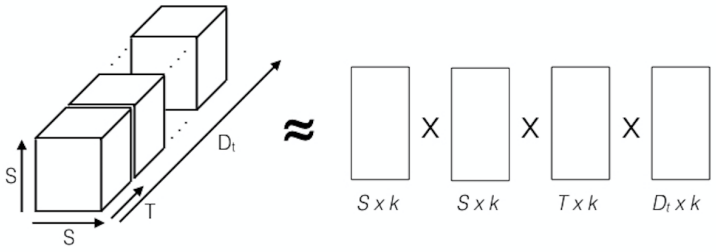


Figure 7.6: Trend Flow Modelling - Tensor Decomposition

- a set of k vectors of latent features of length $|\mathcal{S}|$ describing each source spatial region,
- a set of k vectors of latent features of length $|\mathcal{S}|$ describing each target spatial region,
- a set of k vectors of latent features of length $|T|$ describing each time epoch, and
- a set of k vectors of latent features of length $|\mathcal{DB}_\tau|$ describing each trend.

These k-dimensional feature vectors can be used to identify mutually similar source spatial regions, mutually similar target spatial regions, mutually similar points in time, and mutually similar trends.

7.3.5 Trend Archetype Clustering

In our first mining step, we identify clusters of mutually similar trends, i.e. trends which have a similar feature vector after the factorization, and thus, since the tensor $\mathcal{F}(\mathcal{DB})$ describes the flow of trends over time, exhibit a similar dissemination over space and time. Each of the resulting clusters is called a trend archetype. This approach allows to classify future trends among all archetypes, and allows to predict the future dissemination of a new trend by using the dissemination model of their archetype.

Definition 14 *Let $\mathcal{DB}_\tau$ be a set of trends, and for each trend $\tau \in \mathcal{DB}_\tau$ let feat(τ) be a set of features describing τ. Further, let $\mathcal{C}(\mathcal{DB}_\tau) = \{C_1, \ldots, C_n\}$ be a clustering of all trends in $\mathcal{DB}_\tau$ into n clusters. Then we denote each cluster $C \in \mathcal{C}$ as an archetype, and all trends $\tau \in C$ are said to belong to the same archetype.*

7.3.6 Trend Archetype Flow Modelling

After the trend clustering step of Section 7.3.5, we can identify sets of trends which belong to the same dissemination archetype. Therefore, we return to the full tensor $\mathcal{F}(\mathcal{DB})$, and for each archetype $C \in \mathcal{C}$, we select only the trends $\tau \in C$, thus yielding a $\mathcal{S} \times \mathcal{S} \times T \times C$ tensor $\mathcal{F}(\mathcal{DB}, C)$ for each archetype C. Using $\mathcal{F}(\mathcal{DB}, C)$, we perform a projection on two modes $\mathcal{S} \times \mathcal{S}$ by averaging over all trends $\tau \in C$ and all epochs $T_i \in T$ to obtain the flow model of archetype C.

7.4 Related Work and Discussion

The problem of event detection in social media streams has attracted much attention in recent years. Ritter et al. [151] developed an event extraction system based on Twitter streams. Using the entity recognition and sequence classification, they extracted a 4-tuple representation of each event, showing the entities, mentions, calendar, and type of each event. Schubert et al. [159] proposed a statistical metric based on the term frequency, and reported an event when there was a large deviation in the metric of a particular term. They applied hierarchical clustering to merge terms that burst together into large-scale topics. In addition to textual information, Kalyanam et al. [100] also considered the communities of users who are interested in certain topics. They applied non-negative matrix factorization (NMF) to incorporate both textual and social information in studying the topic detection and evolution. Abdelhaq et al. [1] used the geotags of tweets to distinguish local events from global phenomena and tracked their spatial distribution over time. Lin et al. [124] applied a Gibbs Random Field model regularized by a topic model to track the popular events in social media. For each evolving event, they reported a stream of text information and a stream of network structures indicating the event diffusion. Weng et al. [192] applied Wavelet Transform to build signals for each word. Then they built a graph based on the cross-correlation of signals and clustered words into events using a modularity-based graph partitioning technique. Sayyadi [158] et al. applied community detection technique to detect events in social streams. They built a graph of words based on their co-occurrence. Then they removed the vertices with high betweenness centrality score and regarded the communities that remained as the keywords for events.

However, none of these works exploited the spatio-temporal characteristics of an event. Unankard et al. [184] extracted user locations and event locations from geo-tagged posts. They defined a location correlation score between user and event locations and used it to identify the hotspot events. Zhou et al. [208] extended the Latent Dirichlet Allocation (LDA) to incorporate the location information of social messages, and proposed a novel location-time constrained topic model. Then they detected events by conducting similarity joins in streams of social messages. Sakaki et al. [154] conducted semantic analysis in user posts to detect natural disasters. They used exponential distribution to study the temporal characteristics of disasters. They used kalman filter and particle filter to predict the spatial trajectories of disasters. From the perspective of query processing, Lappas et al. [116] defined two types of spatio-temporal burstiness patterns, aiming at finding terms which had unusually high frequencies in a spatial region within a particular time interval. Sankaranarayanan et al. [155] developed a news system based on Twitter streams. They used Naïve Bayes Classifier to distinguish valuable news from junk posts and used an algorithm called leader-follower clustering to cluster news into topics.

Table 7.1: Trend Archetypes of 2014

#	Size	Example 1 Keywords	Example 2 Keywords
1	8	mh17 malaysia_crash	ferguson michael_brown riot
2	3	ellen degeneres selfie	robin williams suicide
3	5	whatsapp facebook takeover	supreme_court obergefell hodges
4	10	germany fifa14 brazil	germany fifa14 argentina
5	4	brazil world_cup	ebola
6	12	eu_sanction eu_russia	putin peskov conference
7	1	chile iquique earthquake	-
8	10	flappy_bird removed_appstore	how_I_met_your_mother_finale
9	18	mh370 malaysia_missing	qz8501 air_asia missing
10	14	scotland independence_poll	india bharatiya janata election
11	14	sydney siege hostage	ottawa gunman parliament
12	1	merry chistmas	-

7.5 Experimental Evaluation

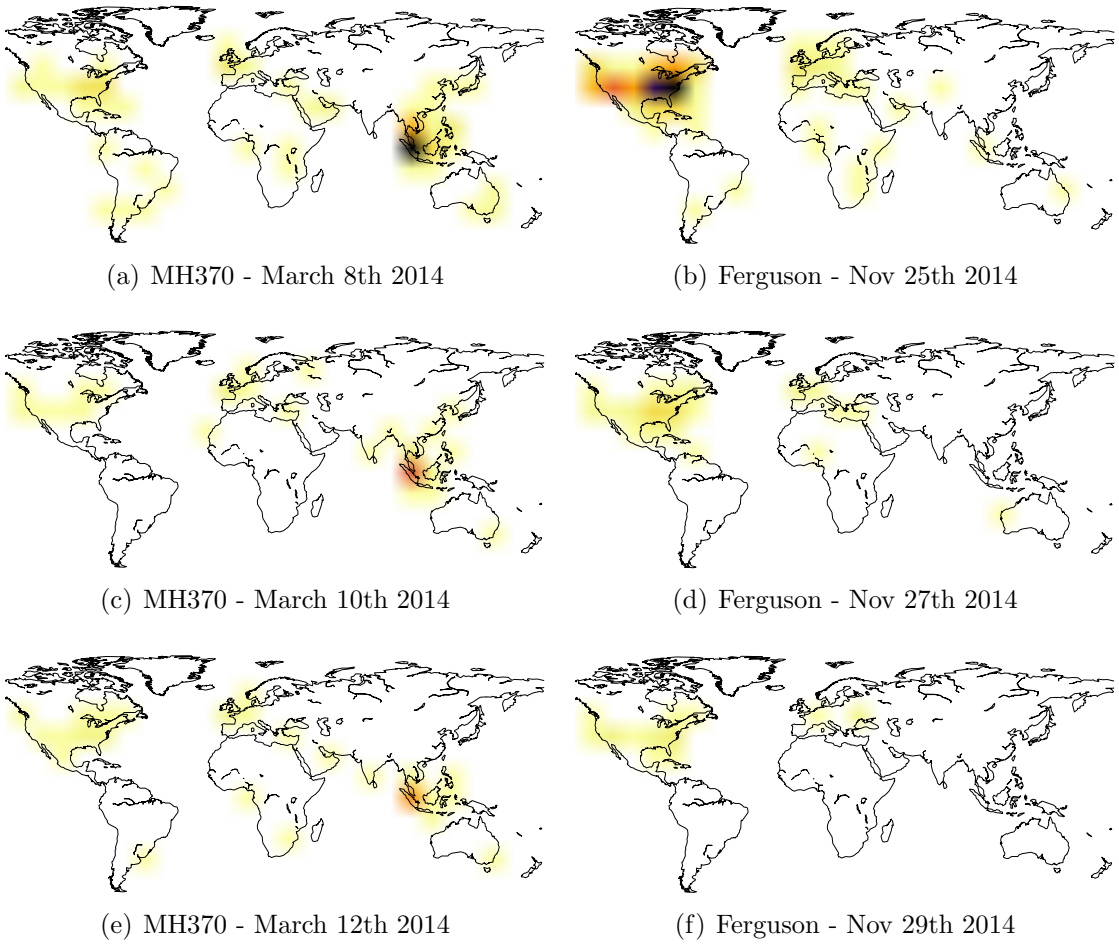
7.5.1 Parameters and dataset

We evaluated our proposed workflow on a dataset mined from Twitter using their public API, feeding from a global 1%-sample over the years 2014 through 2016 (until August of 2016). Out of the tweets returned from the API, we removed those without a geolocation specified. Tweets were aggregated over one-day periods by their UTC-timestamp. The number of tweets per day ranged from around 50,000 to 150,000.

For each trend from the SigniTrend framework, we extracted tweets from one day before and five days after the respective associated date to cover the entire trend dissemination pattern. Unless otherwise specified, each day was subdivided into epochs of six hours to allow for timeshift in different hemispheres. For the majority of our experiments, we used the top-100 trends of the year 2014.

7.5.2 Evaluation of trend archetypes

Table 7.1 depicts some exemplary resulting trend archetypes from data covering the year 2014. Keywords for the top-100 trends were extracted using SigniTrend and used to filter geo-tagged tweets occuring within a 5 day timeframe around the trend date. Underscores



(a) MH370 - March 8th 2014 (b) Ferguson - Nov 25th 2014

(c) MH370 - March 10th 2014 (d) Ferguson - Nov 27th 2014

(e) MH370 - March 12th 2014 (f) Ferguson - Nov 29th 2014

Figure 7.7: Dissemination of trends “MH370” and “Ferguson”

”_” between words denote a boolean conjunction, requiring all connected words to occur in any possible order within one tweet. Spaces between keywords or conjuctions of keywords denote a boolean disjunction. Keywords listed are not exhaustive.

Each line of the table corresponds to a resulting archetype of trends with similar dissemination, resulting from a clustering of the latent feature vector feat(τ). While column “Size” refers to the true cardinality of each cluster, (up to) two examples are given to illustrate the nature of each archetype. Each example lists some keywords for one trend grouped into this archetype.

Some rather interesting results emerge by comparing the keywords to their respective historical events. While archetype #9 contains two trends referring to airplanes going missing without a trace (MH370 in March and QZ8501 in December), another lost airplane is grouped together with riots in the aftermath of a police shooting in the US in archetype #1. Looking at the respective tweet heatmaps in Figure 7.7, a similarity in pattern emerges: a first main event occurs (“plane crashes in Ukraine” vs. “riots after jury decision not

to indict shooter") causing an inital burst mainly in the affected areas (Figures 7.7(a) for MH370 and 7.7(b) for the shooting). After the initial burst, new information sheds different light on the events, making them stand out and causing a more steady flow of messages internationally ("plane grounded by missile" vs. "several people killed as riots spread"). This more steady output can be seen over Figures 7.7(c) and 7.7(e) for MH370 and Figures 7.7(d) and 7.7(f) for the shooting. Bear in mind that the grouping occurred solely based on the numerical features of the respective trends' spatial dissemination, regardless of their content.

Trend archetype #2 grouped some strong international trends themed around society, containing Ellen DeGeneres' selfie picture taken at the Oscar ceremony as well as Robin Williams' sudden suicide. Archetype #3 contains trends with more specialised contents such as financial ("Facebook buys WhatsApp") or judicial ("Obergefell vs. Hodges, Supreme Court deciding on same-sex marriage").

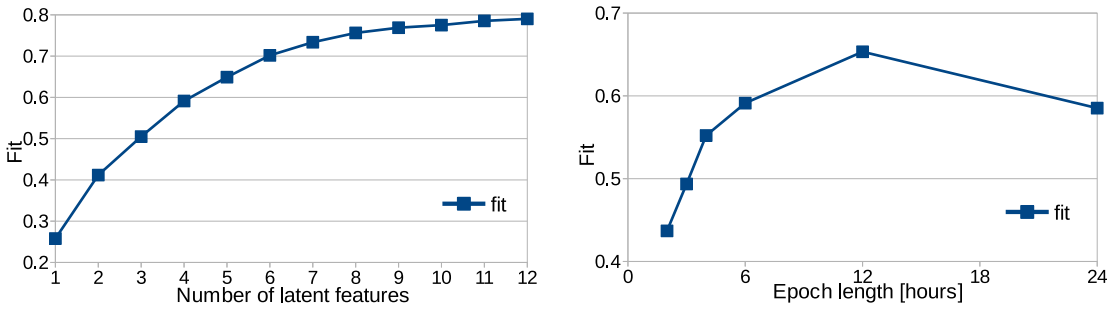
Another destinction is made between archetypes #4 and #5, both containing trends regarding the FIFA world cup 2014 in Brazil: while #4 represents game results and surprising or strong wins, #5 contains the more steady general discussion about the event, as well as other longer–term themes sparking much discussion. Among those is also the repeated outbreak of the Ebola virus in West Africa. Despite the entirely different nature of those topics, both represent a great public interest that dominated news media for longer periods of time.

7.5.3 Evaluation of approximation quality

The tensor decomposition employed in our flow modelling process exhibits a high quality for even low numbers of k, i.e., a small number of latent features per feature vector. This inidicates large eigenvalues of the first k latent features, thus indicating that these features are able to accurately describe the whole tensor with little loss of information. However, some information is still lost compared to an undecomposed tensor. We evaluate the quality of our decomposition by summing up the least-squared error between a reconstruction of the original tensor from its k-feature-vectors, and the original tensor itself. We call the inverse of this error "fit", ranging from 1.0 for an exact match to 0.0 for no correlation.

Figure 7.8(a) shows that for a $k = 4$, the reconstructed tensor matches its original with a fit of 0.6, which is why we chose to set $k = 4$ in all subsequent experiments unless otherwise specified. As can be seen, the gain in fit slows down with additional latent features.

Figure 7.8(b) displays fit for different lengths of trend epochs, the granularity of our analysis in temporal dimension, ranging from 2 hours to 24 hours. The amount of days looked at per trend remained the same, so a longer epoch will result in a smaller number of epochs overall, reducing the size of $\mathcal{F}(\mathcal{DB})$ in the T dimension. Intuitively, a smaller tensor $\mathcal{F}(\mathcal{DB})$ is easier to reconstruct, increasing the fit for longer epochs. However, this does not hold for epochs of 24 hours. We believe this to be due to a counter effect of more diversity in tree cell population as epochs get longer and thus more tweets are grouped in the same epoch. In other experiments, we set the epoch length to 6 hours unless otherwise



(a) Fit for number of latent features.

(b) Fit for length of trend episodes.

Figure 7.8: Approximation fit of factorized tensor.

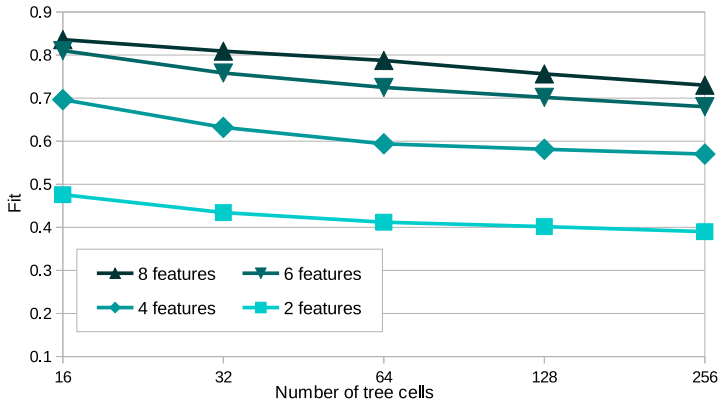


Figure 7.9: Fit over tree cells for varying latent features.

specified – although it is not the peak for fit, we found it to best approximate trends from different global regions, hence being able to compare trends in different hemispheres where peaks happen at different hours in the day.

The effect of varying spatial resolution can be seen in Figure 7.9 for four alternative settings of k. Although the underlying k-d tree is built on global tweet distribution to assure tweets in the same region from different trends are matched to the same cell, varying its node capacity upon indexing results in a higher- or lower resolved spatial grid, hence lowering or increasing the size of $\mathcal{F}(\mathcal{DB})$ in both spatial dimensions. Naturally, a smaller grid is easier to approximate with the same amount of latent features, yet the experiments show that features have a much higher impact on approximation quality than changing spatial resolution. As can be seen, fit values do not deteriorate much for higher numbers of grid cells.

The impact of different numbers of trends $\tau_{K,T}$ is stronger, particularly for smaller k. Figure 7.10 displays fit values for four alternative settings of k and the number of trends ranging from 20 to 100. As in previous experiments, fit decreases as the size of $\mathcal{F}(\mathcal{DB})$ increases. However, for higher k the effect is drastically smaller, maintaining a good approximation quality at the cost of a higher complexity.

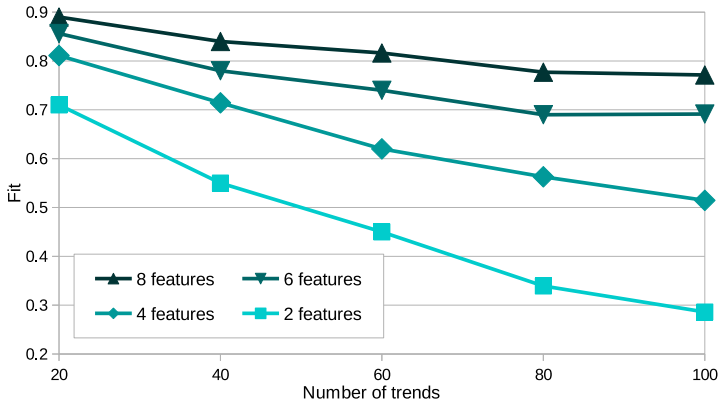


Figure 7.10: Fit over trends for varying latent features.

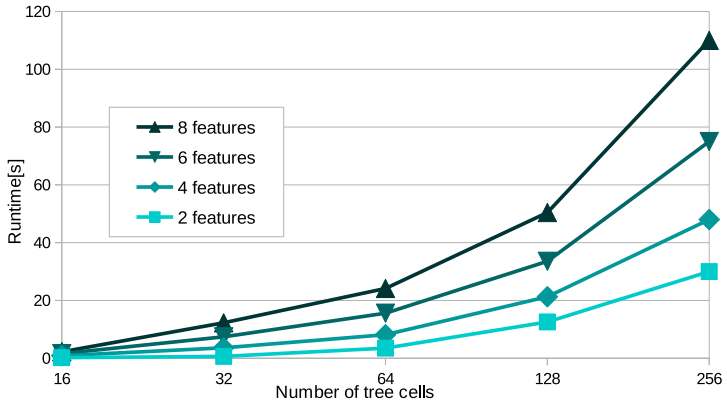


Figure 7.11: Runtime over tree cells for varying latent features.

7.5.4 Evaluation of algorithmic runtime.

The following experiments evaluate runtime of the tensor generation, decomposition and projection on two modes $S \times S$. Filtering of tweets is not included in this evaluation since it depends heavily on the actual keyword settings as well as size of the underlying dataset. All experiments were performed on Arch Linux on an Intel i7 notebook with 16 GB of memory, implemented in the Python language using numpy, pandas, and the sktensor package for tensor decomposition.

Figure 7.11 examines runtime in seconds over spatial resolution, for four different settings of k. Since an increase in the number of tree cells causes a quadradic increase in the size of $\mathcal{F}(\mathcal{DB})$, runtimes scale superlinear for higher spatial resolutions.

The effect of different numbers of trends $\tau_{K,T}$ on runtime is shown in Figure 7.12. Runtimes show only a slight superlinear increase for higher amounts of trends, as the size of $\mathcal{F}(\mathcal{DB})$ increases linearly with trends.

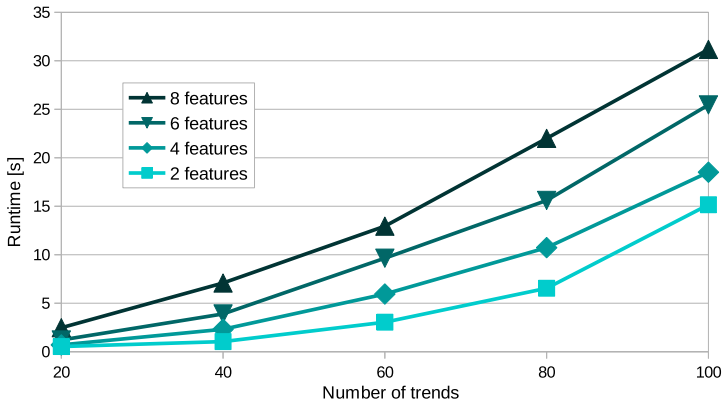


Figure 7.12: Runtime over trends for varying latent features.

7.6 Conclusions

In this work, we studied the dissemination of trends in space and time. For each historic trend, we proposed to constructed a spatio-temporal trend dissemination model, describing the flow of a trend through space and time. By applying a tensor factorization approach, we extracted latent features of trends, to which we applied a clustering approach to obtain sets of trends having a similar dissemination archetype. Our qualitative evaluation of these trend archetypes on Twitter trends show meaningful dissemination archetypes, such as political trends, celebrity trends, and disaster trends. Our quantitative analysis shows that our tensor factorization yields are high approximation quality for a low number of latent features. This result implies that a small number of latent features we derive from the flow of each trend is able to discriminate trends with a high-precision.

The next step of this research direction, is to make our trend flow based classification actionable for decision making. Thus, instead of classifying historic trends, we want to deploy our system in an on-line streaming environment. For this purpose, we want to build a system which observes current and new trends (taken from existing trend mining solutions such as SigniTrend [159]), to classify the archetype of a trend as soon as possible, thus allow to predict the spatio-temporal dissemination of trend. If successful, this approach will allow us to predict the regional news of tomorrow, today.

Part III

Uncertain Enriched Geo-Spatial Data

The following part of this thesis surveys the field of modeling, querying, and mining geo-spatial data enriched with uncertainty models. It is subdivided into the following chapters:

- Chapter 8 introduces the concepts and models used in this thesis for handling uncertain data.

- After the introductory part, Chapter 9 introduces techniques to issue *(k)-Nearest-Neighbor queries on uncertain data using Voronoi decomposition*, where location, shape and extent of Voronoi cells are random variables. To facilitate reliable query processing despite the presence of uncertainty, we employ the concept of *possible-Voronoi cells* and introduce the novel concept of *guaranteed-Voronoi cells*: The possible-Voronoi cell of an object U consists of all points in space that have a non-zero probability of having U as their nearest-neighbor; and the guaranteed-Voronoi cell, which consists of all points in space which must have U as their nearest-neighbor.

  The chapter is based a paper published in 14th International Symposium on Spatial and Temporal Databases *SSTD 2015* [58] and was submitted in an extended version as an invited contribution to *GeoInformatica Journal* "SSTD 2015 Best Paper Special Issue" (accepted with minor revision as of August 2016).

- As one application of mining uncertain data, a novel approach to clustering uncertain data is presented in Chapter 10: a framework based on possible world semantics computes a set of representative clusterings, each of which with probabilistic guarantees not to exceed some maximum distance to the ground truth clustering, i.e., the clustering of the actual (but unknown) data. This framework can employ any existing clustering algorithm. Evaluation shows that in addition to providing quality guarantees, these representative clusterings have a much smaller deviation from ground truth than existing appproaches.

  The presented work was published in 20th SIGKDD Conference on Knowledge Discovery and Data Mining *KDD 2014* [211].

- Chapter 11 applies the methodology of Chapter 10 to query processing on spatial data. The presented framework draws deterministic samples from the uncertain database and applies traditional query processing methods of the user's choice to them. From the set of answers, a user-chosen number of representative answers is generated with associated quality guarantees.

Chapter 8

Models and Techniques

8.1 Uncertain Data: Introduction

In a variety of application domains, our ability to unearth a wealth of new knowledge from a data set is impaired by unreliable, erroneous, obsolete, imprecise, and noisy data. Reasons and sources of such uncertainty are many. Sensing devices are inherently imprecise (e.g., due to signal noise, instrumental errors, and transmission errors [49]). Moving objects can only be monitored sporadically, such that at a certain time the position of an object is not explicitly known [182]. Integration of data from heterogeneous sources may incur uncertainty, for example due to uncertain schema matchings between different data sources [5]. Uncertainty may also be injected to the data on purpose, for privacy preservation reasons [66]. Finally, some of the data may be plain wrong, for instance in Web 3.0 applications where data is collected by humans, which may deliberately or accidently spread misinformation [119].

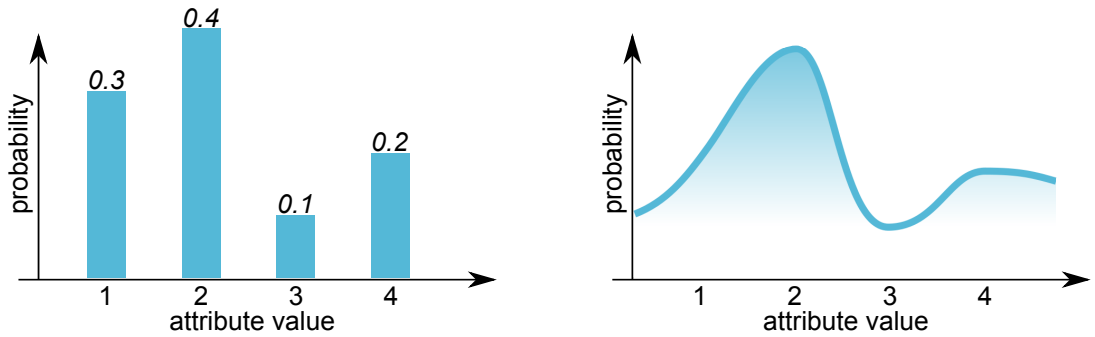
Traditionally, the problem of uncertainty has been addressed by ignoring it. Therefore, the implicit assumption was made that all data records of a given data set are precise, reliable, and up-to-date. Making such assumptions in any context having significant uncertainty, any querying or data mining task may yield skewed and even wrong results. The research field of uncertain database management cannot solve this problem, as there exists no oracle that is able to magically transform measured or observed values stored in the database into their (unknown) true values of their corresponding universe of discourse.

On the other hand, by considering the uncertainty as added information enriching the underlying dataset in querying and data mining, we can assess the reliability of the result, giving the user a notion about its quality and giving an intuition of how likely it is identical, or at least similar, to the result of the mining task when applied to the true (but unknown) data values [179]. For instance, in association rule mining on uncertain data, confidence values of the probability that a given itemset is frequent are derived [18]. This notion of confidence allows the user to make a more educated judgement of the data, thus enhancing the underlying decision-making process.

8.2 Modelling Uncertain Data: Preliminaries

An object o is uncertain if at least one of its attributes is uncertain. Attribute uncertainty can be represented either in a discrete model or a continuous model. A discrete model describes an attribute value using a probability mass function (pmf), where a finite number of alternative instances is associated with its respective probability [112, 144], as shown in Figure 8.1(a). Continuous models such as the example shown in Figure 8.1(b), on the other hand, represent attribute values using a continuous probability density function (pdf) such as Gaussian, uniform, Zipfian, or a mixture model. Therefore, in a continuous model, the number of distinct possible attribute values is uncountably infinite. The probability of an attribute value falling within an interval can be estimated by integration of the object's pdf over said interval [180]. The random variables corresponding to each uncertain attribute of an object o can be arbitrarily correlated.

Positional data from geospatial applications is often associated with uncertainty, e.g., location data from GPS modules. To capture this, latitude and longitude (and optionally, elevation) are treated as two (three) uncertain attributes. In the case of discrete positional uncertainty, the position of an object A is given by a discrete set $a_1, ..., a_m$ of $m \in \mathbb{N}$ possible alternatives in space, as exemplarily depicted in Figure 8.2(a) for two uncertain objects A and B. Each alternative a_i is associated with a probability value $p(a_i)$, which may for example be derived from empirical information about the turn probabilities of intersection in an underlying road network. In a nutshell, the position A is a random variable, defined by a probability mass function pdf_A that maps each alternative position a_i to its corresponding probability $p(a_i)$, and that maps all other positions in space to a zero probability. An important property of uncertain spatial databases is the inherent correlation of spatial attributes. In the example shown in Figure 8.2(a) it can be observed that the uncertain attributes *Latitude* and *Longitude* are highly correlated: given the value of one attribute, the other attribute is certain, as there is no two alternatives of objects A and B having identical attribute values in either attribute.



(a) Discrete Probability Mass Function

(b) Continuous Probability Density Function

Figure 8.1: Models for Uncertain Attributes

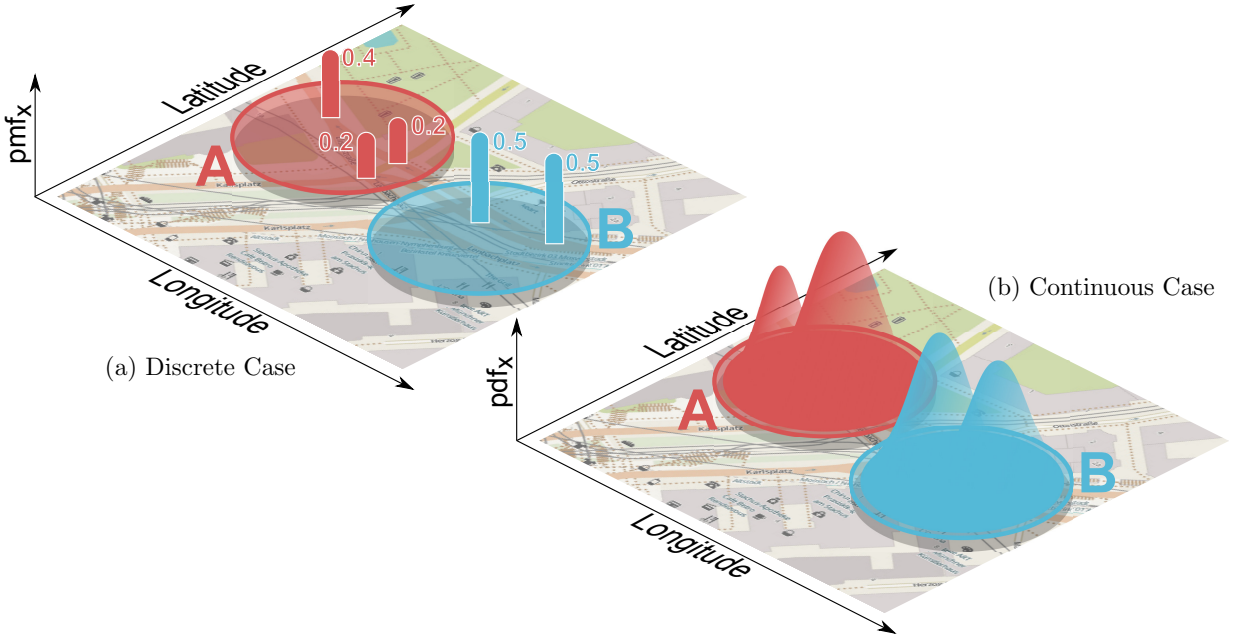


Figure 8.2: Uncertain Objects

Intuitively, it must hold that the sum of probabilities of all alternatives must sum to at most one:

$$\sum_{i=1}^{m} p(a_i) \leq 1$$

In the case where $\sum_{i=1}^{m} p(a_i) \leq 1$ object A has a non-zero probability of $1 - \sum_{i=1}^{m} p(a_i) \geq 0$ to not exist at all. This case is called *existential uncertainty*, and A is referred to as *existentially uncertain* [203]. Should the total number of possible instances m exceed one, A is denoted as *attribute uncertain*. In the context of uncertain spatial data, attribute uncertainty is also referred to as *positional uncertainty* or *location uncertainty*. An object can be both existentially uncertain and attribute uncertain. In Figure 8.2(a), object A is both existentially uncertain and attribute uncertain, while object B is attribute uncertain while is existence is not uncertain.

In the case of continuous uncertainty, the number of possible alternative positions of an object A is infinite, and given by the non-zero domain of the probability density function pdf_x. The probability of A to occur in spatial region r is given by integration

$$\int_r pdf_A(x)dx.$$

Since arbitrary pdfs may be represented by an infinitely large number of $(position, probability)$ pairs, such pdfs may require infinite space to represent. For this reason, assumptions on the shape of a pdf are made in practice. All continuous models for positionally uncertain data therefore use parametric pdfs, such as Gaussian, uniform, Zipfian, mixture models,

or parametric spline representations. For illustration purpose, Figure 8.2(b) depicts two uncertain objects modelled by a mixture of gaussian pdfs. Similar to the discrete case, the constraint

$$\int_{\mathcal{R}^d} pdf_A(x)dx \leq 1$$

must be satisfied, where $\mathcal{R}^d$ is a d dimensional vector space. In the case of spatial data, d would typically equal two or three. The notion of existentially and attribute uncertain objects is defined analogous to the discrete case.

The following section reviews related work and state-of-the-art on the field of modeling uncertain data.

8.3 Uncertain Database Models

This section gives a brief survey on existing models for uncertain spatial data used in the database community. Many of the presented models have been developed to model uncertainty in relational data, but can be easily adapted to model uncertain spatial data. Since one of the main challenges of modeling uncertain data is to capture correlation between uncertain objects, this section will elaborate details on how state-of-the-art approaches tackles this challenge. Both discrete and continuous models are presented.

8.3.1 Discrete Models

In addition to reviewing related work defining discrete uncertainty models, the aim of this section is to put these papers into context of Section 8.2. In particular, models which are special cases or equivalent to the model presented in Section 8.2 will be identified, and proper mappings will be given.

Independent Tuple Model.

Initial models have been proposed simultaneously and independently in [62, 209]. These works assume a relational model in which each tuple is associated with a probability describing its existential uncertainty. All tuples are considered independent from each other. This simple model can be seen as a special case of the model presented in Section 8.2, where only existential uncertain but no attribute uncertainty is modelled.

Block-Independent Disjoint Tuples Model and X-Tuple model

A more recent and the currently most prominent approach to model discrete uncertainty is the block-independent disjoint tuples model ([45]), which can capture mutual exclusion between tuples in uncertain relational databases. A probabilistic database is called block independent-disjoint if the set of all possible tuples can be partitioned into blocks such that tuples from the same block are disjoint events, and tuples from distinct blocks are independent. A commonly used example of a block-independent disjoint tuples model is

the *Uncertainty-Lineage Database Model*([15, 157, 175, 202]), also called *X-Relation Model* or simply *X-Tuple Model* that has been developed for relational data. In this model, a probabilistic database is a finite set of *probabilistic tables*. A probabilistic table T contains a set of (uncertain) tuples, where each tuple $t \in T$ is associated with a membership probability value $Pr(t) > 0$. A *generation rule* R on a table T specifies a set of mutually exclusive tuples in the form of $R : t_{r_1} \oplus ... \oplus t_{r_m}$ where $t_{r_i} \in T(1 \leq i \leq m)$ and $P(R) := \sum_{i=1}^{m} t_{r_i} \leq 1$. The rule R constrains that, among all tuples $t_{r_1}, ..., t_{r_m}$ involved in the rule, at most one tuple can appear in a possible world. The case where $P(R) < 1$ the probability $1 - P(R)$ corresponds to the probability that no tuple contained in rule R exists. It is assumed that for any two rules R_1 and R_2 it holds that R_1 and R_2 do not share any common tuples, i.e., $R_1 \cap R_2 = \emptyset$. In this model, a possible world w is a subset of T such that for each generation rule R, w contains exactly one tuple involved in R if $P(R) = 1$, or w contains 0 or 1 tuple involved in R if $Pr(R) < 1$.

This model can be translated to a discrete model for uncertain spatial data as discussed in Section 8.2 by interpreting the set T as the set of all possible locations of all objects, and interpreting each rule R as an uncertain spatial object having alternatives t_{r_i}. The constraint that no two rules may share any common tuples translates into the assumption of mutually independent spatial objects. Finally, the case $P(R) < 1$ corresponds to the case of existential uncertainty.

A similar block-independent disjoint tuples model is called *p-or-set* [148] and can be translated to the model described in Section 8.2 analogously. In [10], another model for uncertainty in relational databases has been proposed that allows to represent attribute values by sets of possible values instead of single deterministic values. This work extends relational algebra by an operator for computing possible answers. A normalized representation of uncertain attributes, which essentially splits each uncertain attribute into a single relation, a so-called U-relation, allows to efficiently answer projection-selection-join queries. The main drawback of this model is that it is not possible to compute probabilities of the returned possible answers. Sen and Deshpande [161] propose a model based on a probabilistic graphical model, for explicitly modeling correlations among tuples in a probabilistic database. Strategies for executing SQL queries over such data have been developed in this work. The main drawback of using the proposed graphical model is its complexity, which grows exponential in the number of mutually correlated tuples. This is a general drawback for graphical models such as Bayesian networks and graphical Markov models, where even a *factorized representation* may fail to reduce the complexity sufficiently: The idea of a factorized representation is to identify conditional independencies. For example, if a random variable C depends on random variables A and B, then the distribution of C has to be given relative to all combination of realizations of A and B. If however, C is conditionally independent of A, i.e., B depends on A, C depends on B, and C only transitively depends on A, then it is sufficient to store the distribution of C relative only to the realizations of B. Nevertheless, if for a given graphical model a random variable depends on more than a hand-full of other random variables, then the corresponding model will become infeasible.

And/Xor Tree Model.

A very recent work by Li and Deshpande [120] extends the block-independent disjoint tuples model by adding support for mutual co-existence. Two events satisfy the mutual co-existence correlation if in any possible world, either both happen or neither occurs. This work allows both mutual exclusiveness and mutual co-existence to be specified in a hierarchical manner. The resulting tree structure is called an *and/xor tree*. While theoretically highly relevant, the and/xor tree model becomes impracticable in large database having non-trivial object dependencies, as it grows exponentially in the number of database objects.

8.3.2 Continuous Models

In general, similarity search methods based on continuous models involve expensive integrations of the PDFs, hence special approximation and indexing techniques for efficient query processing are typically employed [38, 180]. In order to increase quality of approximations, and in order to reduce the computational complexity, a number of models have been proposed making assumptions on the shape of object PDFs. Such assumptions can often be made in applications where the uncertain values follow a specific parametric distribution, e.g. a uniform distribution [36, 34] or a Gaussian distribution [34, 49, 143]. Multiple such distributions can be mixed to obtain a mixture model [183, 22]. To approximate arbitrary PDFs, [121] proposes to use polynomial spline approximations.

8.4 Uncertain Database Systems

The problem of managing uncertain data has been well-studied by the database research community in the past. While traditional database literature [28, 13, 12, 115, 62] has studied the problem of managing uncertain data, this research field has seen a recent revival, due to modern techniques for collecting inherently uncertain data. Consequently, novel concepts for managing uncertain data have been developed. Most prominent concepts for probabilistic data management are MayBMS [10], MystiQ [23], Trio [4], and BayesStore [189]. These uncertain database management systems (UDBMS) provide solutions to cope with uncertain relational data, allowing to efficiently answer traditional selection/projection/join(SPJ)-queries. Extensions to the UDBMS also allow to answer important classes of spatial queries such as top-k and distance-ranking queries [83, 41, 122, 17, 121]. However, these existing UDBMS offer no support for data mining tasks. A likely reason for this gap is the theoretic result of [44] which shows that the problem of answering complex queries is #P-hard in the number of database objects. Consequently, approximate solutions are used in the MCDB system [93] allows to find approximate query answers, by sampling possible worlds from the database. However, for complex tasks such as data mining tasks, the problem arises of how to combine a large set of possible data mining results, obtained from each database sample. Answering this non-trivial question is an open research question.

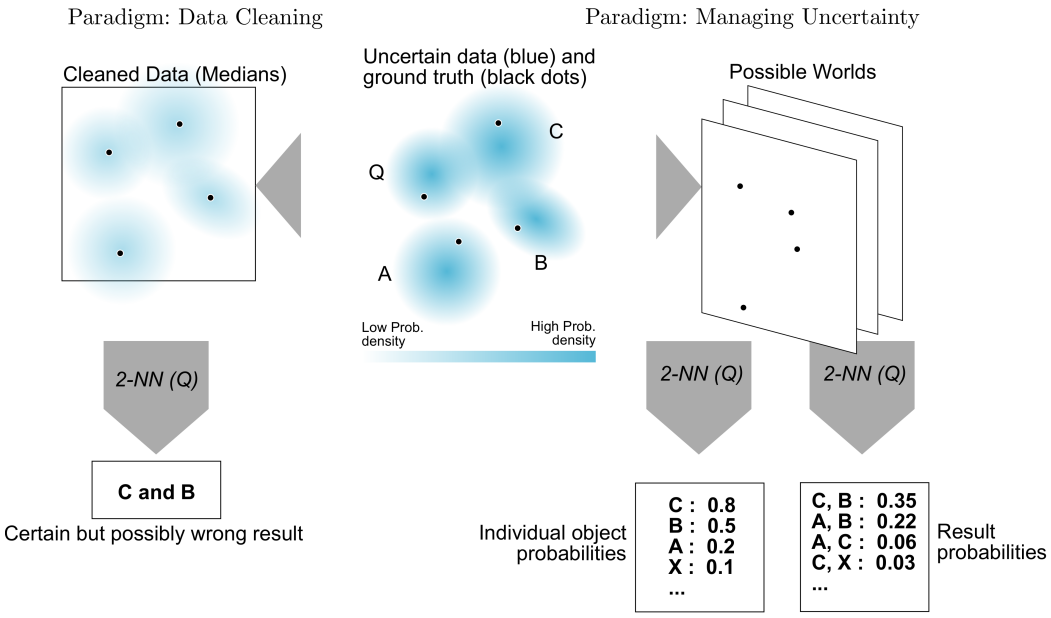


Figure 8.3: Different Uncertain Query Workflows.

8.5 Handling Uncertain Data

Two traditional paradigms have manifested to handle uncertain data – illustrated on an excerpt from an uncertain database in Figure 8.3. Here, the exact position of each object is a random variable, such as a two-dimensional Gaussian. The white dots illustrate the unknown ground-truth position of objects. Note that in an uncertain database, the ground-truth attribute values of an object are not known for certain - but the probability for any point (or region) to be equal to (or contain) the ground-truth is derived from probability density functions of objects. In this example, assume we want to perform a two-nearest-neighbor query for query object Q. The correct ground-truth result is A and C. To find this result in the presence of uncertainty, the following two general paradigms are commonly used.

Data Cleaning. In the first paradigm a cleaning step is performed on the uncertain data yielding a certain database. Queries can then be processed as if the uncertainty never existed. Obviously, this approach has two grave disadvantages. First, the quality of the result is highly dependent on quality of the cleaning step. However, data cleaning is a non-trivial application dependent task and inherently suffers from loss of information. This loss of (uncertainty) information may compromise the quality of the result. Second, the result has no indication of the quality of the result. Using this paradigm, there is no notion of significance or reliability associated with the results, thus it becomes impossible to assess the likelihood of this result being identical, or even just similar, to the ground-truth. Being unable to assess the similarity to the (unknown) ground-truth, educated decision making based on these results is jeopardized. A possible cleaning step is shown in Figure 8.3 (middle-left), where the center of mass of an object pdf is assumed to be the real position of the object. The result based on this cleaning is then deterministic (C and B).

Managing Uncertainty The second paradigm enriches query results by probability values indicating the likelihood of any individual result to be part of the (unknown) ground-truth. Such *probabilistic approaches* allow the user to assess the quality of the results returned to the user, thus enhancing the underlying decision-making process. The likelihood of a query result is defined by possible world semantics: Given any uncertainty model, the probability of a query result r is the sum of likelihoods of all possible databases where the query returns r. For our example, possible query results and their probabilities are shown in Figure 8.3 (on the lower-right). Note, that the database of this example is much larger than illustrated, thus also other objects (i.e., E, F, K and X) might be in the result. The drawback of any such approach is an exponential run-time of general query processing on uncertain data, as shown in [44]. Consequently, the research community has relaxed the problem of answering queries on uncertain data: Rather than assigning a probability of each possible result, research has focused on computing the probabilities of individual objects to be part of the query result [93, 181, 19, 37, 112, 34, 86, 35, 123, 20, 41, 174, 202, 122, 17] (cf. Figure 8.3). In the following, we denote solutions to the relaxed version of the problem retrieving individual object result probabilities as an *individual object result* approach. Solutions to the harder problem of returning probabilities of whole result sets are called *full-result* approach.

8.6 Possible World Semantics

In an uncertain spatial database $\mathcal{DB} = \{U_1, ..., U_N\}$, the location of an object is a random variable. Consequently, if there is at least one uncertain object, the data stored in the database becomes a random variable. To interpret, that is, to define the semantics of a database that is, in itself, a random variable, the concept of *possible worlds* is described in this section.

Definition 15 (Possible World Semantics) *A possible world $w = \{u_1^{a_1}, ..., u_N^{a_N}\}$ is a set of instances containing at most one instance $u_i^{a_i} \in U_i$ from each object $U_i \in \mathcal{DB}$. The set of all possible worlds is denoted as $\mathcal{W}$. The total probability of an uncertain world $P(w \in \mathcal{W})$ is derived from the chain rule of conditional probabilities:*

$$P(w) := P(\bigwedge_{u_i^{a_i} \in w} U_i = u_i^{a_i}) = \prod_{i=1}^{N} P(u_i^{a_i} | \bigwedge_{j<i} u_j^{a_j}). \tag{8.1}$$

By definition, all worlds w having a zero probability $P(w) = 0$ are excluded from the set of possible worlds $\mathcal{W}$. Equation 8.1 can be used if conditional probabilities of the position of objects given the position of other objects are known, e.g. by a given graphical model such as a Bayesian network or a Markov model. In many applications where independence between object locations can be assumed, as well as in applications where only the marginal probabilities $P(u_i^{a_i})$ are known, and thus independence has to be assumed due to lack of better knowledge of a dependency model, the above equation simplifies to

$$P(w) = \prod_{i=1}^{N} P(u_i^{a_i}). \tag{8.2}$$

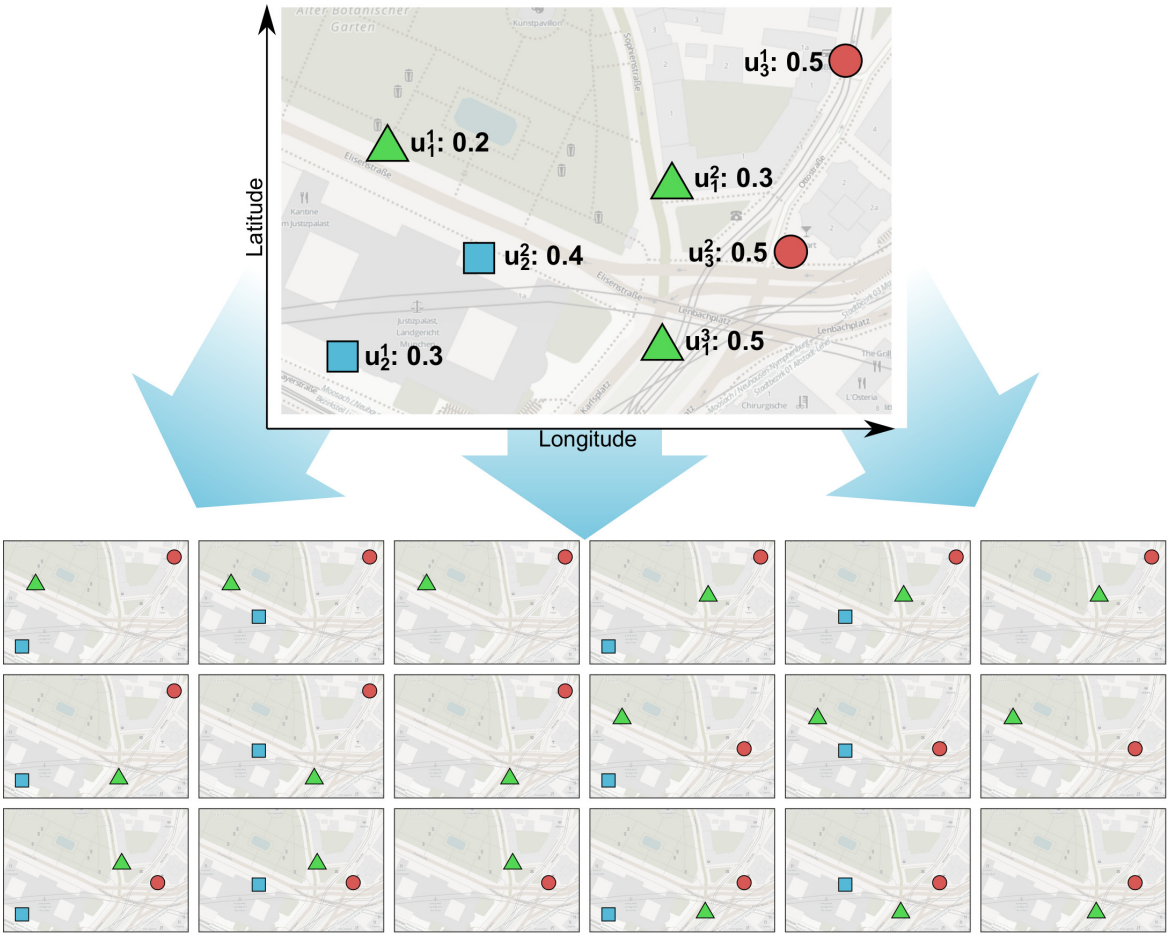


Figure 8.4: An uncertain database and all of its possible worlds.

Example 2 *As an example, consider Figure 8.4 where a database consisting of three uncertain objects* $\mathcal{DB} = \{U_1, U_2, U_3\}$ *is depicted. Objects* $U_2 = \{u_2^1, u_2^2\}$ *and* $U_3 = \{u_3^1, u_3^2\}$ *each have two possible instances, while object* $U_1 = \{u_1^1, u_1^2, u_1^3\}$ *has three possible instances. The probabilities of these instances is given as* $P(u_1^1) = 0.2$, $P(u_1^2) = 0.3$, $P(u_1^3) = 0.5$, $P(u_2^1) = 0.3$, $P(u_2^2) = 0.4$, $P(u_3^1) = P(u_3^2) = 0.5$. *Note that object* U_2 *is the only object having existential uncertainty: With a probability of* $1 - 0.3 - 0.4 = 0.3$ *object* U_2 *does not exist at all. Assuming independence between spatial objects, the probability for the possible world where* $U_1 = u_1^1$, $U_2 = u_2^1$ *and* $U_3 = u_3^1$ *is given by applying Equation 8.2 to obtain the product* $0.2 \cdot 0.3 \cdot 0.5 = 0.03$. *All possible worlds spanned by* $\mathcal{DB}$ *are depicted in Figure 8.4. The probability of each possible world is shown in Table 8.1, including possible worlds where* U_2 *does not exist.*

Predicates can evaluate to logical values true or false on deterministic (non-uncertain) databases. An example for such predicates is *There are at least five relevant database objects in a 500 meter range around the location "Munich Airport"*. To evaluate a predicate φ on an uncertain database using possible world semantics, the query predicate is evaluated on

World	Probability	World	Probability
$\{u_1^1, u_2^1, u_3^1\}$	$0.2 \cdot 0.3 \cdot 0.5 = 0.03$	$\{u_1^1, u_2^1, u_3^2\}$	$0.2 \cdot 0.3 \cdot 0.5 = 0.03$
$\{u_1^1, u_2^2, u_3^1\}$	$0.2 \cdot 0.4 \cdot 0.5 = 0.04$	$\{u_1^1, u_2^2, u_3^2\}$	$0.2 \cdot 0.4 \cdot 0.5 = 0.04$
$\{u_1^1, u_3^1\}$	$0.2 \cdot 0.3 \cdot 0.5 = 0.03$	$\{u_1^1, u_3^2\}$	$0.2 \cdot 0.3 \cdot 0.5 = 0.03$
$\{u_1^2, u_2^1, u_3^1\}$	$0.3 \cdot 0.3 \cdot 0.5 = 0.045$	$\{u_1^2, u_2^1, u_3^2\}$	$0.3 \cdot 0.3 \cdot 0.5 = 0.045$
$\{u_1^2, u_2^2, u_3^1\}$	$0.3 \cdot 0.4 \cdot 0.5 = 0.06$	$\{u_1^2, u_2^2, u_3^2\}$	$0.3 \cdot 0.4 \cdot 0.5 = 0.06$
$\{u_1^2, u_3^1\}$	$0.3 \cdot 0.3 \cdot 0.5 = 0.045$	$\{u_1^2, u_3^2\}$	$0.3 \cdot 0.3 \cdot 0.5 = 0.045$
$\{u_1^3, u_2^1, u_3^1\}$	$0.5 \cdot 0.3 \cdot 0.5 = 0.075$	$\{u_1^3, u_2^1, u_3^2\}$	$0.5 \cdot 0.3 \cdot 0.5 = 0.075$
$\{u_1^3, u_2^2, u_3^1\}$	$0.5 \cdot 0.4 \cdot 0.5 = 0.1$	$\{u_1^3, u_2^2, u_3^2\}$	$0.5 \cdot 0.4 \cdot 0.5 = 0.1$
$\{u_1^3, u_3^1\}$	$0.5 \cdot 0.3 \cdot 0.5 = 0.075$	$\{u_1^3, u_3^2\}$	$0.5 \cdot 0.3 \cdot 0.5 = 0.075$

Table 8.1: Possible worlds corresponding to Figure 8.4.

each possible world. The probability that the query predicate evaluates to true is defined as the sum of probabilities of all worlds where φ is satisfied, formally:

Definition 16 *Let $\mathcal{DB}$ be an uncertain spatial database inducing the set of possible worlds $\mathcal{W}$, let φ be some query predicate, and let*

$$\mathcal{I}(\varphi, w \in \mathcal{W}) := P(\varphi(\mathcal{DB}) | \mathcal{DB} = w) \in \{0, 1\}$$

be the indicator function that returns one if world w satisfies φ and zero otherwise. The marginal probability $P(\varphi(\mathcal{DB}))$ of the event $\varphi(\mathcal{DB})$ that predicate φ holds in $\mathcal{DB}$ is defined as using the theorem of total probability [212]:

$$P(\varphi(\mathcal{DB})) = \sum_{w \in \mathcal{W}} \mathcal{I}(\varphi, w) \cdot P(w). \tag{8.3}$$

8.7 Approximate Queries

The main challenge of analyzing uncertain data is to efficiently and effectively deal with the large number of possible worlds induced by an uncertain database $\mathcal{DB}$. In the case of continuous uncertain data, the number of possible worlds is uncountably infinite, even in the case where the database contains only one uncertain object. Thus, expensive integration operations or numerical approximation are required for most spatial database queries and spatial data mining tasks. Even in the case of discrete uncertainty, the number of possible worlds grows exponentially in number of objects: in the worst case, any combination of alternatives of objects may have a non-zero probability, as shown exemplary in Figure 8.4.

This large number of possible worlds makes efficient query processing and data mining an extremely challenging problem. In particular, any problem that requires an enumeration of all possible worlds is $\#P$-hard. In particular, a number of probabilistic problems have been proven to be in $\#P$ [185]. Following this argumentation, general query processing in the case of discrete data using object independence has proven to be a $\#P$-hard problem [44] in the context of relational data. The spatial case is a specialization of the relation

case, but clearly, the spatial case is in $\#P$ as well, which becomes evident by construction of a query having an exponentially large result, such as the query that returns all possible worlds. Consequently, there can be no universal solution that allows to answer *any* query in polynomial time. This implies that querying processing on models that are generalizations of the discrete case with object independence, e.g., models using continuous distribution, or models that relax the object independence assumption, must also be a $\#P$ hard problem. For such queries, an analytical solution is infeasible. Furthermore, even queries that can be answered in $PTIME$ may be too slow in practice. Yet, exact result probabilities are not required in many applications. A good approximation of result probabilities with probabilistic guarantees may suffice in such applications.

In the case of uncertain data, two fundamental semantics to return possible results can be distinguished and will be referred to as *object based answer semantics* and *result based answer semantics*, respectively. Informally, the former semantics return possible result objects and their probability of being part of the result, while the later semantics return possible results, consisting of one or more objects, and their probability of being the result as a whole. A formal introduction of these concepts can be found in [149].

8.7.1 Monte Carlo Algorithms

Using object based query semantics, each object $o \in \mathcal{DB}$ has a probability $P(o \in \varphi(\mathcal{DB})) \in [0,1]$ to satisfy the spatial query predicate. Thus, the event that o satisfies φ is a binomial random variable, having a probability of $P(o \in \varphi(\mathcal{DB}))$ to be true, and a probability of $1-P(o \in \varphi(\mathcal{DB}))$ to be false. To estimate the a-priori unknown probability $P(o \in \varphi(\mathcal{DB}))$, random worlds can be instantiated. For a given possible world $w \in \mathcal{W}$, a traditional (non-uncertain) φ query can be executed to decide whether $o \in \varphi(q, w)$.

Definition 17 (Monte-Carlo Approach) *Let $P\psi\varphi(q,\mathcal{DB})$ be a spatial query, where $\mathcal{DB}$ is an uncertain spatial database, q is a query point, φ is a spatial query predicate and ψ is a probabilistic query predicate. A* Monte-Carlo *approach creates a multi-set $\mathcal{S} = \{w_1, ..., w_{|\mathcal{S}|}\}, w_i \in \mathcal{W}$ of randomly and independently sampled possible worlds.*[1] *The spatial query predicate $\varphi(q, w)$ is evaluated on each possible world. The estimator*

$$\hat{P} := \frac{\sum_{w \in \mathcal{S}} f_{ind}(o \in \varphi(q, w))}{|\mathcal{S}|}. \tag{8.4}$$

yields an unbiased estimator of the probability $P(o \in \varphi(q, \mathcal{DB}))$, where $f_{ind}(o \in \varphi(q, w))$ is an indicator function that returns one if $o \in \varphi(q, w)$ and zero otherwise.

Note that $\hat{P}$ is a random variable, since each world $w \in S$ is chosen at random, each having a probability of $P(o \in \varphi(q, \mathcal{DB}))$ to satisfy φ. In particular $\hat{P}$ is the relative fraction of successful Bernoulli trials out of a total of $|\mathcal{S}|$ Bernoulli trials, each having a

[1] This implies that the same world may be sampled in $\mathcal{S}$ more than once. Thus $\mathcal{S}$ is defined as a multi-set rather than a set, to allow duplicate elements.

success probability of $P(o \in \varphi(q, \mathcal{DB}))$. Consequently, $\hat{P}$ follows a Binomial $B(|\mathcal{S}|, P(o \in \varphi(q, \mathcal{DB})))$ distribution.

Since the number of possible worlds increases exponential in the number of uncertain objects in $\mathcal{DB}$, any probabilistic φ query algorithm that explicitly considers each possible world individually will be inapplicable to handle databases of non-trivial size. A Monte-Carlo algorithm entirely avoids to use any expensive probabilistic query processing algorithm. Instead, any non-uncertain φ query processing algorithm can be utilized to compute the indicator function $f_{ind}(o \in \varphi(q, w))$. This has to be repeated a total of $|\mathcal{S}|$ times leading to a total time complexity of $O(|S| \cdot \rho)$, where ρ is the time complexity of the problem of answering a spatial φ query on (certain) point data.

Lemma 6 *If the sample of possible worlds S is drawn in an unbiased way, then the Monte-Carlo estimator $\hat{P}$ is unbiased, i.e.,*

$$E(\hat{P}) = P(o \in \varphi(q, \mathcal{DB})),$$

where $P(o \in \varphi(q, \mathcal{DB}))$ is the probability that an object o is in the result of a spatial query predicate φ having query object q on database $\mathcal{DB}$.

Proof 6 *Using Definition 17 we get*

$$E(\hat{P}) = E(\frac{\sum_{w \in \mathcal{S}} f_{ind}(o \in \varphi(q, w))}{|\mathcal{S}|}).$$

Exploiting linearity of expectation we get

$$E(\hat{P}) = \sum_{w \in \mathcal{S}} \frac{E(f_{ind}(o \in \varphi(q, w)))}{|\mathcal{S}|} \tag{8.5}$$

By definition of the expected value as $E(X) = \sum_x x \cdot P(x)$, and by assuming that any world w is sampled randomly without any bias we get

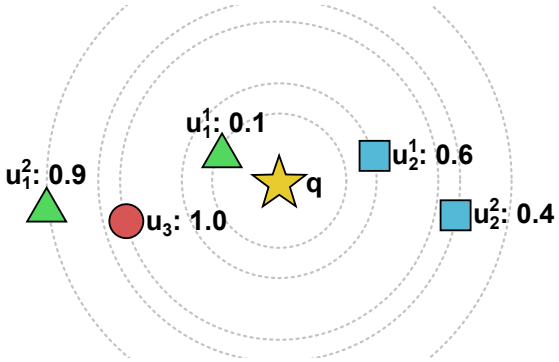
$$E(f_{ind}(o \in \varphi(q, w)) = \sum_{w \in \mathcal{W}} f_{ind}(o \in \varphi(q, w)) \cdot P(w)$$

By definition of possible world semantics (Definition 16), it holds that the sum on the right hand side of the above equation is equal to $P(o \in \varphi(q, \mathcal{DB}))$. This yields

$$E(f_{ind}(o \in \varphi(q, w)) = P(o \in \varphi(q, \mathcal{DB})) \tag{8.6}$$

Substitution of Equation 8.6 in Equation 8.5 yields

$$E(\hat{P}) = \sum_{w \in \mathcal{S}} \frac{P(o \in \varphi(q, \mathcal{DB}))}{E(|\mathcal{S}|)}$$



World	1st	2nd	3rd	Prob
u_1^1, u_2^1, u_3	U_1	U_2	U_3	0.04
u_1^1, u_2^2, u_3	U_1	U_3	U_2	0.06
u_1^2, u_2^1, u_3	U_2	U_3	U_1	0.54
u_1^2, u_2^2, u_3	U_3	U_2	U_1	0.36

Figure 8.5: Example Database showing possible positions of uncertain objects and their corresponding probabilities.

Since the expected value $E(|\mathcal{S}|)$ *of a constant* $|\mathcal{S}|$ *equals* $|\mathcal{S}|$ *we get*

$$E(\hat{P}) = \sum_{w \in \mathcal{S}} \frac{P(o \in \varphi(q, \mathcal{DB}))}{|\mathcal{S}|}$$

Since the body $\frac{P(o \in \varphi(q,\mathcal{DB}))}{|\mathcal{S}|}$ *of the above sum no longer contains the parameter* w*, this body is added once for each iteration of the sum. As the sum is iterated exactly* $|\mathcal{S}|$ *times (once for each element of* $\mathcal{S}$*) we obtain*

$$E(\hat{P}) = |\mathcal{S}| \frac{P(o \in \varphi(q, \mathcal{DB}))}{|\mathcal{S}|}$$

which simplifies to

$$E(\hat{P}) = P(o \in \varphi(q, \mathcal{DB})).$$

Example 3 *Consider the example in Figure 8.5. The probability of object* U_1 *to be the result of a 2NN spatial query, is* 0.1*. This result was derived by explicitly considering all possible worlds, and thus, is not applicable for large databases. Using a Monte-Carlo approach, we draw a number of* $|\mathcal{S}| = 1000$ *sample worlds, getting a total of* 94 *worlds where* U_1 *is in the 2NN result. This yields an estimator of* $\hat{P} = \frac{92}{1000} = 0.092$*, having an approximation error of only* 0.008*, which may be "good enough" in practice.*

8.7.2 Probabilistic Guarantees

Whether a sample size is "good enough" is hard to assess in practice where the true value of $P(o \in \varphi(q, \mathcal{DB})$ is not known. In the above example, there is a non-zero (albeit small) probability that out of 1000 worlds, there exists no world where U_1 is a 2NN result. This would yield an estimator of $\hat{P} = 0$ which is quite far from the true value of $P(P(o \in \varphi(q, \mathcal{DB})) = 0.1$. To ensure that the error is small, probabilistic guarantees are required.

Definition 18 *A probabilistic guarantee ensures that the error of an approximate technique does not exceed some error threshold ϵ with a probability of at least τ, i.e.,*

$$P(|\hat{P} - \mu| \leq \epsilon) \geq \tau,$$

where $\hat{P}$ is an estimator of a parameter μ.

In the case of Monte-Carlo estimation of the probability that an object $o \in \mathcal{DB}$ satisfies some spatial query predicate φ, we can exploit that by Definition 17 the estimator $\hat{P}$ follows a Binomial $B(|\mathcal{S}|, P(o \in \varphi(q, \mathcal{DB})))$ distribution. This observation allows to apply Hoeffding's inequality [80] to obtain the following probabilistic guarantee:

$$P(|\hat{P} - P(o \in \varphi(q, \mathcal{DB}))| < \epsilon) \geq 1 - 2e^{-2\epsilon^2|\mathcal{S}|}, \tag{8.7}$$

where $\hat{P}$ is is defined by Equation 8.4.

Example 4 *Reconsider the example in Figure 8.5 where the true probability $P(A)$ of object A to be the result of a 2NN spatial query, is* 0.1*. Assume that the number of Monte Carlo samples $|\mathcal{S}|$) is* $10,000$*. According to Equation 8.7, the probability that the estimation error of $\hat{P}$ is less that* 0.01 *is at least*

$$P(|\hat{P} - P(P(o \in \varphi(q, \mathcal{DB}))| < 0.01) \geq 1 - 2e^{-2 \cdot 0.01^2 \cdot 10000}$$

$$= 1 - 2 \cdot e^{-2} = 1 - 2 \cdot 0.135 = 0.73.$$

Note that this approximation is not very tight. For example, if $|\mathcal{S}| = 1000$), then Equation 8.7 yields

$$P(|\hat{P} - P(o \in \varphi(q, \mathcal{DB}))| < 0.01) \geq 1 - 2e^{-2 \cdot 0.01^2 \cdot 1000}$$

$$= 1 - 2 \cdot e^{-0.2} = 1 - 2 \cdot 0.819 = -0.637$$

which is a trivial statement.

While the bounds acquired by Equation 8.7 are rather loose, they can be evaluated very efficiently. Most importantly, these bounds allow to avoid evaluating the exact pdf of the binomial distribution $B(|\mathcal{S}|, P(P(o \in \varphi(q, \mathcal{DB}))))$, which requires to evaluate large binomial coefficients.

8.8 Conclusions

Handling uncertainty in data makes use of various techniques and components which have been described in this section and summarized in Table 8.2. Probabilistic data models must be chosen in such a way to best represent the data. Uncertainty models make assumptions on the physical world in order to reduce computational complexity, choosing a model depends on application and permissible loss of information. Based on the choice of model, there are options for analytical solutions to compute exact results, and numerical

Data Model	Discrete
	Continuous
Uncertainty	Existence
	Attribute
	Both
Accuracy	Exact
	Approximate
Answer Semantics	Object based
	Result based

Table 8.2: Components of Uncertainty Models

solutions returning approximate results. Further, probabilistic answer semantics must be specified defining whether result probabilities correspond to result sets, or individual objects. The choice of each individual component has a strong impact on the semantic and the complexity of a probabilistic spatial query.

Subsequently, the main contributions of this thesis in the field of geo-spatial data enriched with uncertainty models are presented:

- Chapter 9 introduces techniques to issue *(k)-Nearest-Neighbor queries on uncertain data using Voronoi decomposition*, where location, shape and extent of Voronoi cells are random variables. To facilitate reliable query processing despite the presence of uncertainty, we employ the concept of *possible-Voronoi cells* and introduce the novel concept of *guaranteed-Voronoi cells*: The possible-Voronoi cell of an object U consists of all points in space that have a non-zero probability of having U as their nearest-neighbor; and the guaranteed-Voronoi cell, which consists of all points in space which must have U as their nearest-neighbor.

- As one application of mining uncertain data, a novel appproach to clustering uncertain data is presented in Chapter 10: a framework based on possible world semantics computes a set of representative clusterings, each of which with probabilistic guarantees not to exceed some maximum distance to the ground truth clustering, i.e., the clustering of the actual (but unknown) data. This framework can employ any existing clustering algorithm. Evaluation shows that in addition to providing quality guarantees, these representative clusterings have a much smaller deviation from ground truth than existing appproaches.

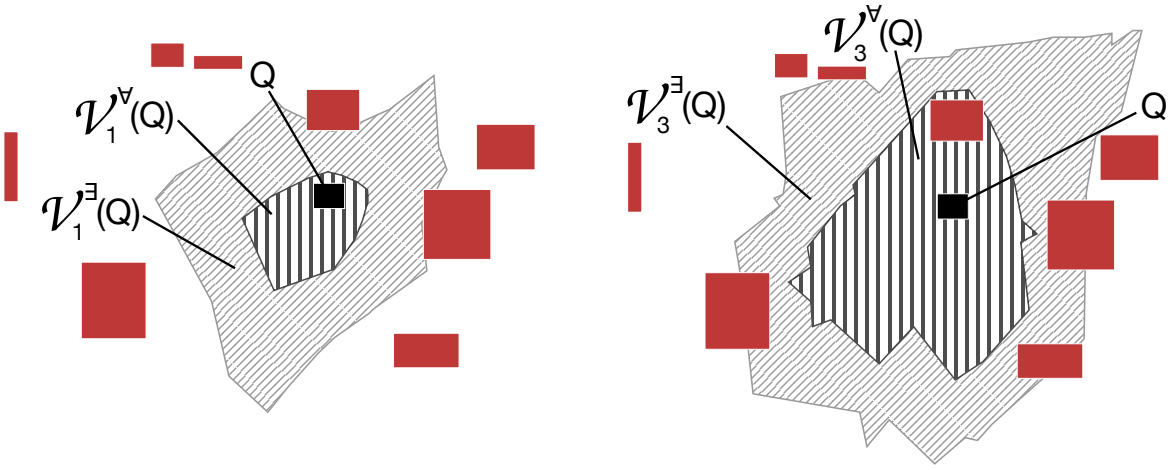
- Chapter 11 gives an outlook on further possibilities building on the techniques described earlier, in particular on how to apply the concept of representative answers from Chapter 10 to queries on uncertain data such as ϵ-range queries, ranking, or k-nearest-neighbor queries.

Chapter 9

Approximate Probabilistic Nearest Neighbor Queries

9.1 Introduction

The extensive use of social media, s.a. smartphones, and social networks produce a huge flood of geo-spatial and geo-spatio-temporal data. This data allows to assess information about the current positions of mobile entities, such as friends in social networks, unoccupied cabs in a taxi application, or the current position of users in augmented reality games. However, our ability to unearth valuable knowledge from large sets of spatial and spatio-temporal data is often impaired by the quality of the data. In this work, we revisit the problem of reliably answering nearest-neighbor queries in uncertain data. The problem of finding the closest uncertain object, which has applications such as taxi-customer matching, has gained much attention in recent years [20, 39, 7, 16]. Following a common approach in uncertain data management, these approaches assume that uncertain objects are represented by rectangular or circular uncertainty regions, which are guaranteed to enclose the true (but unknown) position of the respective spatial objects. We carry the concept of Voronoi cells to uncertain data, following the idea of [206] which is to approximate the *possible Voronoi cell* $\mathcal{V}^{\exists}(O)$ of an object O, which is defined as the space where a query point q can possibly have O as its nearest neighbor, i.e., there exists such a possible world. We leverage this work by defining the notion of the *guaranteed Voronoi cell* $\mathcal{V}^{\forall}(O)$, which we define as the space for which we can guarantee, that any query point q in that space must have the uncertain object O as their nearest neighbor in any possible world. Applications for possible Voronoi cells include geo-location-based services, such as taxi assignments: The possible Voronoi cell (guaranteed Voronoi cell) of an individual taxi cab c covers the space of a city where customers may possibly (must certainly) have c as their nearest taxi. In such an application, as we see in taxi-GPS data sets such as the T-drive dataset [205, 204], the GPS position $c(t)$ of a cab c at a time t may be highly obsolete, due to infrequent GPS updates. Models to infer the uncertainty region of a mobile object on a road network given past observations have been given in the literature [136].



(a) Inner and outer cell for $k = 1$.

(b) Inner and outer cell for $k = 3$.

Figure 9.1: Uncertain Voronoi cells.

Furthermore, our approach is the first one to allow an efficient and effective approximation of a higher-order possible or guaranteed Voronoi cell. The k'th-order possible (guaranteed) Voronoi cell of an uncertain object O is defined as the subset of space where a query object q in that space may possible (must certainly) have O as one of it's k-nearest neighbors. Our concept of k'th-order possible (guaranteed) Voronoi cells, allows to leverage geoinformation systems that require the computation of k-nearest neighbors by allowing to efficiently find guaranteed and possible k-nearest neighbors in the presence of uncertain data. For example, applications to notify your k-nearest-friends based on obsolete and thus uncertain location data, will benefit from our approach by allowing to assess which of your friends are guaranteed to be in the result set, which of them are candidates, and which cannot be your nearest neighbors.

To illustrate our notion of uncertain Voronoi cells, consider Figure 9.1, where rectangles correspond to the uncertainty regions of objects. For a query object Q, the large shaded region in Figure 9.1(a) depicts the subspace $\mathcal{V}_1^{\exists}(Q)$ for which it holds that any point $p \in \mathcal{V}_1^{\exists}(Q)$ may possibly have object Q as its nearest neighbor. We define this region as the *possible Voronoi cell* of Q. In addition, Figure 9.1(a) further shows the region $\mathcal{V}_1^{\forall}(Q)$, containing all the points in space for which we can guarantee that any such point $p \in \mathcal{V}_1^{\forall}(Q)$ must have Q as their nearest-neighbor, regardless of the exact position of Q, and regardless of the positions of all other uncertain objects. The latter region is called the *guaranteed Voronoi cell* of Q.

Additionally, Figure 9.1(b) depicts the uncertain Voronoi cells of order three for the same object Q. More specifically, the large shaded region denoted by $\mathcal{V}_3^{\exists}(Q)$ corresponds to all points in space such that for any such point $p \in \mathcal{V}_3^{\exists}(Q)$ there must no more than two database objects which are certainly closer to p than to Q. In order words, but equivalently, point p may possibly have the uncertain object Q as one of its 3-nearest-neighbors, that is, there exists a possible world of object locations such that Q is a 3-nearest-neighbors of

p. Analogously to these concepts, the region $\mathcal{V}_3^\forall(Q)$ contains all points p such that there must be less than two uncertain database objects closer to p than Q, thus guaranteeing that p has Q as one of its 3-nearest-neighbors.

Finding the regions $\mathcal{V}_k^\exists(Q)$ and $\mathcal{V}_k^\forall(Q)$ for an arbitrary value of k is the goal of this work. This is not a trivial task: The shape of these regions is a non-convex region which is bounded by hyperbolic curves. As explained in [39, 55, 206], an exact construction of $\mathcal{V}(A)$ requires exponential time. For this reason, an approximate technique for deriving the possible Voronoi cell $\mathcal{V}_1^\exists(Q)$ was given in [206]. We propose a new solution for this problem, which extends the existing solution of [206] by the following aspects:

- Unlike previous solutions, our approach offers full index support, indexing the object space using an R^*-tree [14] and indexing the data space using a *kd*-trie [140].
- Rather than approximating the Voronoi cell $\mathcal{V}_1^\exists(Q)$ by a single rectangle ([206]), we use a set of *kd*-trie partitions allowing much higher approximation quality. This gain in approximation quality not only improves query times, as our experiments show, but can also be used to gain a detailed visual exploration of possible Voronoi cells.
- In contrast to previous solutions presented in [58], we extend our solution to additionally compute the guaranteed Voronoi cell $\mathcal{V}_1^\forall(Q)$, a problem which has not been studied in previous literature.
- Another novel extension, we extend our algorithm to find possible Voronoi cells $\mathcal{V}_k^\exists(Q)$ and guaranteed Voronoi cells $\mathcal{V}_k^\forall(Q)$ of higher order, i.e., for values of k greate than one. This problem has not been studied before either.
- Our experiments further show that our provided index support for both data and space enables the scaling of uncertain Voronoi cell computation to large databases.

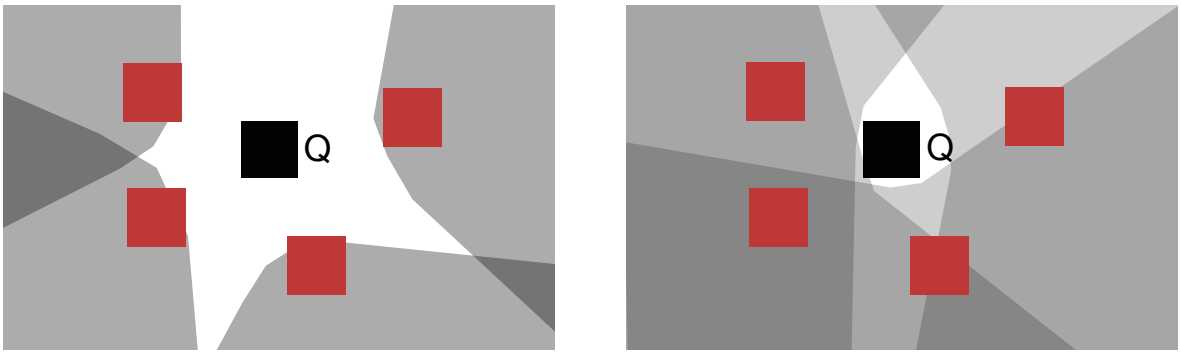
9.2 Related Work

The problem of answering nearest neighbor queries on uncertain data generally involves two steps: A filter approach and a refinement step. In the filter step, a (possibly small) set of objects is returned that contains all objects having a non-zero probability of being the result object. In the refinement step, the exact probability of each candidate object is computed. The refinement step is the main research topic of [37, 122, 17], showing how to compute exact probabilities of an object to be the nearest neighbor of a query object, given the probability density functions of objects. In contrast, other existing work focuses on the filter step, applying spatial filter steps in order to identify object that are guaranteed to have a zero probability to be the result object [16, 206, 29]. In all of these works, (reverse-) nearest neighbor queries on uncertain data are supported by evaluating spatial domination at query time, in order to identify objects that must (not) be part of the query result. Thus, these *lazy approaches*, have in common that the main computation is performed at query time. In this work, we propose an *eager approach*: for each database object U, we

precompute their possible-Voronoi cell (and their guaranteed-Voronoi cell), i.e., the regions in space which may (and must) have U as one of their k-nearest neighbor. Given these regions, we can reduce a kNN query on uncertain data to a simply polygon intersection test, vastly reducing query times.

In this work, we focus on the filter step, i.e., the step of finding objects having a non-zero probability to be the nearest neighbor of an object using Voronoi-cells. The idea of using Voronoi diagrams to answer nearest neighbor (NN) queries over points has been widely studied [11] . In this context, Voronoi diagrams have been used to support nearest neighbor queries in geo-spatial applications [165], location-based services [207, 139], in spatial data streams [164] and in distributed spatial environments [6] as well as in spatial network environments [109]. To support nearest neighbor queries on uncertain data, initial approaches have been presented in [37, 20]. However, in these work, only the database objects are assumed to be uncertain, whereas the query object is assumed to be a point. In [39] a solution to compute possible Voronoi-cells for the case of circular uncertainty regions has been presented. This exact approach has exponential construction and storage cost. Due to this computational drawback, an approximate solution was presented in [206] which approximates the true (but unknown) possible Voronoi-cell $\mathcal{V}(O)$ of an uncertain object O using two rectangles: A single conservative rectangle $h(O)$ which is guaranteed to completely contain $\mathcal{V}(O)$, and a single progressive rectangle $l(O)$ which is guaranteed to be completely contained by $\mathcal{V}(O)$. These two approximation rectangles are obtained by iteratively expanding the progressive rectangle $l(O)$, and iteratively shrinking the conservative rectangle $h(O)$. However, considering examples such as shown in Figure 9.1, it is evident that such approximations may be rather inaccurate. Thus, $h(O)$ may cover a large body of space not belonging to $\mathcal{V}(O)$, while $l(O)$ may miss a large body of $\mathcal{V}(O)$, even in the case where $h(O)$ is the smallest conservative bounding rectangle and $l(O)$ is the largest progressive bounded rectangle.[1] Furthermore, an approach for nearest neighbor search on moving uncertain objects has been presented in [7]. A problem common to [39] and [7] is that their solutions are customized for 2D data, making extensive use of intersection and rotation operations of 2D hyperbolic curves. Our approach, as well as the approach of [206] is applicable to arbitrary dimensionality. In comparison to [206], the main contribution of this work is that we can accurately approximate an arbitrarily shaped possible Voronoi-cell, rather than using a single rectangular approximation only. This allows to answer nearest-neighbor queries more efficiently, since less candidates have to be checked, and it allows to more precisely illustrate the Voronoi-region of an uncertain object. Finally, a first solution to compute the first-order possible Voronoi cell $\mathcal{V}_1^{\exists}(U)$ has been presented in [58]. This work extends the solution of [58] to the concept of guaranteed-Voronoi cells $\mathcal{V}_1^{\forall}(U)$. Further, this work extends both concepts $\mathcal{V}_1^{\exists}(U)$ and $\mathcal{V}_1^{\forall}(U)$ to higher-order Voronoi cells $\mathcal{V}_k^{\exists}(U)$ and $\mathcal{V}_k^{\forall}(U)$, thus allowing to employ these concepts for $k > 1$-nearest neighbor queries. These novel concepts and notions will be formally defined in the following section.

[1] The later case can not be guaranteed by the approach of [206] due to the numeric nature of their approach.



(a) $V_1^{\exists}$ Pruning Regions (b) $V_1^{\forall}$ Pruning Regions

Figure 9.2: Spatially dominated regions of objects surrounding Q.

9.3 Problem Definition

Figure 9.2(a) shows how the possible Voronoi cell $\mathcal{V}_1^{\exists}(Q)$ of an uncertain object Q is defined. For each of the anonymous database objects of Figure 9.2, each shaded region corresponds to the pruning region $S_O(Q)$, i.e., the smallest region such that for any $p \in S_O(Q)$, the corresponding object O must be closer to p than Q. Formally,

Definition 19 (Nearest Neighbor Pruning Region) *Let $\mathcal{DB} = \{O_1, ..., O_N\}$ be an uncertain database where each object $O_i \in \mathcal{DB}$ is represented by a rectangular uncertainty region in $\mathcal{R}^d$. Let dist(.,.) denote any L_p norm.*[2] *For any $A, B \in \mathcal{DB}$, we define the nearest neighbor pruning region where any point must be closer to A than to B as follows:*

$$S_A(B) := \{q \in \mathcal{R}^d : maxDist(q, A) < minDist(q, B)\},$$

where $maxDist(q, A)$ and $minDist(q, B)$ denote the maximum and minimum distance between a point q and a rectangle A or B, respectively, as defined in [152].

Figure 9.2(a) shows four nearest neighbor pruning regions $S_{O_1}(Q), ..., S_{O_4}(Q)$ as shaded regions. Any region implies, the the respective object O_1 must be closer to this region than Q. Using Definition 19, we can now define the possible Voronoi cell $\mathcal{V}_1^{\exists}(Q)$ of an object Q as the space that does not intersect any nearest neighbor pruning region $S_O(Q)$ of a database object O. Formally:

Definition 20 (Possible Voronoi Cell) *Let $Q \in \mathcal{DB}$ be an uncertain object. Then the possible Voronoi cell $\mathcal{V}_1^{\exists}(Q)$ is defined as*

$$\mathcal{V}_1^{\exists}(Q) = \mathcal{R}^d \setminus \bigcup_{O \in \mathcal{DB} \setminus \{Q\}} S_O(Q).$$

[2] We use Euclidean distance in all examples and illustrations, but any L_p norm can be applied.

Notation	Meaning	Notation	Meaning
$\mathcal{DB}$	Uncertain Object Database	$O, Q, U \in \mathcal{DB}$	uncertain objects
$\mathcal{I}_{\mathcal{DB}}$	Hierarchical Data Index	$\mathcal{I}_{\mathcal{S}}$	Hierarchical Space Index
$\mathcal{G}$	d-dimensional grid	$g_i \in \mathcal{G}$	Rectangular Grid Cell
$\mathcal{V}_k^{\exists}(U)$	k-th order possible Voronoi cell of U		
$\mathcal{V}_k^{\forall}(U)$	k-th order guaranteed Voronoi cell of U		
$S_A(B) \subseteq \mathcal{R}^d$	The region where object A dominates object B		
$Dom(A, B, R)$	Predicate that is true iff rectangle R is fully contained $S_A(B)$. Can be evaluated efficiently [56].		
$PDom(A, B, R)$	Predicate that is true iff rectangle R intersects $S_A(B)$. Can be evaluated efficiently [56].		
$h \subseteq \mathcal{R}^d$	Rectangular Space Index Entry obtained from $\mathcal{I}_{\mathcal{S}}$: Partition of Space for which we want to decide if it belongs to $\mathcal{V}(U)$		
$e \subseteq \mathcal{R}^d$	Rectangular Data Index Entry obtained from $\mathcal{I}_{\mathcal{DB}}$: Spatial approximation of a set of data objects if e is non-leaf entry, Uncertainty region of a single data object if e is a leaf entry.		

Table 9.1: Table of Notations.

In Figure 9.2, the white (i.e., non-shaded) region corresponds to the Voronoi cell $\mathcal{V}_1^{\exists}(Q)$.

Figure 9.2(b) depicts shades regions corresponding to the four nearest neighbor pruning regions $S_Q(O_1), ..., S_Q(O_4)$ as shaded regions. Again following Definition 19, any such region $S_Q(O)$ corresponds to the region that is guaranteed to be closer to Q than to O. This observation allows us to define the guaranteed Voronoi cell $\mathcal{V}_1^{\forall}(Q)$ as the space intersection all of these regions.

Definition 21 (Guaranteed Voronoi Cell) *Let $Q \in \mathcal{DB}$ be an uncertain object. Then the guaranteed Voronoi cell $\mathcal{V}_1^{\forall}(Q)$ is defined as*

$$\mathcal{V}_1^{\forall}(Q) = \bigcap_{O \in \mathcal{DB} \setminus \{Q\}} S_Q(O).$$

The challenge of this work is to accurately approximate both uncertain Voronoi regions $\mathcal{V}_1^{\forall}(Q)$ and $\mathcal{V}_1^{\exists}(Q)$ efficiently.

9.4 Spatial Domination Revisited

The concept of spatial domination and efficient techniques to verify it were introduced in [56]. Spatial domination describes the spatial relation of three rectangles to each other. Since the spatial domination can also be utilized for the computation of uncertain voronoi cells, we briefly want to review the concept. Notations used throughout this chapter are explained in Table 9.1.

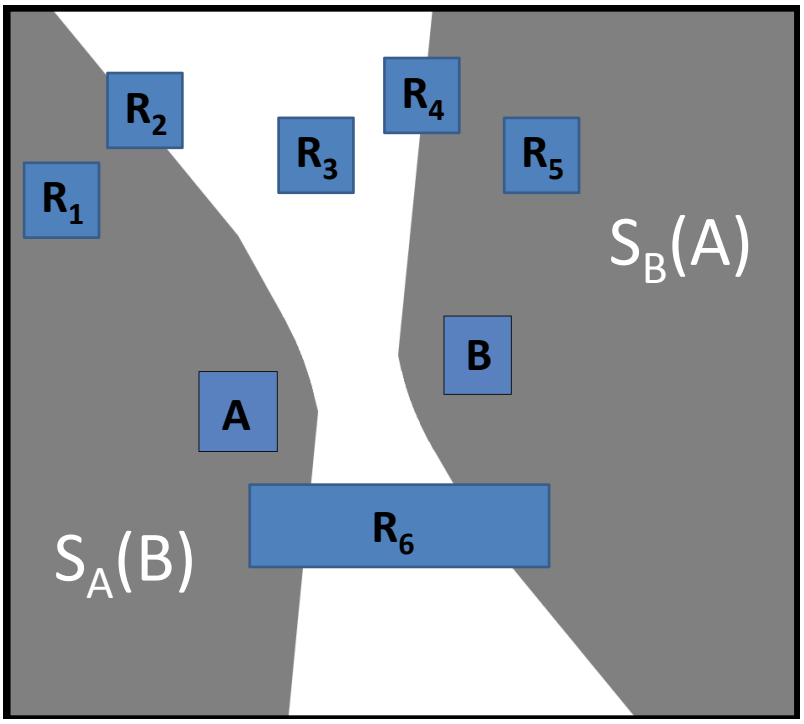


Figure 9.3: Domination relation

Definition 22 (Spatial Domination) *Let $A, B, R \subseteq \mathcal{R}^d$ be rectangles in a d-dimensional space and dist() be a distance function defined on that space. The rectangle A dominates B w.r.t. R iff for all points $r \in R$ it holds that every point $a \in A$ is closer to r than any point $b \in B$, i.e.*

$$Dom(A, B, R) \Leftrightarrow \forall r \in R, \forall a \in A, \forall b \in B : dist(a, r) < dist(b, r)$$

Informally speaking, $Dom(A, B, R)$ is thus true if A is "certainly" closer to R than B. In addition the concept of partial spatial domination was introduced.

Definition 23 (Partial Spatial Domination) *Let $A, B, R \subseteq \mathcal{R}^d$ be rectangles in a d-dimensional space and dist() be a distance function defined on that space. The rectangle A dominates B partially w.r.t. R , denoted by PDom(A, B, R) if A dominates B for some, but not all $r \in R$, i.e.*

$$\begin{aligned} PDom(A, B, R) \Leftrightarrow (\exists r \in R : \forall a \in A, \forall b \in B : dist(a, r) < dist(b, r)) \wedge \\ (\exists r \in R : (\exists a \in A, \exists b \in B : dist(a, r) \leq dist(b, r)) \wedge \\ (\exists a \in A, \exists b \in B : dist(a, r) \geq dist(b, r))). \end{aligned}$$

In [16] it was shown that spatial domination can be utilized when the rectangles conservatively approximate uncertain objects. In this case Dom(A, B, R) means P("R is closer to A than to B") = 1 and PDom(A, B, R) means $0 \leq$ P("R is closer to A than to B") ≤ 1.

Using the Dom()- and the PDom()-function it is thus possible to decide the location of a rectangle w.r.t. the uncertain bisector of two uncertain objects. The uncertain bisector between two uncertain objects A and B (conservatively approximated by rectangles) defines three spaces: In $S_A(B) = \{s \in \mathcal{S} : Dom(A, B, \{s\})\}$ all objects are certainly closer to A than to B, in $S_B(A) = \{s \in \mathcal{S} : Dom(B, A, \{s\})\}$ object are certainly closer to B than to A and in the space in between no certain decision can be made. This relation is shown in Figure 9.3. We are thus able to decide where the rectangle R is located w.r.t. the bisector $S_B(A)$ and $S_A(B)$ of A and B respectively by performing the Dom() and the PDom() function [56]. The following six cases are defined using a function DomCase(A, B, R) as follows.

Definition 24 (Domination Cases) *Let A and B be rectangles. For any rectangle R, one of the following cases holds:*

Case 1: *R is fully contained in $S_A(B)$ iff $Dom(A, B, R)$;*

Case 2: *R intersects $S_A(B)$ but not $S_B(A)$ iff $PDom(A, B, R) \wedge \neg PDom(B, A, R)$;*

Case 3: *R intersects neither $S_A(B)$ nor $S_B(A)$ iff*

$\neg Dom(A, B, R) \wedge \neg PDom(A, B, R) \wedge \neg PDom(B, A, R) \neg Dom(B, A, R)$;

Case 4: *R intersects $S(B)$ but not $S(A)$ iff $\neg PDom(A, B, R) \wedge PDom(B, A, R)$;*

Case 5: *R is fully contained in $S(B)$ iff $Dom(B, A, R)$;*

Case 6: *R intersects both $S(A)$ and $S(B)$ iff $PDom(A, B, R) \wedge PDom(B, A, R)$;*

Figure 9.3 depicts all possible cases. Here, each rectangle R_i corresponds to Case i in Definition 24. Note that the materialization of the pruning regions $S_A(B)$ and $S_B(A)$ is a hard problem [206]. Nevertheless, the function DomCase(A, B, R) allows to efficiently decide between the six possible domination cases defined above. In the next section we will show how to use these relations in order to compute uncertain Voronoi cells.

9.5 Possible and Guaranteed Voronoi Cell Approximation

9.5.1 Naive solution

Computing uncertain Voronoi cells is a daunting task for two reasons: First, it is challenging to find the objects in the database that have an effect on its shape. Second, the representation of the cell is hard since it consists of many linear and parabolic parts that grow exponentially with the dimensionality. This section will present four algorithms that apply the concept of spatial domination to efficiently approximate the possible-Voronoi cell $\mathcal{V}_k^{\exists}(U)$ and the guaranteed-Voronoi cell $\mathcal{V}_k^{\forall}(U)$ of an object U as tight as possible. The first, naive, algorithm divides the space into equi-distant grid cells and labels the cells according to their membership to the possible-Voronoi cell. The second algorithm, additionally uses

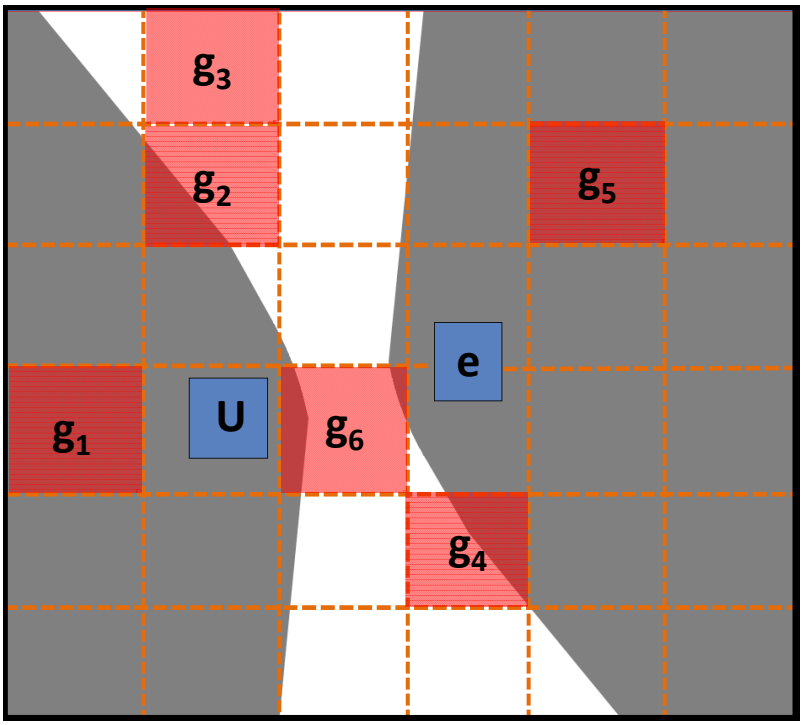


Figure 9.4: Cases of domination of a grid cell

an R*-tree to index the data objects to avoid exploration of irrelevant objects. The third algorithm uses a kd-trie to index the grid cells, in order to identify large regions of space for which their relation to $\mathcal{V}_k^{\exists}(U)$ and $\mathcal{V}_k^{\forall}(U)$ can be decided without exploring smaller subregions. The fourth algorithm uses both a kd-trie to index the space and an R-tree to index the data. For the later algorithm, the main challenge is to smartly descend both hierarchical index structures in parallel, to minimize the computational overhead.

A straightforward way of computing $\mathcal{V}_k^{\exists}(U)$ and $\mathcal{V}_k^{\forall}(U)$ is to apply an equi-distant d-dimensional grid to partition the data space. For each cell g we decide weather it belongs to $\mathcal{V}_k^{\exists}(U)$, to $\mathcal{V}_k^{\forall}(U)$, or to neither.

Algorithm

The algorithm takes as input the target object U, an uncertain database $\mathcal{DB}$ and a grid $\mathcal{G}$ covering the space of $\mathcal{DB}$. We iterate over all grid cells $g \in \mathcal{G}$ and in order to decide whether g is part of the UV cell of U. Therefore, domination against all objects $O \in \mathcal{DB} \setminus U$ has to be checked. All possible cases of domination of a grid-cell g are depicted in Figure 9.4. We can decide the following cases for each grid cell g: (i) g is completely outside of $\mathcal{V}_k^{\exists}(U)$ or (ii) g is a boarder cell intersecting the edge of $\mathcal{V}_k^{\exists}(U)$, or (iii) g is completely inside $\mathcal{V}_k^{\exists}(U)$ but completely outside $\mathcal{V}_k^{\forall}(U)$, or (iv) g is a boarder cell touching the edge of $\mathcal{V}_k^{\forall}(U)$, or (v) g is completely inside $\mathcal{V}_k^{\forall}(U)$. To make this decision, we can exploit the six cases of Definition 24. In the following, the operator $\exists_k O \in DB : \varphi$ denotes that the

predicate φ holds for at least k objects in $\mathcal{DB}$. Formally, our first algorithm proceeds as follows:

i) If $\exists_k O \in \mathcal{DB} \setminus U : Dom(O, U, g)$ then g is neither part of $\mathcal{V}_k^{\exists}(U)$ nor $\mathcal{V}_k^{\forall}(U)$, since at least k database object dominate g. This domination corresponds to **Case 5** of Definition 24 and cell g_5 in Figure 9.4.

ii) Otherwise, if $\exists_k O \in \mathcal{DB} : PDom(O, U, g)$ then at least a small part of g must be part of $\mathcal{V}_k^{\exists}(U)$. This case corresponds to the cases of cells g_4 and g_6 in Figure 9.4, i.e., **Case 4** or **Case 6** of Definition 24.

iii) Otherwise we can conclude that g is completely contained in $\mathcal{V}_k^{\exists}(U)$. Since for all but at most $k-1$ database objects, it holds that g corresponds to one of the remaining cases **Case 1**, **Case 2** and **Case 3** of cells g_1, g_2 or g_3, respectively, as shown in Figure 9.4. If $\exists_k O \in \mathcal{DB} : \neg PDOM(U, O, g)$, i.e., if there exist k objects that are not dominated by g, then g is completely outside of $\mathcal{V}_k^{\exists}(U)$. The case $PDOM(U, O, g)$ corresponds to **Case 1**, **Case 2** and **Case 6** of Definition 24 and cell g_5 in Figure 9.4.

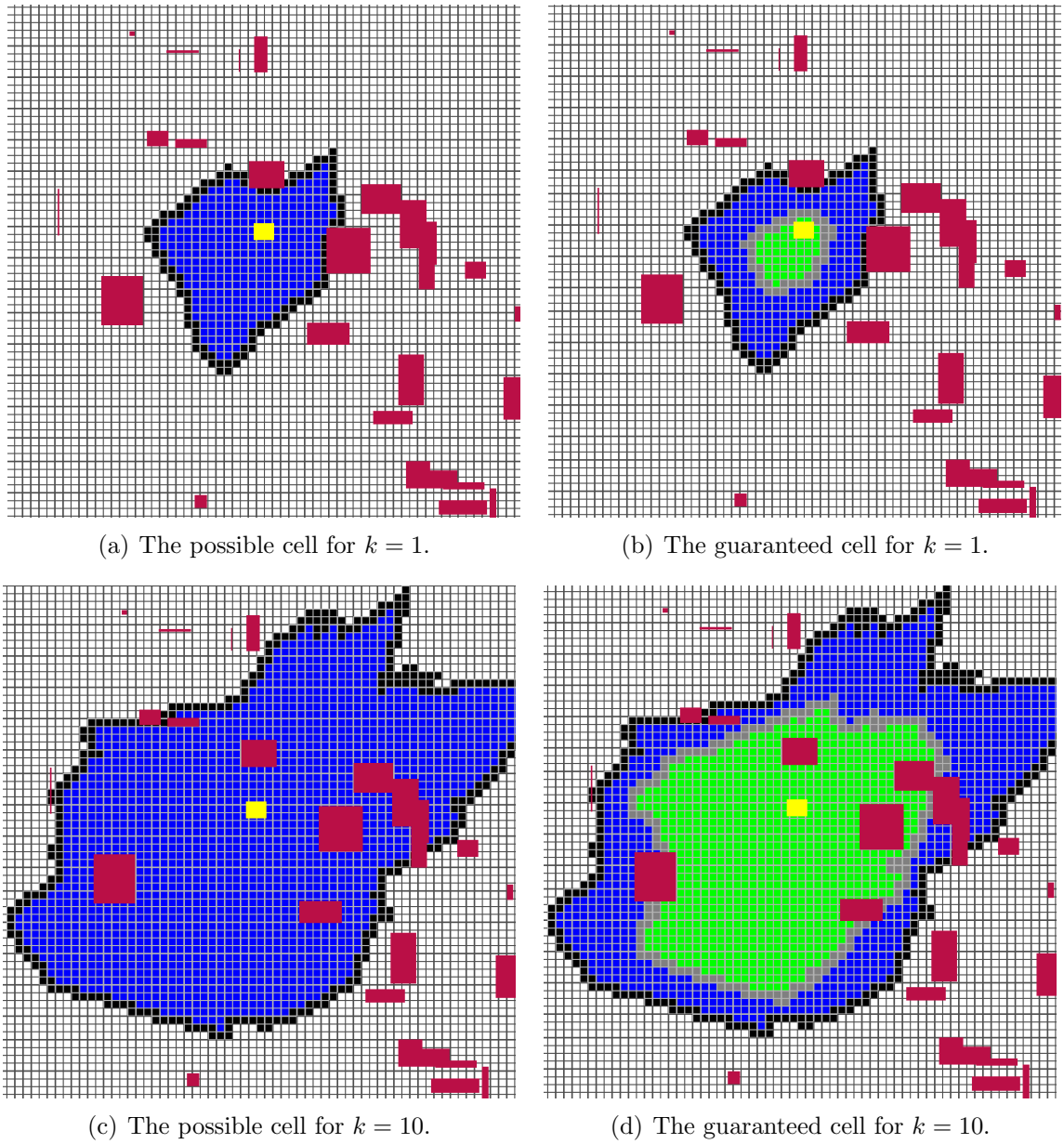
iv) Otherwise, we can guarantee that g intersects $\mathcal{V}_k^{\forall}(U)$. If $\exists_k O \in \mathcal{DB} : \neg DOM(U, O, g)$, then k objects may have a chance to dominate g, such that g cannot be fully contained in $\mathcal{V}_k^{\forall}(U)$. The case $DOM(U, O, g)$ corresponds solely to **Case 1** in Definition 24.

v) Otherwise, we can conclude that $DOM(U, O, g)$ holds for all but $k-1$ database objects, such that any point in g is guaranteed to have U as its k-nearest neighbor. Thus g must be completely contained in $\mathcal{V}_k^{\forall}(U)$.

The set of all grid cells satisfying v) define a lower bound of $\mathcal{V}_k^{\forall}(U)$, and the grid cells satisfying iv) or v) define an upper bound of $\mathcal{V}_k^{\forall}(U)$. Analogously, all grids cells satisfying iii-v) define a lower bound of $\mathcal{V}_k^{\exists}(U)$ and all grid cells satisfying ii-v) define an upper bound of $\mathcal{V}_k^{\exists}(U)$.

For a small database of uncertain objects an exemplary result of this approach is depicted in Figure 9.5. The object U for which the possible and guaranteed Voronoi cells are computed is highlighted in yellow. Furthermore, a space grid is shown, where (i) unfilled space cells are guaranteed to be outside of $\mathcal{V}_k^{\exists}(U)$ (and thus outside of $\mathcal{V}_k^{\forall}(U)$), (ii) black cells are guaranteed to be on the border of $\mathcal{V}_k^{\exists}(U)$, (iii) blue cells are guaranteed to be inside $\mathcal{V}_k^{\exists}(U)$, but outside of $\mathcal{V}_k^{\forall}(U)$, (iv) grey cells are on the boarder of $\mathcal{V}_k^{\forall}(U)$, and green cells are located inside $\mathcal{V}_k^{\forall}(U)$. In particular, Figure 9.5(a) shows only the possible Voronoi cell $\mathcal{V}_1^{\exists}(U)$, for the computation of which previous solutions have been presented [206, 58] in the literature. In addition, Figure 9.5 illustrates the new concepts presents in this work:

- the novel concept of a guaranteed Voronoi-cell $\mathcal{V}_1^{\forall}(U)$, which is additionally shown in Figure 9.5(b), denoting the space for which any point is guaranteed to have uncertain object U as its nearest neighbor,



(a) The possible cell for $k = 1$.

(b) The guaranteed cell for $k = 1$.

(c) The possible cell for $k = 10$.

(d) The guaranteed cell for $k = 10$.

Figure 9.5: Uncertain Voronoi cells.

- the novel concept of a higher order possible-Voronoi cell $\mathcal{V}^{\exists}_{k>1}(U)$, depicted in Figure 9.5(c). Here, the 10th order possible-Voronoi cell $\mathcal{V}^{\exists}_{10}(U)$ is shown. Thus, any space cell highlighted in blue in Figure 9.5(c) has a non-zero probability to have the yellow object U as one of its ten nearest neighbors. Note that, for illustration purpose, only a small part of the database is shownin Figure 9.5(c) and some of the objects responsible for the shape of $\mathcal{V}^{\exists}_{10}(U)$ are not shown,

- and the combination of both new concepts, which creates the notion of a higher-order guaranteed Voronoi cell $\mathcal{V}^{\forall}_{k>1}(U)$, which is illustrated in Figure 9.5(d).

The naive solution presented in this section allows to compute all of the above solutions, i.e., the possible Voronoi cell $\mathcal{V}_k^{\exists}$ and the guaranteed Voronoi cell $\mathcal{V}_k^{\forall}$. However, this solution requires to individually consider each space cell of a space grid, paired with each database object, to compute the respective domination cases. Clearly, the resulting complexity, linear in the number of space cells and linear in the number of database objects, is inapplicable to large data sets. The next sections improve this computational complexity by employing index structures for both the object space and the data objects. In Section 9.6, we employ an aggregate R-tree to index uncertain objects. In particular, we show how we can use a higher level R-tree entry e for the computation of $\mathcal{V}_k^{\exists}$ and $\mathcal{V}_k^{\forall}$ without having to refine e. Then, in Section 9.7, we proceed to employ a quad-tree to index the space grid. We show how we can decide, for non-leaf entry h of this quad-tree, how to decide if h must (not) part of $\mathcal{V}_k^{\exists}$ and $\mathcal{V}_k^{\forall}$. Finally, our main contribution is given in Section 9.8, where we propose an algorithm to compute Voronoi-cells $\mathcal{V}_k^{\exists}$ and $\mathcal{V}_k^{\forall}$ using both the aforementioned index structures. The main challenge solved in this section is to find heuristics to descent both index structures, for space and data, in parallel, in order to minimize the number of domination checks incurred by the algorithm.

9.6 Indexing $\mathcal{DB}$

Obviously, checking an object U against all uncertain objects $O \in \mathcal{DB}$ is very expensive. Instead, we can use an MBR based index structure $\mathcal{I}_{\mathcal{DB}}$ (such as an R*-Tree) to organize the objects hierarchically.

Algorithm

The algorithm takes as input the target object U, $\mathcal{I}_{\mathcal{DB}}$ and a grid covering the space of $\mathcal{I}_{\mathcal{DB}}$. For each grid cell g the algorithm traverses the entries e of $\mathcal{I}_{\mathcal{DB}}$ in a best first manner [78] according to $MinDist(e, U)$. Note that the entry e can be a single uncertain object (i.e., a leaf-entry) or an intermediate node that conservatively approximates multiple uncertain objects. Since we assume that our data index uses rectangular approximations, we can then apply Definition 24 to decide which index entries have to be accessed. For reference, the following cases are shown in Figure 9.4. Keep in mind that in this case, the entries e are data index entries, which may be intermediate entries representing multiple data objects. Intuitively, by refining a data entry e into its children entries, the uncertainty of the domination regions of these children become smaller. Therefore, the dominated regions $S_U(e)$ and $S_e(U)$ (c.f. Figure 9.3 and Figure 9.4) grow larger, and the white region for which no decision can be made grows larger. Consequently, refining an entry e may the case of some cells to change. Recall that for each cell, we need to decide for one of the five following cases: (i) definitely outside of $\mathcal{V}_k^{\exists}(U)$, (ii) on the border of $\mathcal{V}_k^{\exists}(U)$, (iii) inside $\mathcal{V}_k^{\exists}(U)$ but outside of $\mathcal{V}_k^{\forall}(U)$, (iv) on the border of $\mathcal{V}_k^{\forall}(U)$ or (v) completely inside $\mathcal{V}_k^{\forall}(U)$. As an example, these cases are color-coded by the colors white, black, blue, grey and green, respectively in Figure 9.6.

Case 1: $Dom(U, e, g_1)$: e and none of its children can exclude g_1 from either UV-cell $\mathcal{V}_k^{\exists}(U)$ nor $\mathcal{V}_k^{\forall}(U)$. Thus refinement of e cannot yield any new information.
Case 2: $PDom(U, e, g_2)$: If e is not a leaf entry, then for some (or even all) of the child nodes e' of e the predicate $Dom(U, e', g_1)$ might hold, thus possibly reducing **Case 2** to **Case 1** for some child nodes of e.

Case 3: $\neg PDom(U, e, g_3) \wedge \neg PDom(e, U, g_3)$: As long as e is not a leaf entry (an object), there might exist a child of e which excludes g_3 from $\mathcal{V}_k^{\forall}(U)$ (**Case 5**), or for which g_3 is guaranteed to be closer to U (**Case 2**).

Case 4: $PDom(e, U, g_4)$: In this case, if e is not a leaf entry, then for some (or even all) of the child nodes e' of e the predicate $Dom(e', U, g_1)$ might hold, thus possibly reducing **Case 4** to **Case 5** for some child nodes of e. .

Case 5: $Dom(e, U, g_5)$: U is dominated by e and thus by all the children of e. Entry e will no longer be refined, and the number of **Case 5** objects is increased by the number of children of e.

Case 6: $PDom(U, e, g_6) \wedge PDom(e, U, g_6)$: In this case, parts of g are dominated by e, for some parts e dominates g, and for some parts no decision can be made. In this case, entry e does no longer need to be refined, any children of e must yield the same case.

Clearly, using the data index $\mathcal{I}_{\mathcal{DB}}$ allows to avoid a large fraction of dominance checks: for a large page regions e that is located far distance from object U, we can already decide on a high directory level that e and all its children can not influence the shape of $\mathcal{V}_k^{\exists}(U)$ and $\mathcal{V}_k^{\forall}(U)$. To illustrate the effect of employing the data index $\mathcal{I}_{\mathcal{DB}}$, a small sample database is given in Figure 9.6. In addition to the Voronoi-cells $\mathcal{V}_1^{\exists}(U)$ and $\mathcal{V}_1^{\forall}(U)$, Figure 9.6 also shows to what degree the employed data index $\mathcal{I}_{\mathcal{DB}}$ has been refined. Solid rectangle correspond to fully refined uncertain objects. Unfilled rectangles correspond to intermediate R-tree index entries. Clearly, the number of fully refined uncertain objects decreases drastically, thus decreasing the overall number of domination checks that need to be computed, and thus decreasing the overall run-time. Our experimental evaluation in Section 9.9 will give a quantitative analysis of the run-time improvement gained by employing the data index $\mathcal{I}_{\mathcal{DB}}$.

Still, the problem remains that each space cell has to be checked individually. Intuitively, it should be possibly to identify for large regions located far away from U, that these cannot be part of $\mathcal{V}_k^{\exists}(U)$ and $\mathcal{V}_k^{\forall}(U)$. Furthermore, it might be possible to decide for large regions close to (and possibly overlap with) U, that they must be part of $\mathcal{V}_k^{\forall}(U)$. And finally, there might be large regions for which we might be able to decide that they must be part of $\mathcal{V}_1^{\exists}(U)$ but not of $\mathcal{V}_1^{\forall}(U)$. In the next section, Section 9.7 we introduce a space index, which hierarchically structures space cells, allowing us to decide whether a large space region is completely outside of $\mathcal{V}_k^{\exists}(U)$, completely inside $\mathcal{V}_k^{\exists}(U)$ but completely outside $\mathcal{V}_k^{\forall}(U)$, or completely inside $\mathcal{V}_k^{\forall}(U)$. Finally, Section 9.8 will combine both index structures for data and for space, by showing heuristics to traverse both index structures in an efficient way.

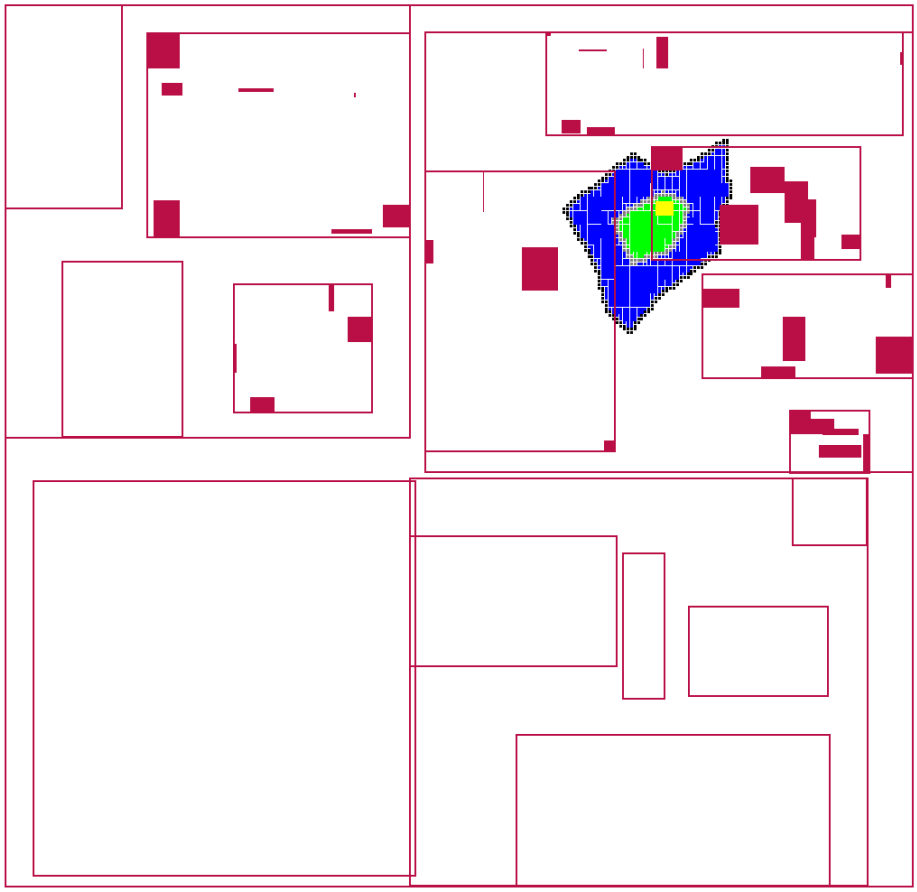


Figure 9.6: Refined page regions of $\mathcal{I}_{\mathcal{DB}}$ for a small example database.

9.7 Indexing $\mathcal{S}$

Instead of indexing the data objects, we show how to index the space grid cells in this section. We propose to use a tree based index structure (denoted as $\mathcal{I}_{\mathcal{S}}$ to organize the data space (e.g. Quadtree, kd-trie). For each entry $h \in \mathcal{I}_{\mathcal{S}}$ it can be checked if it is part of the UV cell of U.

Algorithm

The algorithm takes as input the target object U, $\mathcal{I}_{\mathcal{S}}$, *maxdepth* and a list of all data objects $O \in \mathcal{DB}$. The entries $h \in \mathcal{I}_{\mathcal{S}}$ are traversed in a depth-first manner. For each entry h we check all $O \in \mathcal{DB}$ to decide if the traversal has to go deeper (to the children of h) or its subtree can be discarded for further processing. The parameter *maxdepth* defines the maximum depth that $\mathcal{I}_{\mathcal{S}}$ is traversed. Thus the larger *maxdepth*, the finer the granularity of the UV-cell approximation.

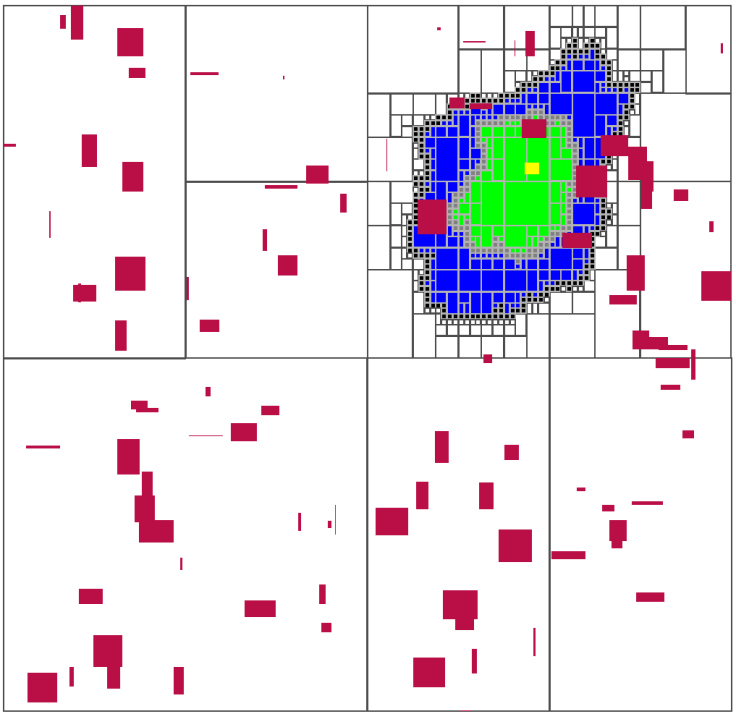


Figure 9.7: Refined page regions of $\mathcal{I}_\mathcal{S}$ for $\mathcal{V}_1^\exists(U)$ and $\mathcal{V}_1^\forall(U)$.

We can again distinguish the same cases as in Section 9.5.1:

i) If $\exists_k O \in \mathcal{DB} \setminus U : Dom(O, U, g)$ then h is neither part of $\mathcal{V}_k^\exists(U)$ nor $\mathcal{V}_k^\forall(U)$, since at least k database object dominate h. Thus h no longer needs to be refined (and can be colored in white in Figure 9.7).

ii) Otherwise, if $\exists_k O \in \mathcal{DB} : PDom(O, U, g)$ then at least a small part of h must be part of $\mathcal{V}_k^\exists(U)$. If h is a directory entry (i.e., an entry having a depth less than *maxdepth*), then h is refined. Otherwise, h is a leaf entry (i.e., a space entry of depth *maxdepth* and is is colored black in Figure 9.7).

iii) Otherwise, if $\exists_k O \neg DOM(U, O, g)$, then there exists at least k objects that might dominate O with respect to h, such that h cannot be part of $\mathcal{V}_k^\forall(U)$. In this case, h can be colored in blue regardless of whether h is a directory of leaf node.

iv) Otherwise, if $\exists_k O \in \mathcal{DB} : \neg DOM(U, O, g)$, then k objects may have a chance to dominate h, such that h cannot be fully contained in $\mathcal{V}_k^\forall(U)$. If h is a directory entry, then h is refined. If h is a leaf entry, then h is colored in grey in Figure 9.7.

v) Otherwise, we can conclude that $DOM(U, O, g)$ holds for all but $k - 1$ database objects, such that any point in h is guaranteed to have U as its k-nearest neighbor. Thus h must be completely contained in $\mathcal{V}_k^\forall(U)$, regardless of whether h is a directory of leaf entry, so h is colored green in Figure 9.7.

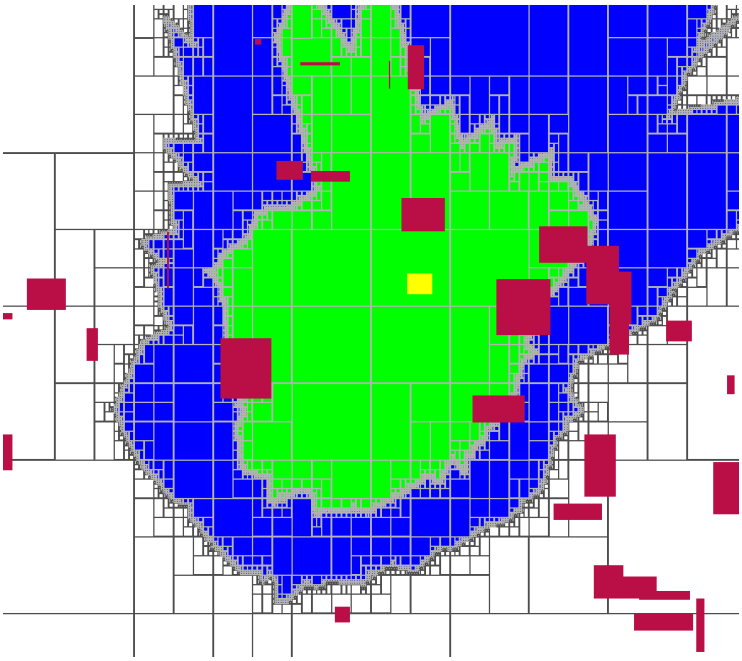


Figure 9.8: Close view on the refined page regions of $\mathcal{I}_{\mathcal{S}}$ for $\mathcal{V}_{10}^{\exists}(U)$ and $\mathcal{V}_{10}^{\forall}(U)$.

Again, to show the effect of employing the space index $\mathcal{I}_{\mathcal{S}}$, a small sample database is given in Figure 9.7 where $\mathcal{V}_1^{\exists}(U)$ and $\mathcal{V}_1^{\forall}(U)$ are computed. Here, no data index $\mathcal{I}_{\mathcal{DB}}$ is used, thus all uncertain objects are shown as solid rectangles. It can be observed in Figure 9.7 that, using the algorithm above, very large space entries of $\mathcal{I}_{\mathcal{S}}$ can be identified as being outside of $\mathcal{V}_5^{\exists}(U)$, illustrated by large white rectangles. In this example, the whole quadrant of the south-east database can be pruned. Are more detailed view on $\mathcal{V}_{10}^{\exists}(U)$ and $\mathcal{V}_{10}^{\forall}(U)$ is given in Figure 9.8 for $k = 10$. Some larger entries of $\mathcal{I}_{\mathcal{S}}$ can be identified as completely contained in $\mathcal{V}_{10}^{\forall}(U)$, represented by the green rectangles of varying size. Finally, some space entries of $\mathcal{I}_{\mathcal{S}}$ are guaranteed to be outside of $\mathcal{V}_{10}^{\forall}(U)$ but inside of $\mathcal{V}_5^{\exists}(U)$, which correspond to the blue rectangles.

Yet, by employing exclusively the space index $\mathcal{I}_{\mathcal{S}}$ without using the data index $\mathcal{I}_{\mathcal{DB}}$, dominance checks are required for every single uncertain database object. In the next section, we present our approach towards employing both index structures $\mathcal{I}_{\mathcal{DB}}$ (indexing the uncertain objects) and $\mathcal{I}_{\mathcal{S}}$ (indexing space) at the same time. For this purpose, we need to find heuristics for a good trade-off between refining data entries of $\mathcal{I}_{\mathcal{DB}}$ and refining space entries of $\mathcal{I}_{\mathcal{S}}$. Especially, we have to expect such hybrid algorithm to refine more data entries than the algorithm shown in Section 9.6, and to refine more space entries than the algorithm presented in this section. The reason is that the previous algorithms, which use only one index, assume that all the respective other data type is completely refined, thus all information is available. When both index structures are traversed however, then, at any time, only partial knowledge is known about both the data index and space index. Thus, it is possible that a space entry is refined which, while at a later stage of the algorithm,

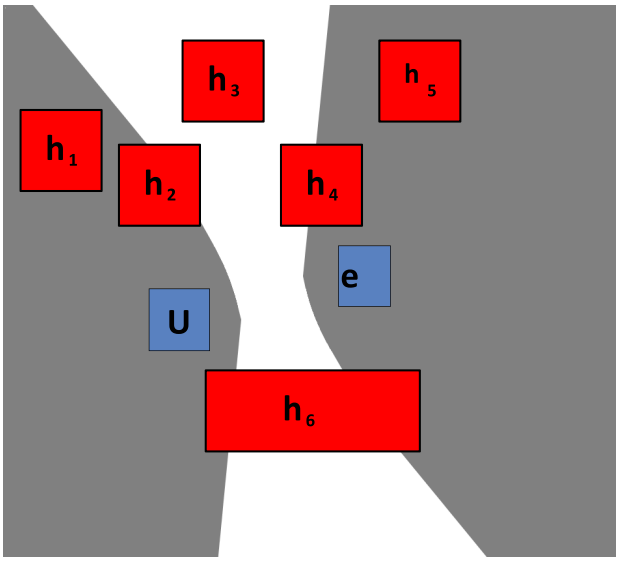


Figure 9.9: Cases of domination for a data index entry e.

where more data entries are refined, it turns out that this refinement was unnecessary. Yet, we will show that our hybrid algorithm, presented in the following Section 9.8 is capable of reducing both the number of refined data entries, and the number of refined space entries significantly.

9.8 Indexing $\mathcal{DB}$ and $\mathcal{S}$

It seems apparent to combine the ideas of Section 9.6 and Section 9.7 and utilize both index structures ($\mathcal{I}_{\mathcal{DB}}$ and $\mathcal{I}_{\mathcal{S}}$) to boost the performance. The non trivial task is the definition of a traversal order to minimize necessary operations.

Prelude

Our approach is basically a depth-first traversal of $\mathcal{I}_{\mathcal{S}}$. Additionally we define $AS_{\mathcal{DB}}$ to be the active set of entries of the index $\mathcal{DB}$. Each entry $h \in \mathcal{I}_{\mathcal{S}}$ has its own active set and passes it on to its children (always removing irrelevant entries $e \in AS_{\mathcal{DB}}$). $AS_{\mathcal{DB}}$ contains all entries of $\mathcal{DB}$ which have already been seen and not yet resolved during the traversal of the algorithm. For each entry $h \in \mathcal{I}_{\mathcal{S}}$ we first try to identify one of the two following properties (cf Figure 9.9):

$\sum_{e \in AS_{\mathcal{DB}}: Dom(e,U,h)} weight(e) \geq k$. In the case $Dom(e, U, h)$, all objects in data entry e are guaranteed to dominate U. Thus, we can increase a counter $count_{black} := \sum_{e \in AS_{\mathcal{DB}}: Dom(e,U,h)} weight(e)$, which corresponds to the number of objects dominating U, by the weight of e. If $count_{black}$ reaches k, then h must not part of the possible UV cell $\mathcal{V}_k^{\exists}$ of U, and thus is colored unfilled in our visualization such as shown in Figure

9.7. If $count_{black} < k$, then h is colored black, but might be assigned a different color later in the algorithm.

$\sum_{e \in AS_{\mathcal{DB}}: Dom(U,e,h)} weight(e) > |\mathcal{DB}| - k$. In the case $Dom(e, U, h)$, no object in data entry e can possibly dominate U. If this is guaranteed to hold for at least $|\mathcal{DB}| - k + 1$ objects, then at most $k-1$ database objects can possibly dominate h, and thus h must be part of the guaranteed Voronoi cell $\mathcal{V}_k^{\forall}$ and is colored green in our visualization. Our algorithm uses a counter $count_{green}$ to count the weight of entries e for which $Dom(U, e, h)$ does *not* hold. As long as $count_{green} < k$, the space cell h must be colored in green in Figure 9.7.

If neither of the above cases hold, then we check if $\sum_{e \in AS_{\mathcal{DB}}: PDom(e,U,h)} weight(e) < k$ and $\sum_{e \in AS_{\mathcal{DB}}: PDom(U,e,h)} weight(e) \leq |\mathcal{DB}| - k$. If both conditions hold, then space region U must be inside $\mathcal{V}_k^{\exists}$ and must not be inside $\mathcal{V}_k^{\forall}$. This corresponds to the case where h is colored in blue in Figure 9.7. Our algorithm uses the counters $count_{green}$ and $count_{grayorblue}$ to check this case.

If neither of the above two cases hold, then no final decision can be made for space region U, and either the current entry h or an entry $e \in AS_{\mathcal{DB}}$ has to be resolved. Here we propose the following heuristics:

Case 2: $PDom(U, e, h) \Rightarrow$ resolve e or h depending on which one covers more space.
Intuition: uncertain area becomes small if both constructing objects are small

Case 3 $\neg PDom(U, e, h) \wedge \neg PDom(e, U, h) \Rightarrow$ resolve e.
Intuition: Resolving h can not yield any new information, since any child of h must also yield Case 3.

Case 4 $PDom(e, U, h) \Rightarrow$ resolve h if we find another data entry for which Case 4 holds (for this space entry h). Otherwise resolve e or h depending on which one covers more space. If e is a leaf entry only resolve h.
Intuition: If more than one data entry constructs Case 4, chances are good that large portions of h can be decided.

Case 6 $PDom(U, e, h) \wedge PDom(e, U, h) \Rightarrow$ resolve h. (cf Figure 9.9, case 6)
Intuition: Resolving e can not yield any new information

Clearly, at one point there may be multiple data entries in the activate set of a space node h, which may yield different cases. It may be smart to prioritize the refinement of some data entries. In a nutshell, a data entry should be chosen which maximizes the chance that we can guarantee that h is not part of $\mathcal{V}(U)$. We propose to choose an entry e according to the following priority schema:

1. directory entries are prioritized over leaf entries.
2. prioritize cases in order 5, 4, 6, 3, 2, 1.

Algorithm 5: UV-Cell computation

Require: U,$\mathcal{I}_{\mathcal{DB}}$,$\mathcal{I}_{\mathcal{S}}$

1: $AS_{\mathcal{DB}}$ = windowQuery*(U,$\mathcal{I}_{\mathcal{DB}}$)
2: UVCellCheck(U,$\mathcal{I}_{\mathcal{S}}.root$,$AS_{\mathcal{DB}}$)

3. prioritize entries according to mindist to query

For ease of presentation of our algorithm, we define the function $maxprio(U \in \mathcal{DB}, h \in \mathcal{I}_{\mathcal{S}}, E \subseteq \mathcal{I}_{\mathcal{DB}})$ which maps an uncertain object U, a space region h and a set of data index entries E to the object which has the highest priority corresponding to the heuristics above.

Algorithm 5:

Takes as parameters the object U for which the UV-cell is to be computed; the database $\mathcal{DB}$ indexed by an R^*-tree $\mathcal{I}_{\mathcal{DB}}$; and the Quadtree/KD-trie $\mathcal{I}_{\mathcal{S}}$ indexing the space. The idea of Algorithm 5 is to build an initial active set $AS_{\mathcal{DB}}$ that is reasonable for all space partitions $h_i \in \mathcal{I}_{\mathcal{S}}$ to come during query processing. For this we perform a window-query-like operation. windowQuery*(U,$\mathcal{I}_{\mathcal{DB}}$) basically performs a window query on $\mathcal{I}_{\mathcal{DB}}$, but discards entries $e \in \mathcal{DB}$ that fall in the window (since these entries cannot help to decide the borders of $\mathcal{V}(U)$). The result are now all entries $e \in \mathcal{I}_{\mathcal{DB}}$ that have been seen during the window-query but have not been resolved. This set is then used as an initial *active set*(denoted as $AS_{\mathcal{DB}}$) in the recursive Algorithm 6 which is initiated by Algorithm 5.

Algorithm 6:

This algorithm requires the uncertain object U for which the UV-cell is being computed, *one* region of the result space h(initially the root of the kd-tree), the active set $AS_{\mathcal{DB}}$ containing a set of $\mathcal{I}_{\mathcal{DB}}$-entries, and the parameter k. The algorithm works as follows:

- The main loop (line 3) iterates over the active set until either the space region has been flagged or there are no more $\mathcal{I}_{\mathcal{DB}}$-entries to consider.
- In another loop (lines 4 – 25), the algorithm tests whether the current active set suffices for flagging the space region. To achieve this, a set of counter variables is defined:
  - $count_{green}$ checks for entries that include parts of the result space in their outer cell ($domCase > 1$). The space can only be part of $\mathcal{V}_k^{\forall}(U)$ if this counter is below k.
  - $count_{gray}$ tracks border regions around the inner cell that cannot be further expanded because the spatial index is at its maximum depth (isLeaf(h) is true). All domCases greater than 2 qualify to increase this counter.

Algorithm 6: UVCellCheck

Require: U,h,$AS_{\mathcal{DB}}$,k

1: e_{max} *//entry with maximum priority*
2: $count_{green}$, $count_{gray}$, $count_{grayorblue}$, $count_{blue}$, $count_{black} = 0$
3: **while** $AS_{\mathcal{DB}} \neq \emptyset$ **do**
4: **for all** $e \in AS_{\mathcal{DB}}$ **do**
5: **if** case$(e, U, h) > 1$ **then**
6: $count_{green}$ += weight
7: $e_{max} = maxprio(e_{max}, e)$
8: **end if**
9: **if** case$(e, U, h) \in 3, 4, 5, 6$ **then**
10: $count_{gray}$ += weight
11: **end if**
12: **if** case$(e, U, h) \in 3, 4, 5$ **then**
13: $count_{grayorblue}$ += weight
14: **end if**
15: **if** case$(e, U, h) \in 4, 5, 6$ **then**
16: $count_{blue}$ += weight
17: **end if**
18: **if** case$(e, U, h) = 5$ **then**
19: $count_{black}$ += weight
20: **end if**
21: **if** $count_{black} \geq k$ **then**
22: h is not part of UVCell
23: return
24: **end if**
25: **end for**
26: **if** $count_{green} < k$ **then**
27: h is part of inner UVCell $\mathcal{V}_k^{\forall}$
28: return
29: **end if**
30: **if** $count_{gray} < k$ **and** isLeaf(h) **then**
31: h is border of inner UVCell $\mathcal{V}_k^{\forall}$
32: return
33: **end if**
34: **if** $count_{blue} < k$ **and** $count_{grayorblue} \geq k$ **then**
35: h is part of outer UVCell $\mathcal{V}_k^{\exists}$
36: return
37: **end if**
38: **if** $count_{black} < k$ **and** isLeaf(h) **then**
39: h is border of outer UVCell $\mathcal{V}_k^{\exists}$
40: return
41: **end if**
42: $splitData = false$, $splitSpace = false$
43: **if** isLeaf(h) **or** case$(e_{max}, U, h) = 3$ **or** p(e_{max}) > p(h) **then**
44: **if** ¬isLeaf(e_{max}) **and** case$(e_{max}, U, h) \neq 6$ **then**
45: $splitData = true$
46: **end if**
47: **end if**
48: **if** ¬isLeaf(h) **and** case$(e_{max}, U, h) \neq 3$ **then**
49: **if** isLeaf(e_{max}) **or** case$(e_{max}, U, h) = 6$ **or** p$(e_{max}) \leq$ p(h) **or** $\neg splitData$ **then**
50: $splitSpace = true$
51: **end if**
52: **end if**
53: **if** $\neg splitSpace$ **and** ¬isLeaf(h) **then**
54: $splitSpace = true$
55: **end if**
56: *//Execute splits*
57: **if** $splitData$ **then**
58: $AS_{\mathcal{DB}} = AS_{\mathcal{DB}} \backslash e_{max} \cup e_{max}.children$
59: **end if**
60: **if** $splitSpace$ **then**
61: **for all** $h_c \in h.children$ **do**
62: UVCellCheck(U,h_c,$AS_{\mathcal{DB}}$.clone())
63: **end for**
64: **end if**
65: **end while**

- $count_{grayorblue}$ keeps track of undecided cases ($domCase = 3$) that need to be further explored before flagging.
- $count_{blue}$ counts entries that include parts of the result space in their inner cell ($domCase > 3$). The space can only be part of $\mathcal{V}_k^{\exists}(U)$ if this counter is below k.
- $count_{black}$ tracks border regions around the outer cell that cannot be further expanded because the spatial index is at its maximum depth (isLeaf(h) is true). Only domCase 5 increments this counter.

If an entry qualifies to increment a counter, the entry's weight is added. If entry is a leaf, i.e., is not an index page, the weight is one. Otherwise, weight is the number of all leaf entries included in this entry. Consideration of $\mathcal{I}_{\mathcal{DB}}$-entries is interrupted if more than k dominating cases ($domCase = 5$) have been found (lines 21 – 23). For further exploration, the entry with highest priority is determined in line 7.

- Once every entry has been checked, the algorithm tests if the space region can be assigned a flag based on the counters in lines26 – 40. Once a flag has been assigned, the recursion branch ends.
- If no flag could be assigned, the algorithm decides which index will be refined next – e_{max} (data split) or h (space split). The dicision is made based on whether once of the indices is already fully refined (isLeaf() is true), which of the two pages has the larger perimeter (p(e_{max}) $<$ / $>$ p(h)), and the domCase of e_{max}, namely:

Case 4: there is a chance that refining h may allow child entries of h to be pruned, and refining e_{max} may allow child entries of e_{max} to prune all of h. Therefore, we refine both entries in this case.

Case 6: refining e cannot possibly allow us to prune h. However, refining h may allow us to either prune children of h or to return children of h as true hits. Thus we refine h.

Case 3: no children of h can possibly be pruned. [3] Thus we split e_{max}, which may allow h to be pruned.

Case 2: we refine h.

- Finally, space index entries h which must be completely contained in $\mathcal{V}(U)$ are identified as entries having only **Cases 1-3** in their active set. Computation breaks if this is the case. After splitting the objects according to the rules above. We recursively restart the algorithm with the new objects.

Figure 9.10 illustrates in which manner the algorithm resolves entries of $\mathcal{I}_{\mathcal{DB}}$ and $\mathcal{I}_{\mathcal{S}}$. The figures shows all pages and objects of $\mathcal{I}_{\mathcal{DB}}$ which have been seen during the computation of the possible Voronoi-cell $\mathcal{V}(U)$ of the green objects U. Refined data objects are

[3] recall that if $e_{max}^{\mathcal{DB}}$ corresponds to case 3, then there exists no R^*-entry such that case 4 holds

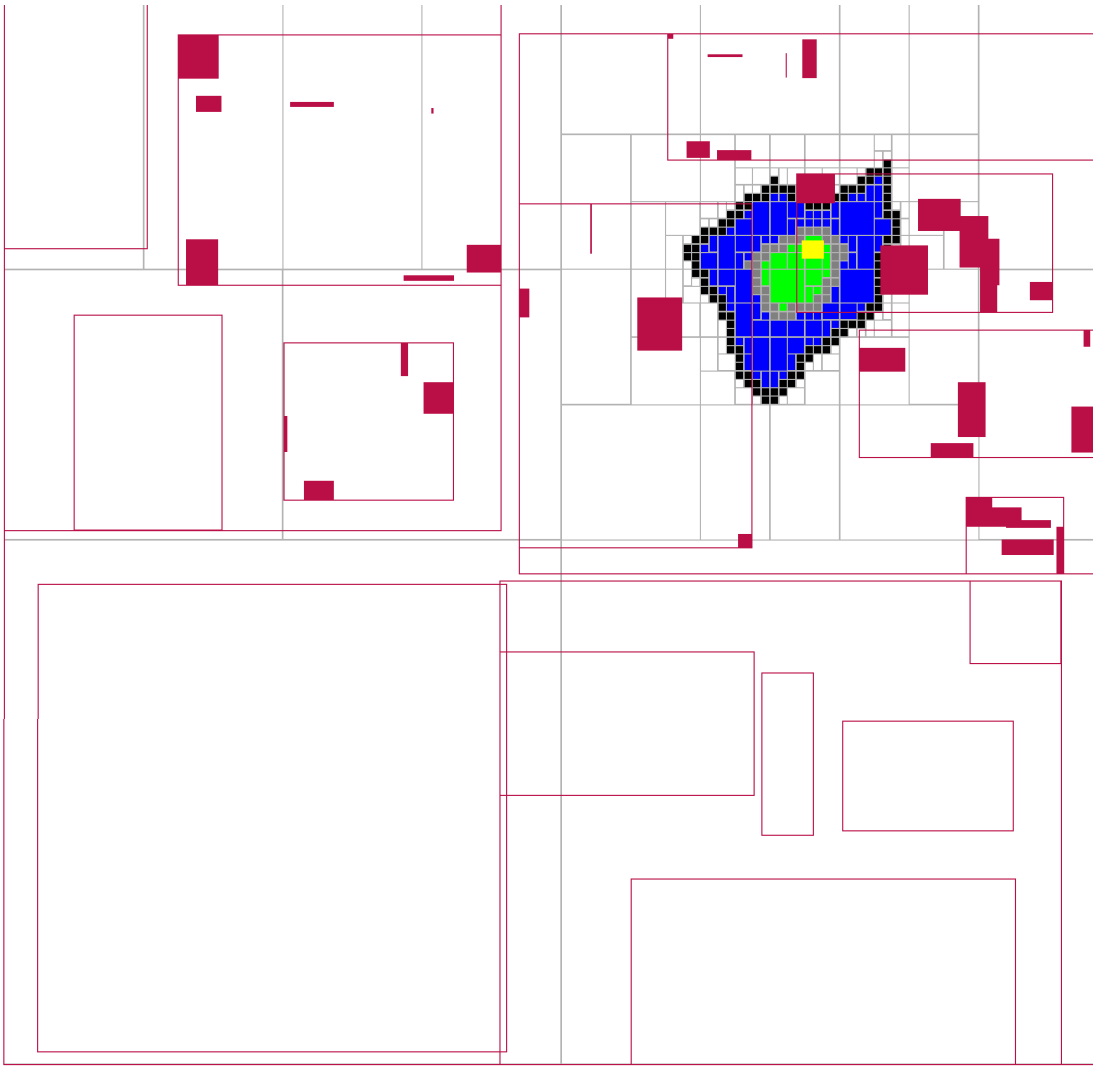


Figure 9.10: Example of refinement

represented by filled red rectangles and refined directory nodes are represented by unfilled red rectangles. Furthermore, refined entries of $\mathcal{I}_{\mathcal{S}}$ are shown as (i) unfilled black rectangles if they are guaranteed to be fully outside of $\mathcal{V}(U)$, (ii) as black rectangles if on the border of $\mathcal{V}(U)$, and (iii) as blue rectangles if completely inside $\mathcal{V}(U)$. We can observe that in areas far away from the UV cell, $\mathcal{I}_{\mathcal{S}}$ is resolved coarse whereas at the border of the cell it is resolved at very fine granularity. The entries of $\mathcal{I}_{\mathcal{DB}}$ are also only resolved around the UV cell. Note that although the number of resolved objects seems large, most of the objects are only needed for a small fraction of the computations, especially on coarser levels of $\mathcal{I}_{\mathcal{S}}$. Finally, note that a nice side effect of this computation is that we obtain a tight superset of the (uncertain-) Delaunay neighbors of U. This can be achieved by memorizing the objects O for which Case 4 or Cast 6 (see Definition 24) holds.

9.9 Experiments

Our experimental evaluation investigates algorithm behaviour w.r.t. maximum kd-trie depth, database size, uncertainty, and data dimension. To assess the uncertainty in our synthetic data, we define a parameter *extent* to control the size of the uncertain objects (i.e., the object's MBR) and corresponds to the maximum extent of an object in one dimension. Experiments use synthetically generated datasets as well as an excerpt from the T-Drive taxi-cab trajectory dataset[205, 204] where we added synthetic uncertainty to each GPS signal of a taxi cab. We implemented all approaches in the ELKI framework[3], which also provided an R-tree implementation.

The data space is always normalized to [0,1] in each dimension. For our synthetic data, objects are uniformly distributed over space with a randomly assigned side length between 0 and maximum extent. Most examples previously used in this work use synthetic data generated this way, including Figures 9.5(a)-9.5(d), Figure 9.6, Figure 9.7, Figure 9.8 and Figure 9.10.

Data points from the real world dataset were sampled as a single snapshot of the world, on the afternoon of February 2nd, 2008. Therefore, one data point corresponds to the position of one taxicab within the city of Beijing, China. After removing some outliers, this dataset contains 890 separate entities. To suit our application of location obfuscation, sample locations were randomized using a Gaussian distribution based on this object's location. A single sample from this distribution is then set as center of the object's new MBR, with its extent set to 6σ of this object's Gaussian (3σ to each direction). On said city scale, an extent of 0.01 would equal an area of 100m side length.

Parameter	default value	Notation	Algorithm
Dimension	2	DI	Data Index traversal (Section 9.6)
db size	1000	SI	Space Index traversal (Section 9.7)
extent	0.01	DSI	Data & Space Index traversal (Section 9.8)
tree depth	14	SR	Single Rectangle (Implementation of [206])

Table 9.2: Default settings.

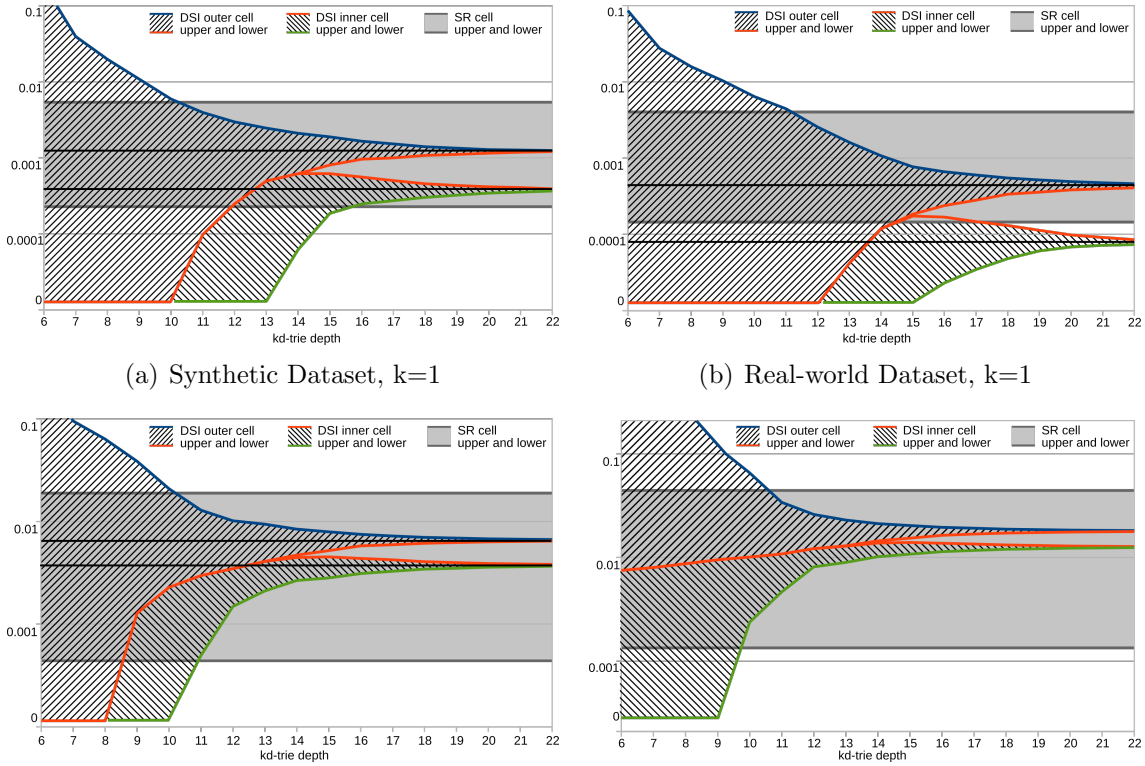
Table 9.2 denotes input parameters and their default settings, as well as an explanation of our algorithm notation. If not otherwise specified, the following experiments use the specified default values. Our evaluation focusing on approximation quality use DSI exemplarily for all algorithms from Sections 9.6–9.8, since these algorithms do not differ in the resulting approximation quality, but in efficiency only. Naturally, our real world dataset T-Drive has inherent values that override parameters, namely dimension and size of database. The standard depth of 14 refers to a maximum of 14 splits in our index structure, corresponding to $16384 (= 2^{14})$ individual grid cells. Applied to a city scale of 10 by 10 kilometers, each grid cell side would measure some 78 meters. As the proposed approach is later scaled up to a depth of 22, grid cells correspond to an area of only 4.8 by 4.8 meters, which on a city scale is extremely precise.

9.9.1 State of the Art Competitor

As a competitor solution to validate our computation of uncertain first-order Voronoi-cells, in terms of run-time and effectiveness, is the approach of [206]. This approach approximates the possible Voronoi-cell of an uncertain object by a single rectangular region. However, this approach is only applicable to approximate the possible first-order Voronoi cells $\mathcal{V}_1^\exists(U)$ of an uncertain object U. To the best of our knowledge, our solution is the first to compute higher-order Voronoi cells for uncertain data, and the first to compute a guaranteed Voronoi cell. In order to allow a fair experimental evaluation, we extend the solution of [206] to approximate higher-order possible Voronoi cells as described in the following.

In a nutshell, the approach of [206] computes a progressive and a conservative rectangular approximation of $\mathcal{V}_1^\exists(U)$. An initial conservative approximation is obtained by bounding the whole data space by a single rectangle. This approximation trivially covers $\mathcal{V}_1^\exists(U)$. Then, this rectangle is split recursively, where split dimensions and locations are chosen heuristically. Whenever a split is performed, the concept of spatial domination (c.f. Section 9.3) is applied to see if any of the new residual regions obtained from the split completely contains $\mathcal{V}_1^\exists(U)$. If not, the split if rejected and a new split dimension and location are chosen heuristically. If one residual split region does contain $\mathcal{V}_1^\exists(U)$, then the algorithm recursively continues with that region. After each split, only at most one region can completely contain $\mathcal{V}_1^\exists(U)$. The algorithm terminates when the gain by further splitting the current approximation of $\mathcal{V}_1^\exists(U)$ drops below a specified threshold. A progressive approximation of $\mathcal{V}_1^\exists(U)$ is obtained by choosing the boundaries of U itself as an initial approximation. This initial approximation is progressive, as any point inside U is guaranteed to have a chance U as a nearest neighbor, as U might be located at the exactly same location. Then, the progressive approximation is again expanded iteratively, using the concept of spatial domination to check whether a new approximation is completely contained within $\mathcal{V}_1^\exists(U)$. Note that for each of the spatial domination checks, each database object has to be tested for spatial domination, since no index structures are employed.

To apply this solution for higher-order Voronoi cells, we adapt the spatial domination check of [206] as follows: given the current conservative approximation C, rather than finding a single database object O such that $Dom(O, U, C)$, i.e., such that O is guaranteed to dominate U with respect to C, we now continue the domination checks until k such objects are found. Analogously, for the current progressive approximation P, we now find k objects O such that $PDom(O, U, P)$ holds, that is, we find k objects which are might possibly dominate U, rather than just attempting to find one such object, as done by the algorithm proposed in [206]. In the following experimental evaluation, the resulting adapted algorithm of [206], which computes a single rectangular progressive approximation and a single rectangular conservative approximation, will be used as our competitor approach. This approach will be denoted as *Single Rectangle (SR)*.



(a) Synthetic Dataset, k=1

(b) Real-world Dataset, k=1

(c) Synthetic Dataset, k=5

(d) Synthetic Dataset, k=15

Figure 9.11: Approximation Quality for DSI and SR

9.9.2 Approximation Quality

Our first evaluation explores how well the generated bounds approximate a cell. Therefore, we set the tree depth for our implementation to various levels between 5 and 22, corresponding to the number of splits. Evidently, smaller grid cells can more closely follow the outline of a UV-cell.

Figure 9.11 visualizes how upper and lower bounds converge with larger tree depth. The dark blue line refers to the upper bound of DSI, the orange line to its lower bound, each represented by the total volume of their cells. The hatched space in between the two lines refers to the range in which the true cell volume must be located. As a point of reference, upper and lower bounds from the *Single Rectangle (SR)* approach have also been denoted in the same graphic, with the area shaded in grey corresponding to the approximation error. Since SR does not use an index, its results remain unchanged for all settings of the maximum tree depth.

Performance was tested on different datasets. Figure 9.11(a) represents average results for runs on synthetic data, while Figure 9.11(b) contains the results for our real world dataset. While overall performance is fairly comparable, DSI provides a usable lower bound remarkably early, with as little as 8 tree splits necessary to outperform SR. We can see that for a sufficiently tree depth between 18 and 22, the lower- and upper-bound

approximation of the inner-Voronoi cell $\mathcal{V}_k^{\forall}(U)$ and the outer-Voronou cell $\mathcal{V}_k^{\exists}(U)$ approach each other. Consequently, the approximation error converges to zero, thus indicating that our approach is able to near-optimally approximate $\mathcal{V}_k^{\forall}(U)$ and $\mathcal{V}_k^{\exists}(U)$. We further observe that for larger values of k, the space covered by the possible Voronoi-cell $\mathcal{V}_k^{\exists}(U)$ and the guaranteed Voronoi-cell $\mathcal{V}_k^{\forall}(U)$ seem to approach each other. The reason is that all cells grow radially in k. Thus, the space covered by both cells $\mathcal{V}_k^{\forall}(U)$ and $\mathcal{V}_k^{\exists}(U)$ cover space quadratic in the diameter of each cell. Yet, the space covered by the set-difference $\mathcal{V}_k^{\exists}(U) \setminus \mathcal{V}_k^{\forall}(U)$ forms the shape of a ring, the surface of which grows only linear in it's diameter. Thus, we observe that indeed both cells $\mathcal{V}_k^{\forall}(U)$ and $\mathcal{V}_k^{\exists}(U)$ increase for larger k, but they relative size approaches each other for large values of k. Finally, our competitor approach SR shows fairly similar behaviour on both datasets, with results looking even more similar than they are due to logarithmic scale. We argue that the large approximation error of SR makes it a bad choice for reliable decision making. The following subsection will answer the question if the SR approach can redeem itself in terms of run-time.

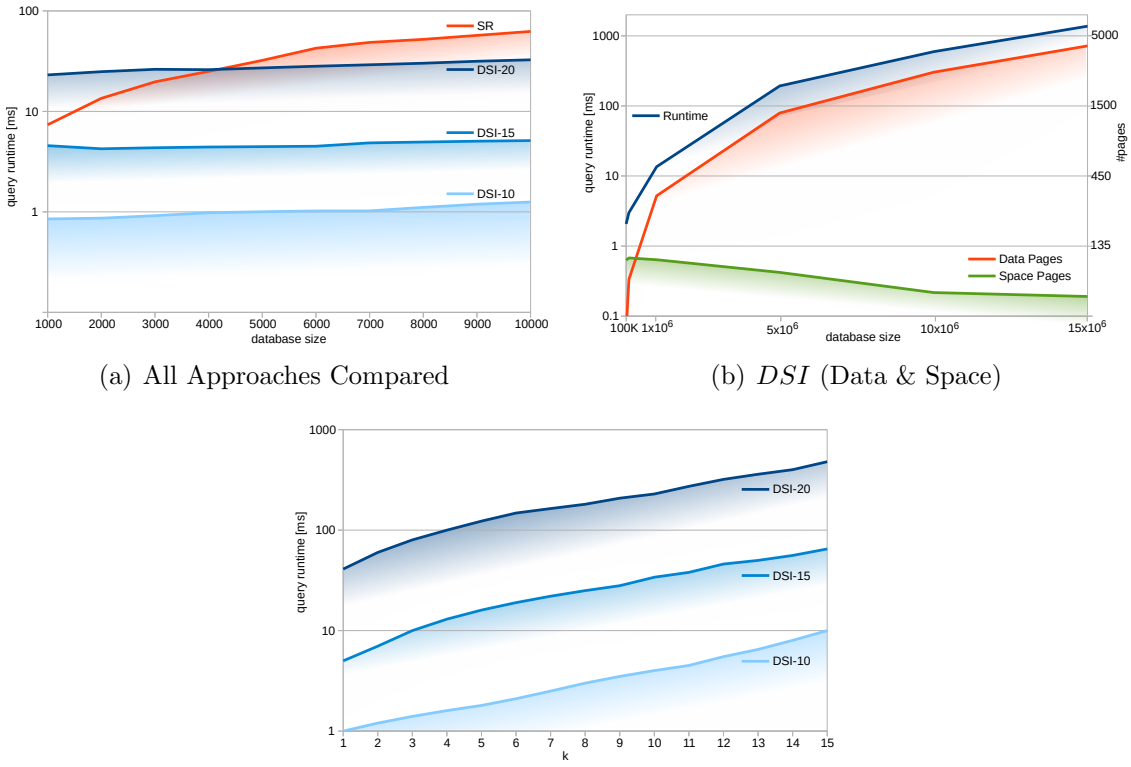
9.9.3 Algorithmic Runtime

Runtime experiments were conducted by varying database size and dimensionality, between our three different traversal approaches compared to SR as well as for DSI alone to cover larger ranges of database size (other approaches have been excluded for large database sizes due to their excessive run-time performance).

In Figure 9.12, run times to calculate one UV-cell are denoted over different database sizes. Figure 9.12(a) contains results for the approach *Data and Space Index Traversal (DSI)* in three different configurations of kdtrie-depth, and SR. As is to be expected, DSI has higher runtimes for higher depth of its spatial index, since border regions will be explored further and require more domination checks. However, runtime increases only very lightly for greater database sizes. This is because the combined approach of data and space index allows for early pruning of large portions of the database. SR starts off at a considerable speed, but since it features pairwise comparisons without the use of an index, it does not scale well for higher numbers ob database objects.

As query performance generally deteriorates for larger datasets, further scaling experiments were conducted using DSI only. Figure 9.12(b) shows the results of database populations frm 10K to 15 Million objects. To avoid gross overlapping of objects, object extent has been lowered to 0.001 for these runs. The left axis again refers to the average time to perform one UV-cell calculation, which corresponds to the blue data line. We observe a slightly superlinear scaling, confirming our theoretical observations that (i) adding more objects leads to linearly more intersections with Voronoi cells, which are at least as big as U, and (ii) a linear increase in object count causes logarithmic tree index growth. This results in a combined log-linear growth in runtime.

The right scale denotes average page views during cell calculation, with the orange line referring to pages of the data index, and the green line for pages of the space index. Note that data index exploration roughly follows runtime development, while the space index is used less for larger databases. This is easily explained by a constant tree depth, resulting



(a) All Approaches Compared

(b) *DSI* (Data & Space)

(c) *DSI* (Data & Space)

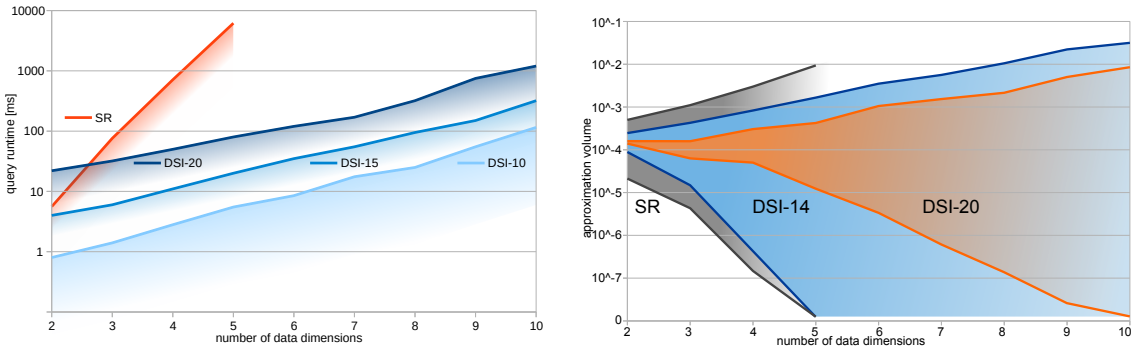
Figure 9.12: A runtime comparison for our *DSI* and the *SR*-approach over different sizes of *DB*

in a constant resolution of space. With a higher database population, the likelyhood of all relevant objects being enclosed in a small space increases.

9.9.4 Effect of data dimensions

Although the simple case of a two-dimensional world is most intuitive for most applications mentioned before, all approaches can manage high-dimensional datasets as well. The main limitation here is keeping the approximation error low in all dimensions at once, as well as balancing computational complexity.

Figure 9.13 displays performance for different kdtrie-depths of our *DSI* approach (depths 10, 15 and 20) as well as *SR* for multi-dimensional datasets. As runtime and memory usage of *SR* do not scale well for more than five data dimensions, experiments excluded this approach for higher dimensionalities than 5. An evaluation of runtime as shown in Figure 9.13(a) shows constant increase for all approaches. The relative steepness of increase is due to the growing inefficiency of pruning methods in high dimensions, which deteriorates searches toward a linear scan, which itself has quadratic complexity. As expected, runtime of *DSI* increases with a higher kdtrie-depth, since border regions in the



(a) Runtime (b) Approximation Quality

Figure 9.13: A comparison for increasing data dimensions.

spatial index will cause repeated domination checks.

Approximation quality for higer dimensions is shown in Figure 9.13(b). As mentioned before, fitting a bound to a more and more complex object leaves much room for approximation error. Therefore, volumes of upper and lower bounds diverge more for higher dimensions. Displayed here are bounds for SR up to dimension 5 (grey) and two different settings of our DSI approach, once with a depth of 14 (blue) and a depth of 20 (orange). As expected, a higher depth allows for more tree splits per dimension and thus a better approximation.

9.10 Conclusions

In this work, we propose an index-supported approach approximating the shape of Voronoi-cells to support nearest neighbor queries on uncertain data. In an uncertain database the attribute values of objects are random variables. Consequently, the location, size and shape of the Voronoi-cell of an object U is a random variable, too. Therefore, we propose to approximate, progressively and conservatively, the space which is guaranteed to be part of the Voronoi-cell of U, denoted as the guaranteed Voronoi-cell $\mathcal{V}_k^{\forall}$. In addition, we also approximate the space which has a non-zero probability to be part of the Voronoi-cell of U, which we call the possible-Voronoi cell $\mathcal{V}_k^{\exists}$. We show how our solutions can be extended to k'th-order Voronoi cells, i.e., the space which is guaranteed to (may possibly) have U as one of their k-nearest neighbors. Our approach uses an R^*-tree as a hierarchical access method to efficiently find the set of uncertain objects that influence the possible Voronoi-cell of an uncertain object U, i.e., the set of Delaunay-neighbors of U. In addition, we propose to use a kd-trie as a hierarchical access method to identify regions of space which must (not) be part of a Voronoi-cell. Compared to the state-of-the-art of computing uncertain Voronoi-cells, our approach allows for much higher approximation quality, since our result approximation consists of a set of rectangular kd-trie nodes, rather than a single bounding rectangle.

Chapter 10

Approximate Probabilistic Representative Pattern Mining

10.1 Introduction

This chapter targets the problem of deriving a meaningful clustering from an uncertain dataset. For this purpose, our aim is not to develop a new clustering algorithm, but rather to allow clustering algorithms designed for certain data to return meaningful, reliable and correct results in the presence of uncertainty. To illustrate the challenge that arises by considering uncertain data, consider the work-flow depicted in Figure 10.1. Figure 10.1(a) shows a dataset containing six two-dimensional points, which correspond to the positions of moving objects at some point of time t. The shaded region in Figure 10.1(a) corresponds to a lake, which none of the moving objects may cross. To cluster the locations of these objects, a domain expert may opt to choose a clustering algorithm $\mathcal{C}$ from a suite of available options (e.g., a density-based algorithm [110], e.g., DBSCAN [59] or OPTICS [9], HDBSCAN [26], or be it one of the numerous variants from the k-means family [88]). Assuming that the true locations of the objects are known, abstractly referred to as $\mathcal{C}$, which can, in a perfect world where there is no uncertainty, be used to obtain the clustering of the true positions of the moving objects as shown in Figure 10.1b). However, the true locations of the objects could be unknown and we may only have access to the last reported observations of these objects (e.g., by their GPS devices), shown as triangles in Figure 10.1(d). In such a scenario, an *uncertainty data model* is typically used to capture the distribution of the possible object locations. Given recent observations of an uncertain object, a large body of research work has been published to obtain accurate uncertainty models for various application domains, including bioinformatics [72, 125], moving-object-databases [57, 37], data integration [52]. Figure 10.1(e) shows exemplary *probability density functions* (PDFs) around the observations shown in Figure 10.1(d). Object A, for instance, is likely to be moving around the lake (since movement inside the lake is impossible), while the movements of other objects are less constrained. If we follow a simplistic approach for clustering the data, by simply clustering the expected (according

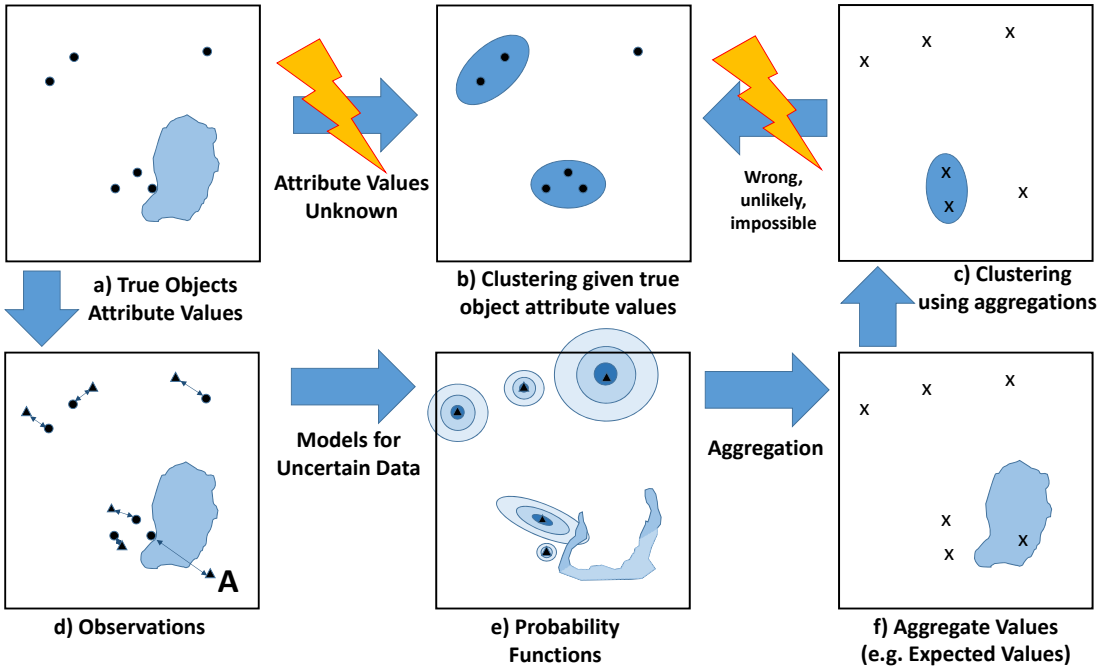


Figure 10.1: Uncertain Clustering Workflow.

to the uncertain data model) values of the objects, as shown in Figure 10.1(f), then we may end up in deriving a clustering which arbitrarily deviates from the clustering of the actual (but unknown) locations of the objects (e.g., compare the clustering of Figure 10.1(c) with true clustering of the data as shown in Figure 10.1(b)). Therefore, approaches that aggregate the information given by the uncertain model to expected values and then apply clustering may yield results of poor quality, due to information loss and potentially due to invalid input to the clustering process (e.g., the expected location of A after aggregating all its possible positions in Figure 10.1(f) is in the lake).

In this work, our goal is neither the definition of new uncertain data models suited for clustering, nor the proposal of new clustering algorithms tailored for uncertain data. Instead, we aim at making possible the application of *any* existing clustering algorithm $\mathcal{C}$ on an uncertain database $\mathcal{DB}$, based on *any* given uncertain data model for $\mathcal{DB}$, assuming that $\mathcal{C}$ would be appropriate for $\mathcal{DB}$, if $\mathcal{DB}$was certain. For example, we take that the clustering of Figure 10.1(b) by algorithm $\mathcal{C}$ is the ideal one, but infeasible to derive, given that we do not know the actual locations of the objects. The objective of our framework is to use the data of Figure 10.1(e) and algorithm $\mathcal{C}$ to derive a clustering that has a probabilistic guarantee (according to the uncertain data model used) to be very similar to that of Figure 10.1(b).

Our approach performs sampling and then represents the original uncertain database $\mathcal{DB}$ as a set of sample deterministic databases. We then run the clustering algorithm $\mathcal{C}$ on each of the sampled databases, to derive a set of possible clusterings PC. Our main contribution is to combine the resulting set PC into a concise set RPC of *representative*

clusterings. Furthermore, using all clusterings in PC, we estimate a probability ϕ of each representative clustering in RPC, defined as the probability that this representative clustering does not exceed some maximum distance τ to the true clustering. Since ϕ has to be estimated from sampled databases, we obtain a lower bound of ϕ that is significant at a user specified level. We provide a methodology to derive an RPC for a given value of τ; and therefore impose quality constraints on the uncertain clustering results, unlike previous approaches on uncertain data clustering [134, 69, 73, 70, 117, 105, 135, 71, 94, 114] which cannot provide quality guarantees.

In summary, our contributions are as follows.

- We propose a sampling-based solution to cluster uncertain data. This solution is generally applicable to all data domains and with any suitable uncertain data model, allowing the application of any existing clustering algorithm, originally designed for certain data. As opposed to previous work on uncertain clustering, our approach conforms to the possible worlds semantics and also considers any dependencies between objects.

- We present a methodology via which we can assess the quality of a clustering of a possible world compared to the true clustering of the data.

- We show how the confidence of clustering results on possible worlds can be improved by computing a set of multiple representative clusterings, each having a significant likelihood to resemble the true, unknown clustering.

The rest of this chapter is organized as follows. Section 10.2 surveys existing methods for clustering uncertain data. Section 10.3 gives general definitions used in the remainder of this work. Section 10.4 shows how we can estimate the probability of the clustering result on a possible world to be the clustering of the true data values. Section 10.5 shows how, from a set of possible clusterings, we can find *representative clusterings* that are probabilistically guaranteed to be similar to the real clustering. Section 10.6 includes an experimental evaluation that supports our findings and Section 10.7 concludes this work.

10.2 Related Work

Clustering is undoubtedly one of the most important tools for unsupervised classification. A large number of clustering algorithms have been developed, as reflected in numerous surveys [89, 90, 111, 88, 110, 168]. Although clustering has proved its applicability in many different domains and scenarios, the problem of clustering uncertain data has only gained little attention so far. Uncertain data clustering approaches either use expected distances between objects or assume that the distances between different pairs of objects are independent. In this work we review these methods and discuss their drawbacks.

10.2.1 Clustering Using Expected Distances.

The main drawback of approaches based on expected distances [134, 69, 73] is the information loss incurred by describing a complex probability distance function by a single scalar. Considering additional moments of a probability distance function, such as deviation [70] works well in specific applications where objects have little uncertainty, e.g., in applications where each object can always be described by a single parametric PDF such as a single Gaussian. Still, the quality of such approaches cannot be assessed, rendering them inappropriate for applications where decisions are to be made based on the computed clustering. As an example consider the setting of Figure 10.2, having two certain objects A and B, and an uncertain object U having two possible values U_1 and U_2. Now, assume a deterministic clustering algorithm $\mathcal{C}$ which clusters two objects only if they distance does not exceed $dist(A, U_1)$ $(= dist(B, U_2))$. Clearly, in the example of Figure 10.2, there are two possible clusterings, either the clustering having cluster $\{A, U\}$ and outlier B, or the clustering having cluster $\{U, B\}$ and outlier A. The probabilities of these possible clusterings equal the probabilities $0 < P(U_1) < 1$ and $P(U_2) = 1 - P(U_1)$ of alternatives U_1 and U_2 of U. However, using the expected distance between A and U given by

$$E(dist(A, U)) = P(U_1) \cdot dist(A, U_1) + P(U_2) \cdot dist(A, U_2),$$

objects A and U can never be located in the same cluster, since it holds that $E(dist(A, U)) > dist(A, U_1)$. The same holds for B and U due to symmetry. Thus, for the example of Figure 10.2, an approach using expected distance yields a clustering, which is entirely impossible. Summarizing, the use of expected distances is a heuristic to obtain a single clustering that represents the whole realm of possible clusterings, however, the derived result is not necessary similar to any of them. Most clustering algorithms using expected distances focus on improving efficiency rather than addressing this drawback [117, 105, 135]. Gullo et al. [71] propose a method, called UK-medoids, for which they utilize the expected distance between two uncertain objects. Jiang et al. [94] propose to use the KL divergence between two uncertain objects which is often used to reflect similarity between two probabilistic density functions. They investigate the applicability to both K-means [130] and DBSCAN [59]. Since all these approaches are based on expected distances, their results are not in accordance with the possible worlds semantics and do not carry any quality guarantee.

10.2.2 Clustering Assuming Independent Distances.

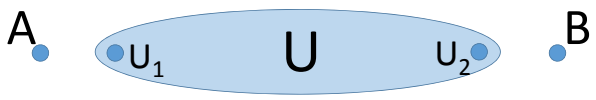


Figure 10.2: Example of Uncertainty

The uncertain clustering algorithms proposed in [113, 114] assume that pairwise distances between uncertain objects are mutually independent. This assumption may yield wrong results based on the possible world semantics; in addition, the results are biased toward overestimating the size of clusters. Kriegel and Pfeifle [113] propose variants of DBSCAN and OPTICS for uncertain

data, based on the distance independence assumption. In specific, the following is assumed about the distances between uncertain objects:

$$P(A \leftrightarrow_\epsilon B \wedge B \leftrightarrow_\epsilon C) = P(A \leftrightarrow_\epsilon B) \cdot P(B \leftrightarrow_\epsilon C),$$

where $P(X \leftrightarrow_\epsilon Y)$ denotes the probability that the distance between the two uncertain objects X and Y is smaller than a threshold ϵ. However this assumption does not hold in the general case. For example, in Figure 10.2, the two random events $A \leftrightarrow_\epsilon U$ and $U \leftrightarrow_\epsilon B$ are negatively correlated: for the case where $dist(A, U_1) < \epsilon < dist(A, U_2)$, it even holds that both random events are mutually exclusive. That is, if U is close to A then it cannot be close to B and vice versa. Relaxing this assumption of independent distances yields a computationally hard problem, as a distance value may depend on a large number of uncertain objects.

10.2.3 Discussion.

To our knowledge, there is no previous work on uncertain data clustering that conforms to the possible worlds semantics. A likely reason for this the hardness of the problem: as shown in [44], general induction on uncertain data is a #P-hard problem. In order to avoid the exponential run-time cost of considering all possible worlds, a common approach to handle uncertain data, in general, is sampling [93]. Given such a number of sample instances of the database (each corresponding to a possible world), a query or data mining task can be performed on each of them, and common results can be returned, associated with confidences of these results to be equal (or sufficiently similar) to the result on the true (but unknown) data. This is exactly the approach that we are following in this work. Our main contribution is the computation of a small number of representative clusterings that have high confidence values to be similar to the to the true (but unknown) clustering.

10.3 Definitions

Definition 25 (Clustering) *A clustering $\mathcal{C}(S)$ of a set $S = \{a_1, ..., a_N\}$ of deterministic objects, using a clustering algorithm $\mathcal{C}$, is a partitioning of S into pairwise disjoint subsets $C_1, ..., C_k \subseteq S$, such that $\bigcup_{1 \leq i \leq k} C_i = S$. Each subset $C_i, 1 \leq i \leq k$ is called a cluster.*

This abstract definition of clustering intentionally omits any objective rules toward a "good clustering", such as the requirement that similar objects should be in the same cluster. The reason is that our approach should be used in conjunction with any clustering algorithm $\mathcal{C}$, independently to the algorithm's objective.

Due to the $\#P$-completeness of general query processing (or mining) on uncertain data [44], coupled with the fact that an uncertain database may yield a number of possible clusterings exponential in the size of the database, we now explore the possibility of using a Monte-Carlo approach to perform clustering over uncertain data.

10.4 Clustering Sampled Worlds

Let $\mathcal{DB}$ be an uncertain database and let $\mathcal{C}$ be a clustering algorithm. Let $\mathcal{S} = \{w_1, ..., w_{|\mathcal{S}|}\}, w_i \in \mathcal{W}$ be a multiset[1] of possible worlds of $\mathcal{DB}$ generated from $\mathcal{DB}$ using generation rule $\mathcal{G}$ and let $\mathcal{C}(\mathcal{S})$ denote the multiset of clusterings obtained by clustering each sample world in X. We denote the set of distinct clusterings in $\mathcal{C}(\mathcal{S})$ as the set PC of *possible clusterings* obtained from sample $\mathcal{S}$. For any clustering C in PC, the *support* $C.supp$ of C is defined as $\sum_{i=1}^{|\mathcal{S}|} I(\mathcal{C}(w_i) = C)$, where $I(.)$ is an indicator function that returns 1 if its operand is true and 0, otherwise. Simply speaking, $C.supp$ is the number of occurrences of clustering C in the multiset $\mathcal{C}(\mathcal{S})$.

Lemma 7 (Approximation of the True Clustering) *Analogous to Section 8.7.1, for any possible clustering $C \in PC$, the probability $\hat{P}(C) = \frac{C.supp}{|\mathcal{S}|}$ is an unbiased estimator the probability $P(C)$ that C is the true clustering $\mathcal{C}(oracle(\mathcal{DB}))$ of $\mathcal{DB}$; i.e., $E(\hat{P}(C)) = P(C)$.*

Proof 7

$$E(\hat{P}(C)) = E\left(\frac{C.supp)}{|\mathcal{S}|}\right) = E\left(\frac{\sum_{w_i \in \mathcal{S}} I(w_i = C)}{|\mathcal{S}|}\right)$$

Since the expectation of a non-random variable is the identity, we obtain

$$E(\hat{P}(C)) = \frac{E(\sum_{w_i \in \mathcal{S}} I(w_i = C))}{|\mathcal{S}|},$$

where $I(pred)$ is an indicator function that returns one if predicate pred is true and zero otherwise. Due to the assumption of all sample databases w_i being drawn independently, and since the expectation of a sum of independent random variables is the sum of their expectations, we get:

$$E(\hat{P}(C)) = \frac{\sum_{w_i \in \mathcal{S}} E(I(w_i = C))}{|\mathcal{S}|}$$

Due to the assumption that each sample w_i is drawn unbiased from the distribution of all worlds of $\mathcal{DB}$, which implies that $E(I(w_i = C)) = P(w_i)$, we obtain

$$\frac{\sum_{w_i \in \mathcal{S}} P(w_i)}{|\mathcal{S}|} = P(w_i)$$

Such a straightforward sampling approach works well for small databases, including the running example depicted in Figure 10.2, where the number of possible clusterings C is reasonably small. In a large database setting, where the probability of finding exactly the same clustering on two samples in $\mathcal{S}$ approaches zero, this approach becomes inapplicable. The reason is twofold. On one hand, the probabilities $P(C)$ of a clustering C being the true clustering of $\mathcal{DB}$, become very small. Due to independent samplings $w_i, 1 \leq i \leq |\mathcal{S}|$, the number of samples where $\mathcal{C}(w_i) = C$ follows a binomial $B(\pi = P(C), n = |\mathcal{S}|)$ distribution.

[1] Due to independent sampling, the same sample may be drawn multiple times.

Estimating the probability parameter π of a binomial distribution given a sample, requires a very large sample size n if π is small. A rule of thumb is that $n \cdot \pi \geq 5$ [40, 188]. On the other hand, the large number of possible clusterings combined with small probabilities makes the exact results meaningless for a user. A huge, potentially exponentially large in the number of uncertain objects, set of possible clusterings, many of which may be very similar, yet they differ between each other are of little use.

10.5 Representative Clusterings

Our goal is to reduce the (potentially huge) set of clusterings produced by the Monte-Carlo approach to a small set of possible clusterings, which are diverse on the one hand and at the same time guaranteed to be similar to the clustering on the real (but unknown) database on the other hand. Our first approach (Section 10.5.1) follows a general concept for determining only one representative from a set, the medoid approach [178]. In Section 10.5.2, we generalize this approach to select a set of multiple representative clusterings and show how we can estimate how well they can approximate the real clustering. Section 10.5.3 presents a methodology for selecting a set of representative clusterings of guaranteed quality.

10.5.1 Sample Medoid

Let PC denote the set of possible clusterings derived from sampled worlds $\mathcal{S} = \{w_1, ..., w_n\}$. Let D be the *distance* $|PC| \times |PC|$ matrix such that

$$D_{i,j} := \text{dist}(PC_i, PC_j).$$

Here, dist denotes a distance measure between two clusterings. Literature on clustering provides a wealth of measures to assess the degree of agreement between two partitionings of the same set of objects. Examples of such similarity measures include the Rand Index [147], Jaccard [87], or the Adjusted Rand Index (ARI) [85]. Similarity usually takes a value between 0 (no agreement) and 1 (identical partitionings) and can be converted to a distance after subtraction from 1. These measures can be treated as distance measures [145, 2] by using the inverse (distance = 1-similarity).

The *median* of PC can be defined as

$$\text{Median}(PC) = \text{Median}_{w_i \in \mathcal{S}}(\mathcal{C}(w_i)) =$$

$$\arg\min_i \sum_{j=1}^{n} \text{dist}(\mathcal{C}(w_i), \mathcal{C}(w_j)) =$$

$$\arg\min_i \sum_{j=1}^{n} D_{i,j} \cdot PC_i.supp$$

The median clustering can be argued to be the most representative clustering out of all sampled clusterings $\mathcal{C}(w_i)$. However, Median(PC) has no associated confidence information, i.e., the deviation of the true clustering $\mathcal{C}(\text{oracle}(\mathcal{DB}))$ from Median(PC) is impossible to assess, hindering decision making. Furthermore, a single representative clustering may not be informative enough to capture most of the information described by all possible worlds.

It is important to note that Median(PC), albeit derived using expected distance between clusterings, does not suffer from the same drawbacks as existing works on clustering uncertain data using expected object positions and expected distances (cf. Section 10.2). The main difference is that Median(PC) is a clustering derived from a possible database instance that was generated consistently to the uncertainty data model, i.e., considering the value distributions and stochastic dependencies between objects.

10.5.2 Multiple Representatives

The possible clusterings of an uncertain database may be very heterogeneous; depending on object attribute values of a world, an individual cluster may become noise, may shatter into multiple clusters, or may be absorbed by another cluster in some worlds, but not in others. Such large changes in the overall clustering may be caused by minimal changes in the underlying dataset: the density of a critical region may drop below the threshold of a density-based clustering algorithm; a partition-based cluster representative may change slightly, yielding a new data partitioning and leading into a spiral of changes. Keeping this potential heterogeneity of possible clusterings in mind, a single sample medoid clustering could be insufficient: it may be an unlikely pivot between a number of likely clusterings and it may not even be similar to the most likely possible worlds.

Instead, a user may be more interested in a smaller set of representative clusterings, all having a significantly high probability to be similar (but not necessarily equal) to the true clustering.

We define a representative clustering as follows:

Definition 26 (Representative Clustering) *Let $\mathcal{DB}$ be an uncertain database and let $\mathcal{C}$ be a clustering algorithm. We call a clustering $\mathcal{C}(w_i)$ a τ-ϕ-representative clustering, if the probability*

$$P(w_i, \tau) := P(dist(\mathcal{C}(w_i), \mathcal{C}(oracle(\mathcal{DB}))) < \tau)$$

that the true clustering $\mathcal{C}(oracle(\mathcal{DB}))$ of $\mathcal{DB}$ has a distance $dist(\mathcal{C}(w_i), \mathcal{C}(oracle(\mathcal{DB})))$ of at most τ is at least ϕ.

Lemma 8 (Approximation of Representatives) *Let $\mathcal{XS} = w_1, ..., w_{|\mathcal{S}|}\}$ be a set of possible worlds of $\mathcal{DB}$ generated from $\mathcal{DB}$ using generation rule $\mathcal{G}$ and let $\mathcal{D}$ be a distance measure on clusterings. Let PC be the set of clusterings obtained from $\mathcal{S}$ associated with their supports. The probability*

$$\hat{P}(w_i, \tau) := \frac{\sum_{j=1}^{|\mathcal{S}|} I(dist(\mathcal{C}(w_i), \mathcal{C}(w_j)) \leq \tau)}{|\mathcal{S}|}$$

is an unbiased estimator of the probability

$$P(w_i, \tau) := P(dist(\mathcal{C}(w_i), \mathcal{C}(oracle(\mathcal{DB}))) \leq \tau)$$

that cluster representative w_i has a distance of at most τ to the true clustering of $\mathcal{DB}$. Here $I(pred)$ is an indicator function that returns one if predicate pred is true and zero otherwise.

Proof 8 *Analogous to the proof of Lemma 7, by substituting the indicator function $I(C = w_i)$ by $I(dist(\mathcal{C}(w_i), \mathcal{C}(w_j)) < \tau)$.*

Albeit unbiased, the probability $\hat{P}(w_i, \tau)$ cannot be used directly to assess the probability $P(w_i, \tau)$ of cluster w_i having a distance of at most τ to the true clustering $\mathcal{C}(\text{oracle}(\mathcal{DB}))$. Thus, w_i can not simply be returned as a τ-$\phi = \hat{P}(w_i, \tau)$-representative according to Definition 26 as the estimator $\hat{P}(w_i, \tau)$ may overestimate the true probability $P(w_i, \tau)$. To return significant representative τ-ϕ clusters to the user, our aim is to find a lower bound $\hat{P}(w_i, \tau, \alpha)$ such that we can guarantee that $P(w_i, \tau) \geq \hat{P}(w_i, \tau, \alpha)$ with a probability of α, where α is a domain specific level of significance (typically, $\alpha = 0.95$).

To derive such a significant lower bound of $P(w_i, \tau)$ we may exploit the fact that sampled possible worlds were drawn independently. Therefore, the absolute number $\hat{P} \cdot |\mathcal{S}|$ of sampled worlds which are represented by w_i follows a binomial $B(P(w_i, \tau), |\mathcal{S}|)$ distribution. To estimate the true probability $P(w_i, \tau)$ given realization $\hat{P} \cdot |\mathcal{S}|$, we borrow techniques from statistics to obtain a one sided $1 - \alpha$ confidence interval of the true probability $P(w_i, \tau)$. A simple way of obtaining such confidence interval is by applying the Central Limit Theorem of Statistics to approximate a binomial distribution by a normal distribution.

Definition 27 (α-Confidence Probabilities) *Let $\mathcal{DB}$ be an uncertain database. For a set of drawn database instances $\mathcal{S}$, and for a possible clustering $\mathcal{C}(w_i), w_i \in \mathcal{S}$, a distance threshold τ and a level of significance α, the probability*

$$\hat{P}(w_i, \tau, \alpha) = \hat{P}(w_i, \tau) - z \cdot \sqrt{\frac{1}{|\mathcal{S}|} \hat{P}(w_i, \tau) \left(1 - \hat{P}(w_i, \tau)\right)}, \tag{10.1}$$

is called α-confidence probability of τ-representative w_i, where z is the $100 \cdot (1-\alpha)$ percentile of the standard normal distribution.

The α-confidence probability $\hat{P}(w_i, \tau, \alpha)$ can be used to return the clustering $\mathcal{C}(w_i)$ as a τ-ϕ-representative clustering to the user, as it guarantees, that by a user specified level of confidence α, the true probability $P(w_i, \tau)$ is guaranteed to be larger than $\hat{P}(w_i, \tau, \alpha)$. To compute $\hat{P}(w_i, \tau, \alpha)$ as in Definition 27 we argue that in our setting the Central Limit Theorem is highly applicable, since the sample size $|\mathcal{S}|$ should be sufficiently large (≥ 30 as a rule of thumb [25]). Furthermore, the probability $P(w_i, \tau)$ should not be extremely small, since a cluster representative having an extremely small value of $P(w_i, \tau)$ is meaningless

and should not be returned to the user in the first place. In the case where all cluster representatives have an extremely small $P(w_i, \tau)$ value, the parameter τ should be increased to obtain meaningful representatives. Yet, we note that more accurate approximations can be obtained using Wilson Score Intervals [195] or using exact binomial confidence intervals [40]. The following definition summarizes the contribution of this section.

10.5.3 Selection of Representative Worlds

Using the techniques of Section 10.5.2 we can estimate, for a given τ the probability of any drawn possible world to be an τ-representative. In this section, we show how good representatives having a high confidence and low τ can be extracted automatically from a set of sampled worlds. Furthermore, when more than a single representative world is returned, a requirement is to minimize redundancy between sets of worlds represented by each representative [43, 91, 210]. This requirement is important in order to avoid overly similar clustering representatives, thus minimizing the information returned to the user. To solve this challenge, we propose a general approach to first derive a clustering of the set of clusterings PC that have been obtained by applying the domain specific clustering algorithm $\mathcal{C}$ to sampled possible worlds $\mathcal{S}$. Then, a single representative clustering R is chosen from each cluster of PC such that τ is minimized while the fraction of drawn possible clusterings is maximized. Formally:

Definition 28 (Representative Worlds Clustering) *Let PC denote the set of possible clusterings derived from sampled worlds $X = \{w_1, ..., w_n\}$. Let D be a $|\mathcal{S}| \times |\mathcal{S}|$ matrix such that*

$$D_{i,j} := dist(w_i, w_j)\,.$$

Furthermore, let $\mathcal{C}'$ be a metric clustering algorithm based on dist and let $\mathcal{C}'(PC)$ denote the meta-clustering returned by applying $\mathcal{C}'$ to the set of PC of possible clusters. For each meta-cluster $Y \in \mathcal{C}'(PC)$ a Representative Worlds Clustering *returns a triple $(R, \tau, \hat{P}(R, \tau, \alpha))$, where $R \in Y$ is the clustering chosen to represent Y, and R is an α-significant representative (cf. Definition 27) with a probability ϕ of at least $\hat{P}(R, \tau, \alpha)$.*

In Definition 28, two parameters are undefined, the choice of the clustering algorithm $\mathcal{C}'(PC)$ and heuristics to obtain a representative from each meta-cluster in $\mathcal{C}'(PC)$. For the choice of clustering algorithm $\mathcal{C}'$, we propose to use the PAM algorithm [108] and refer to numerous clustering surveys [89, 90, 111, 88, 110, 168] including metric algorithms.

For the problem of defining a representative for each a meta-cluster Y, we propose the following two heuristics. Our first heuristic requires all possible clusterings in a meta-cluster $Y \in \mathcal{C}'(PC)$ to be represented.

Definition 29 (Complete Representative) *For a meta-cluster $Y \in \mathcal{C}'(PC)$ the* complete representative *is the clustering*

$$R_{complete} := \arg\min_{R \in Y} \left(\max_{R' \in R} \left(dist(R, R') \right) \right)$$

which has the minimum maximum distance $\tau = \max_{R' \in R} (dist(R, R'))$ *to all others clusterings in* Y*.*

This representative $R_{complete}$ can be returned as a τ-ϕ-representative by computing a confidence probability $\phi = \hat{P}(R_{complete}, \tau, \alpha)$ using a user specified level of confidence α as described in Section 10.5.2.

A drawback of the complete representative approach, is that the value of τ may grow arbitrarily large, being at least half of the corresponding clusters diameter. An τ-representative having an overly large τ value, such as an ARI-distance [85] value greater than 0.2, may have no semantic meaning to the user, as the space of clusterings represented by this τ-representative grows too large to allow meaningful decision making. Furthermore, a large value of τ yields overlapping clusters. For instance, for a pair of complete representatives R_i and $R_j, i \neq j$, where R_i is an τ_i-representative and R_j is an τ_j representative, it may hold that for a single sampled clustering $w_k \in \mathcal{S}$ that $\mathcal{D}(w_k, R_i) \leq \tau_i$ and $\mathcal{D}(w_k, R_j) \leq \tau_j$. This drawback of complete representatives can be particularly bad, if the underlying clustering algorithm $\mathcal{C}$ allows clusters to have a large diameter (e.g., $\mathcal{C}$ is k-means). In contrast, complete representatives may yield good results in settings where density-based clustering algorithms such as DBSCAN are used.

For the general case, we propose a different approach, where a maximum threshold for τ is provided. This parameter, which is specific to the chosen distance function dist, should be chosen in a way that a user should treat two clusterings, having a dist distance of no more than τ as similar.

Definition 30 (τ_{max}-Clustering) *Given a* $\tau_{\max}$ *threshold, for a cluster* $C \in \mathcal{C}'(PC)$ *a* $\tau_{\max}$ representative *is an* τ-ϕ*-representative, such that* $\tau \leq \tau_{\max}$ *given by*

$$R_{\tau_{\max}} := \arg\max_{R \in Y} \left(\sum_{C_i \in Y} I\left(dist(R, C_i) < \tau_{\max}\right) \right) \cdot C_i.supp.$$

Again, this representative $R_{\tau_{\max}}$ can be returned as a τ-ϕ-representative by computing a confidence probability $\phi = \hat{P}(R_{\tau_{\max}}, \tau_{\max}, \alpha)$ using a user specified level of confidence α as described in Section 10.5.2.

The main drawback of τ_{max} clusterings, is that large fractions of possible clusterings may be assigned to no τ-representative. The semantics of such result, however, may be useful, indicating that a large fraction of possible clusterings deviate too much from other clusterings. This indication of high heterogeneity of possible clusterings has to be considered when making decisions based on the uncertain data set $\mathcal{DB}$.

10.6 Experiments

10.6.1 Data

Ground Truth The focus of this chapter is to mitigate the effect of uncertainty by obtaining an uncertain clustering that is similar to applying algorithm $\mathcal{C}$ on the real,

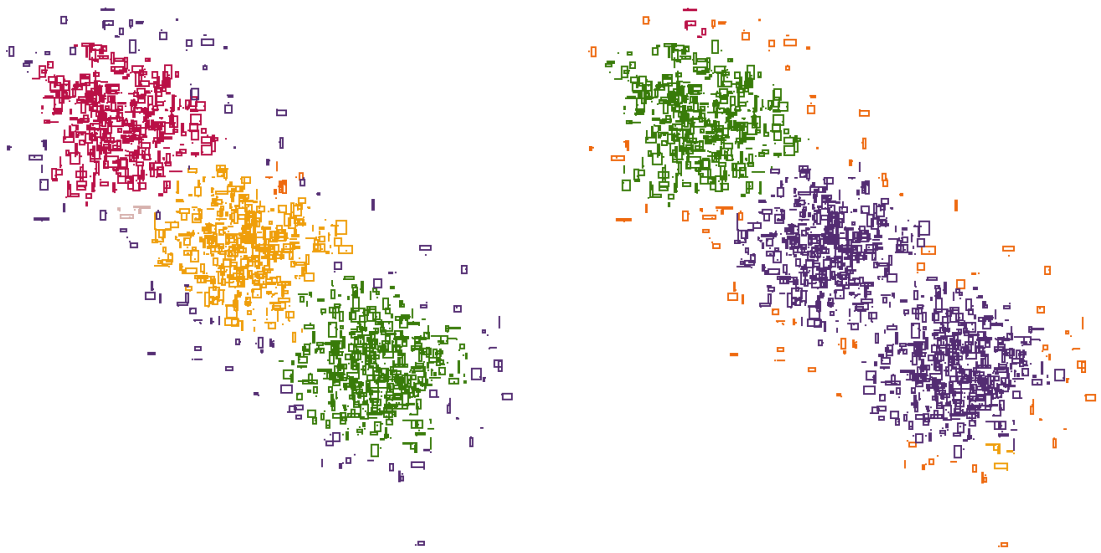
unknown data set oracle($\mathcal{DB}$). Since our methodology is independent of the choice of $\mathcal{C}$, we evaluate it using the following experimental approach. Using the certain datasets $\mathcal{DB}$ summarized in Table 10.1 (taken from the UCI Machine Learning Repository), we applied traditional (certain) clustering algorithms $\mathcal{C}$ (by default $\mathcal{C}$ is the DBSCAN algorithm), to obtain the ground-truth clustering $\mathcal{C}$(oracle($\mathcal{DB}$)). We tuned the clustering algorithm $\mathcal{C}$ (e.g., selected appropriate values for parameters ϵ and MinPts in the case of DBSCAN), in order to yield a high F-measure for predicting the class labels of each database object. Then, we discarded the class-information from the datasets, and treated the result of $\mathcal{C}$ as the ground truth $\mathcal{C}$(oracle($\mathcal{DB}$)). Recall that our goal is to compute clustering results on an uncertain version of each dataset similar to $\mathcal{C}$(oracle($\mathcal{DB}$)), independent of the quality of $\mathcal{C}$ in terms of its F-measure. Yet, the parameters of $\mathcal{C}$ should have meaningful values in our setting, to avoid effects such having only a single cluster or no clusters at all.

Uncertainty Generation In a uncertain setting we do not have access to the certain database oracle($\mathcal{DB}$) and are rather given uncertain database $\mathcal{DB}$. Thus for each object $o \in$ oracle($\mathcal{DB}$) we draw a new object using a multivariate Gaussian or multivariate uniform distribution. In both cases, we use a parameter ext to describe the uncertainty. In the Gaussian case we randomly chose a standard deviation $std \in [0; ext/4]$ in each dimension and generated a center of a generating probability distribution by drawing one sample point g from the Gaussian pdf with $\mu = o$ and $\sigma = std^2$. Using g as observation of o, we generate $i - 1$ additional points from the normal distribution $\mu = g$ and $\sigma = std^2$. The resulting i points correspond to samples of an uncertain object observed at location g.

In case of uniform distribution, a rectangle r was constructed having an extent chosen uniformly in the interval $[0, ext]$ in each dimension. The resulting new object u is chosen uniformly from this interval. Then, the rectangle r is centered at u and $i - 1$ more points are drawn uniformly from r. In addition to o, which is guaranteed to be inside r by construction we generated $i - 1$ additional points uniformly distributed in r. All generated uncertain objects form the uncertain database $\mathcal{DB}$.

10.6.2 Experiments on Synthetic Data

Before evaluating the proposed approach in a broad experimental setting, we first demonstrate the difference regarding the result of the clustering task between our technique and previous work on clustering uncertain objects (cf. Section 10.2). For this purpose, we generated a toy example consisting of three Gaussian distributed point clouds $\{(A, B, C\}$ which represent our ground truth data. After adding Gaussian uncertainty, as described in the previous section, all objects consist of several sample points which can be covered by a minimum bounding rectangle; these rectangles are shown Figures 10.3 and 10.4. Figure 10.3(a) illustrates the clustering of the original data set without uncertainty. Objects belonging to the same cluster are plotted using the same color. Outliers are plotted in a different color. In this experiment, we used DBSCAN for clustering. Figure 10.3(b) shows the result of a competitor approach: *Median Clustering* (MC) performs DBSCAN on the uncertain objects by reducing each uncertain object to one single possible alternative which corresponds to the median of its alternatives. Observe that MC yields a different cluster-



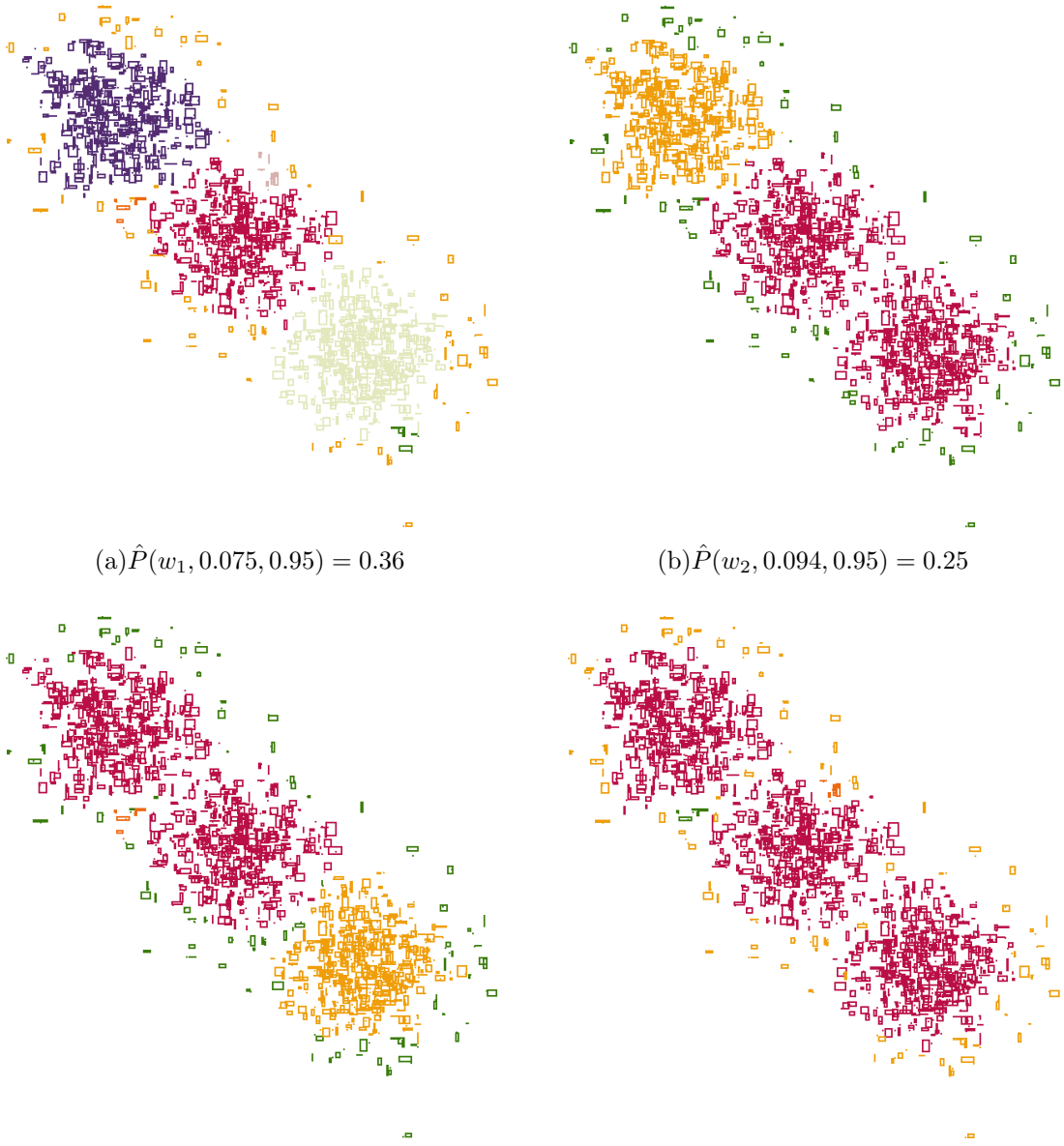
(a) Clustering of original data (b) Clustering of medians

Figure 10.3: Clustering results of sample dataset

ing compared to the original one, since the lower two point clouds are merged to a single cluster.

Next consider the results of our approach when generating four representative clusterings $w_1, ..., w_4\}$ in Figure 10.4 using $|\mathcal{S}| = 100$ samples (i.e., 100 database instances). First, note that the four results coarsely reflect the four expected possible results of a density based clustering aproach ({A}, {B}, {C}), ({A, B}, {C}), ({A}, {B, C}) and ({A, B, C}). The corresponding confidence probabilities $\hat{P}(w_i, \tau, \alpha)$ (cf. Definition 27) are shown for each representative. For instance, representative w_1, shown in Figure 10.4(a), is an $\alpha = 0.95$-significant representative having a probability of 0.38 to have an ARI-distance of at most $\tau = 0.075$ to the ground-truth clustering $\mathcal{C}(\text{oracle}(\mathcal{DB}))$.

The real ARI-distances of the four representatives to $\mathcal{C}(\text{oracle}(\mathcal{DB}))$ are 0.038, 0.400, 0.404 and 0.851 respectively. In this toy example, the clustering with the smallest ARI to the base clustering has the highest probability. This is not always the case and our approach might also return a result having a large distance with the highest probability. Yet, our approach usually returns at least one possible clustering having a very high similarity with the true clustering. However, more importantly, unlike existing approaches such as the method tested in Figure 10.3(b), our approach is able to assess the probabilistic quality of its results and can provide multiple representative clusterings for the user to choose from.



(a) $\hat{P}(w_1, 0.075, 0.95) = 0.36$

(b) $\hat{P}(w_2, 0.094, 0.95) = 0.25$

(c) $\hat{P}(w_3, 0.075, 0.95) = 0.19$

(d) $\hat{P}(w_4, 0.22, 0.95) = 0.12$

Figure 10.4: 4-median clustering representatives

Dataset	Size	Dimension
ABALONE	4177	8
ECOLI	336	8
GLASS	214	10
IRIS	150	4
SEGMENTATION	2310	19
WINE	178	13
YEAST	1484	8
D31	3100	2

Table 10.1: Datasets

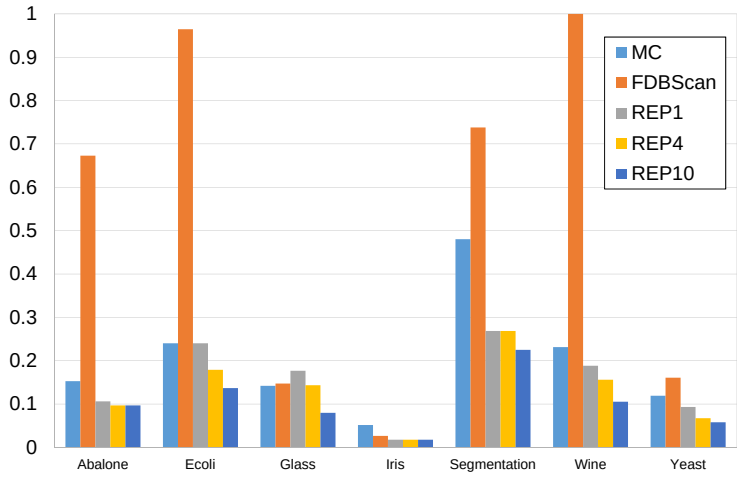


Figure 10.5: ARI-distance on all datasets.

10.6.3 Experiments on Real Data

We now evaluate our proposed method in a broader setting. We compare our approach to FDBSCAN [114] and the approach which performs DBSCAN using the medoid of each object in the database (referred to as MC in Section 10.6.2). Like previous studies on uncertain data clustering (e.g., [73]) we used datasets with different characteristics from the UCI Machine Learning Repository (cf. Table 10.1). For comparison purposes, we normalized each of the datasets to [0,1] in each dimension. Unless mentioned otherwise, we use $\mathcal{C}$=DBSCAN and choose $\epsilon = 0.08, MinPts = 5, extent = 0.04, i = 10$ and for our approach we use $|\mathcal{S}| = 100$ samples.

Clustering Quality

Figure 10.5 illustrates the results under the default values on all tested datasets. Shown are the ARI-distances (1-ARI value) to the certain clustering on the original dataset. Thus, a

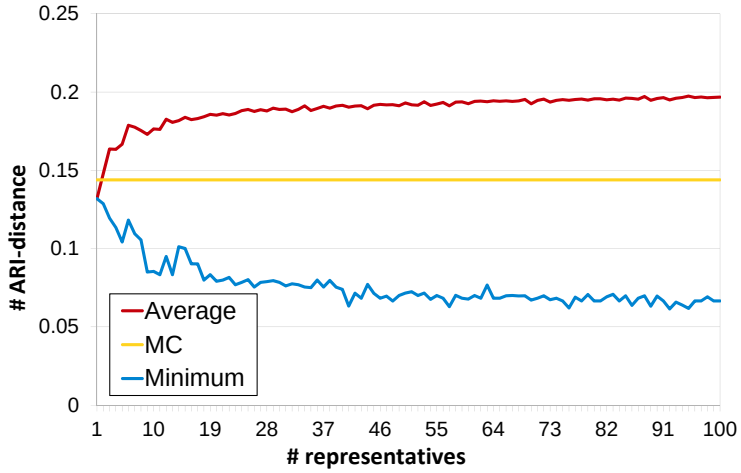


Figure 10.6: ARI-distance vs. the number of representatives.

value of zero means that the method produces the same clustering result on the uncertain data than on the certain data, whereas a value closer to 1 means that the two clusterings differ drastically. As observed, most of the times the rather simple comparison partner (MC) of clustering the medians of the objects performs better than the more sophisticated FDBSCAN. This might be because of the shortcomings of FDBSCAN regarding the consideration of possible worlds semantics as discussed in Section 10.2. Although the MC approach returns a clustering of a possible world, it still cannot assign any measure of confidence to it. Thus the result may be a very unlikely world. This becomes obvious when revisiting the results in Figure 10.5. Even for the case were only one representative is returned by our method (REP1), this representative resembles the original clustering better and in addition it also carries a confidence about its similarity to the true clustering. For instances of our method with multiple representatives, the figure shows the ARI-distance of the representative with the minimum distance. Thus, increasing the number of representatives (REP4, REP10) ensures that at least one representative resembles the original clustering very closely.

Number of Representatives

An important question is how many representatives should be presented to the user. Presenting the user to few representatives may yield an insufficient understanding of the possible outcomes and the result may not contain a clustering which is close to the "true" clustering at all. Presenting too many representatives may overwhelm the user. In Figure 10.6, we show the averaged ARI-distance over all considered datasets when increasing the number of representatives. Observe that the average ARI-distance of all our representatives (weighted by the confidence of the representatives) increases in comparison to the MC approach (we exclude the FDBSAN result in this graph due to its large ARI-distance). This can be explained by the diversity that increases with increasing number of

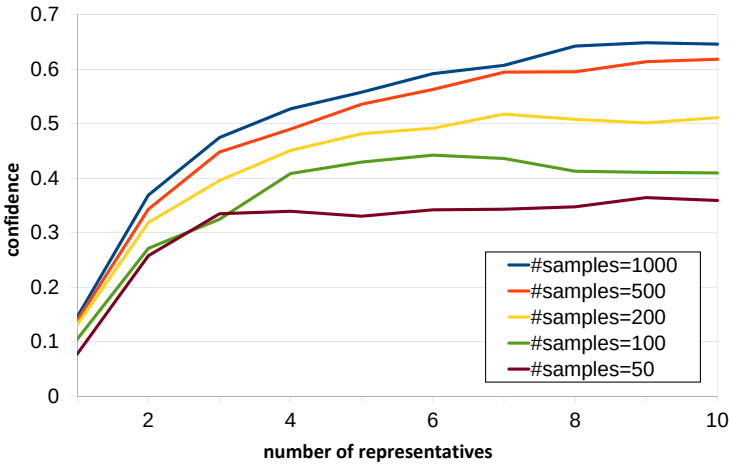


Figure 10.7: Confidence depending on samples.

representatives. On the other hand, the distance of the closest representative to the original clustering decreases when increasing the number of representatives, as a larger number of representatives increases the probability of having a representative close to the ground truth clustering. Summarizing, our experiments show that the quality of our returned representatives not improve significantly when returning more than ten representatives. Also, our experiments have shown that a set of four cluster representatives yields fair results in most cases, while it can still be considered as concise enough to be represented to an average user.

Number of Samples

In our next experiment, we investigate how many samples $|\mathcal{S}|$ are required in order to obtain significant results on the D31 data set (cf. Table 10.1). For this purpose, we aggregated the $\hat{P}(w_i, \tau, \alpha)$ of all cluster representatives w_i for $\tau = 0.1$, $\alpha = 0.95$ for different values of $|\mathcal{S}|$. The result is shown in Figure 10.7, where it can be observed that a larger sample size $|\mathcal{S}|$ increases the lower probability bounds obtained by Definition 27. More information on obtaining confidence intervals for a binomial probability function such as $P(w_i, \tau)$ can be found in the literature [80].

Runtime

All experiments were performed on a 64-Bit Linux Notebook with Intel Core i7 quadcore with 2.10GHz and 8GB RAM. All algorithms were obtained from or implemented into the ELKI-Framework [3] which is written in Java. The runtime of our approach directly corresponds to the number of samples we utilize. Thus our approach will always be slower in terms of runtime than other approaches like MC for the exchange of more valuable information and insights into the dataset, which is normal for data mining tasks such as clustering. Thus we present which modules of the process effect runtime the most. For

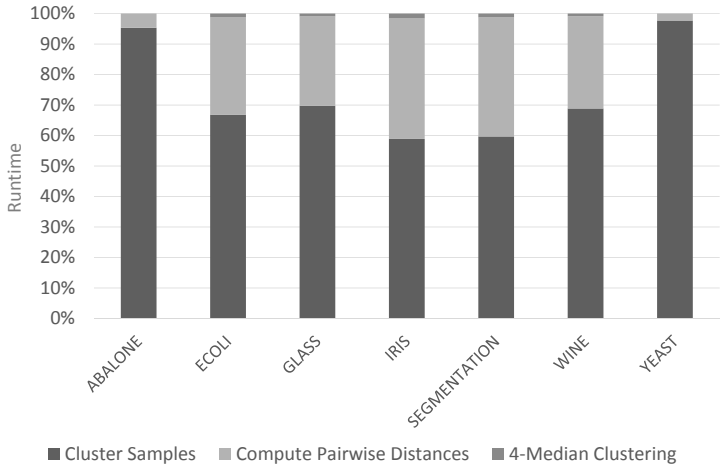


Figure 10.8: Relative Runtime.

a setting having a $|\mathcal{S}| = 100$ samples, using $\mathcal{C}$=DBSCAN as clustering algorithm having default parameters $\epsilon = 0.08, MinPts = 5$ on the D31 data set, we divided the procedure of finding cluster representatives into three steps:

- Cluster Samples: This step includes drawing possible instances of the database and clustering them using DBSCAN. This process is repeated until $|\mathcal{S}| = 100$ instances have been processed.
- Compute Pairwise Distances: During the last step, ARI-Distances between clustered database instances is performed several times, thus we precomputed the $\frac{100 \cdot 100}{2}$ pairwise distances.
- 4-Median Clustering: In this step, we perform k-Medoid Clustering.

The computational bottleneck of the overall procedure comes from the first two steps. Obviously, applying $\mathcal{C}$ a large number of times is computationally expensive. The second step of computing the pairwise distances of the clustering results usually takes less time. This step strongly depends on the characteristics of the outcome from the first step. Specifically, computing the ARI-distance between two clusterings becomes more expensive if the clusterings contain more clusters in them. ARI is based on the precomputation of the cluster contingency table which counts the number of objects which are present in each pair of clusters of two clusterings. The number of clusters in a clustering is of course dependent on the dataset and the parameter settings.

10.7 Conclusions

In this work, we present a general solution to cluster uncertain objects. Our challenge was to develop a framework making any clustering that has been developed for certain

data applicable for the case of uncertain data. We approach this challenge by employing a sampling approach to obtain a number of possible database instances from the uncertain database. Applying a domain specific clustering algorithm to each obtained database instance yields a possibly large set of different clusterings. The new challenge is thus how to find a consensus between all these possible clusterings. For this purpose, we define the notion of representative τ-ϕ-clusterings: a representative τ-ϕ-clustering is a clustering having probability at least ϕ to have a distance of at most τ to the actual clustering of the data if the data were certain. Our solution follows a sampling approach, which returns cluster representatives that are guaranteed to be τ-ϕ-clusterings at a user specified level of significance. To the best of our knowledge, our approach is the first to yield clusterings associated with confidence, allowing the user to assess the quality of the clustering result. Furthermore, by returning multiple representative clusterings to the user, we can improve the quality (and therefore usefulness) of results, as shown by our experimental study.

Chapter 11

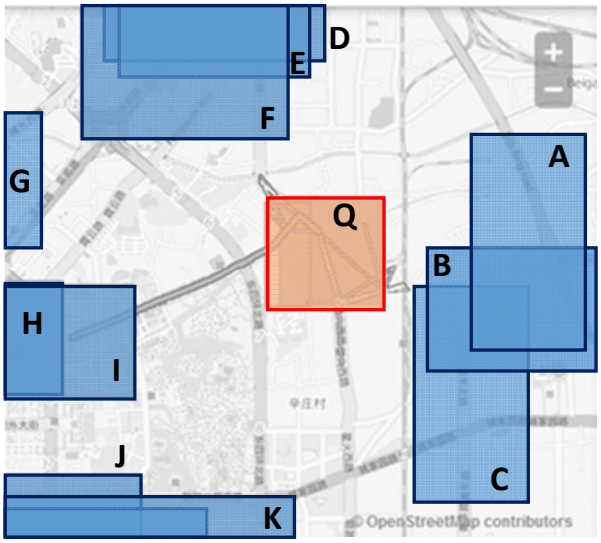
Approximate Probabilistic Representative Spatial Query Processing

11.1 Introduction

In Chapter 10, techniques for mining uncertain data using sampling techniques and representative answers were presented. With some modifications, these techniques can also be applied to the problem of answering queries on uncertain data.

This chapter targets the problem of efficiently deriving query results and their confidence values from an uncertain dataset. Therefore, our approach builds upon existing solutions (and their index structures) for certain data. Rather than returning exact result probabilities which conforms to all possible worlds, we propose to sample a subset of possible worlds. On each sampled world, we apply a traditional query. Then, the main contribution of this work is to unify the set of sampled query results into a smaller set of *representative query results*. A representative query result is a query result, for which we can guarantee, at a specified level of significance, that the true (but unknown) query result must be similar to this result. For an overview of the uncertain query workflow recall Section 8.5.

We argue that any solution that simply returns a set of possible worlds, enriched by likelihood values of this worlds, will lose significance in large databases. It has been shown in [44] that query processing on uncertain data is #P-hard, as the number of possible results grows exponential in the number of objects. Consequently, the average probability of each possible result grows exponentially small, as the probability mass of one is distributed among all possible results. To illustrate this problem, Figure 11.1 shows regions of possible positions of taxi-cabs on a road network in the city of Beijing on Februar 4th, 2008 at noon using the T-drive data set [205, 204]. Uncertainty of taxi-cabs is the result of low GPS-signal frequency: some of the taxi's most recent GPS updated is 30 to 90 seconds old. Rectangular uncertainty regions are derived using the techniques of [137]. Assuming



RID	Result Set	Confidence
R1	{A,B,C,E,F}	0.00132
R2	{A,B,C,D,F}	0.00127
R3	{A,C,I,H,K}	0.00123
R4	{B,D,F,G,H}	0.00126
...	...	...
...	...	...
R792	{C,G,H,I,K}	0.00121

Figure 11.1: Probabilistic 5NN query example: (left) exemplary uncertain database with a snapshot of possible locations of taxi cabs in Beijing, (right) set of possible query results with confidences.

we wish to perform a five-nearest neighbor (5NN) query using the uncertain query object Q (pink region). For instance, if 12 objects (blue regions) have a non-zero chance to be a 5NN of q, then up to $\binom{12}{5} = 792$ possible results may have a non-zero chance to be the true result as depicted in the table on the right side in Figure 11.1. It is obvious to see that the set of possible 5NN query results can grow extremely large, depending on the number of involved objects. Enumerating and reporting all these possible results is not viable in terms of computational cost and usability.

Therefore, we propose a different approach. Instead of returning all possible worlds along with their probability to be the (unknown) ground-truth, we aim to return a smaller and, thus, more appropriate set of *representative query results*, which are associated with the probability of being sufficiently similar (but not necessarily equal) to the (unknown) ground-truth. Exemplary, a representative query result for the example in Figure 11.1 could be: *"With a probability of 60%, the (unknown) true result contains at least three out of the following result* $\{A, B, C, F, E\}$*, and with a probability of 30% contains at least four objects out of the result* $\{A, B, C, I, K\}$*"*.

Analogous to Chapter 10, we aim at allowing the application of *any* existing query processing method $\mathcal{Q}$ on an uncertain database $\mathcal{DB}$, based on *any* given uncertain data model for $\mathcal{DB}$, assuming that $\mathcal{Q}$ would be appropriate for $\mathcal{DB}$, if $\mathcal{DB}$was certain. This approach performs sampling and then represents the original uncertain database $\mathcal{DB}$as a set of sample deterministic databases. We then run the query algorithm $\mathcal{Q}$ on each of the sampled databases, to derive a set of possible results PR. Our main contribution is to combine the resulting set PR into a concise set RPQ of *representative results*. Furthermore, using all query results in PR, we estimate a probability ϕ of each representative

result in RPQ, defined as the probability that this representative result does not exceed some maximum distance τ to the true result. Since ϕ has to be estimated from sampled databases, we obtain a lower bound of ϕ that is significant at a user specified level. We provide a methodology to derive an RPQ for a given value of τ; and therefore impose quality constraints on the uncertain query results.

11.2 Problem Definition

This section gives general definitions for querying uncertain databases, generalizing uncertainty models and query predicates. While existing works assume that objects are points in a multi-dimensional feature space [37, 34, 86, 35, 20, 41, 174, 202, 122, 17], or assume that data is represented as a relational database [84, 23, 4, 189], instead, we allow objects to have any abstract type $\mathcal{O}$, where $\mathcal{O}$ is an object space, such as a multi-dimensional feature space, the set of all strings, the set of all images, etc. Generalizing the state-of-the-art models to express uncertainty, each *uncertain object* is represented by a (possibly infinite) set of values in $\mathcal{O}$, each associated with a non-zero probability.

Next we provide a general definition of a query on uncertain data. Since our solutions aim at solving general queries for uncertain data, our definition of a query is abstract rather than defining a specific query or a set thereof. The only assumption we make to a query is that it returns a set of result objects, and that the corresponding certain query (on certain data) has a solution. A general query on certain data is defined as follows

Definition 31 (Query (Certain data))
Let $\mathcal{DB} \subseteq \mathcal{O}$ be a certain database. A query Q is a function

$$\mathcal{DB} \rightarrow \mathcal{DB}^*$$

which maps a database $\mathcal{DB} \subseteq \mathcal{O}$ to a set of results objects $Q(\mathcal{DB}) \subseteq \mathcal{DB}$.

Following possible world semantics, a query on uncertain data maps each possible result set to its probability to be the true (but unknown) query result.

Definition 32 (Query (Uncertain data))
Let $\mathcal{DB} = \{O_1, ..., O_N\}$ be an uncertain database. A query Q is a function

$$\mathcal{DB} \rightarrow (\mathcal{DB} \rightarrow [0, 1])*$$

which maps an uncertain database $\mathcal{DB}$ to a set $Q(\mathcal{DB})$ of query result sets, each associated with the probability $P(q \in Q(\mathcal{DB}))$ of being the true query result.

Since again the result set $Q(\mathcal{DB})$ may grow exponentially in the number of database objects, the application of a Monte-Carlo approach is advised.

11.3 Querying Sampled Worlds

Let $\mathcal{DB}$ be an uncertain database and let $\mathcal{Q}$ be a query. Let $\mathcal{S} = \{w_1, ..., w_{|\mathcal{S}|}\}$ be a multiset[1] of possible worlds of $\mathcal{DB}$ generated from $\mathcal{DB}$ using generation rule $\mathcal{G}$ and let $\mathcal{Q}(\mathcal{S})$ denote the multiset of query results obtained by applying query $\mathcal{Q}$ to each sample world in $\mathcal{S}$. We denote the set of distinct query results in $\mathcal{Q}(\mathcal{S})$ as the set PR of *possible results* obtained from sample $\mathcal{S}$. For any result r in PR, the *support* $r.support$ of r is defined as $\sum_{i=1}^{|\mathcal{S}|} I(\mathcal{Q}(w_i) = r)$, where $I(\cdot)$ is an indicator function that returns 1 if its operand is true and 0, otherwise. Simply speaking, the *support* is the number of occurrences of the result set r in the multiset $\mathcal{Q}(\mathcal{S})$.

As shown in Lemma 7, the probability $\hat{P}(r) = \frac{r.support}{|\mathcal{S}|}$ is an unbiased estimator of the probability $P(r)$ that r is the true query result $\mathcal{Q}(\text{oracle}(\mathcal{DB}))$ of $\mathcal{DB}$, i.e., $E(\hat{P}(r)) = P(r)$.

11.4 Representative Query Results

Not only the number of possible results of a query is huge, we also saw in the example in Figure 11.1, that different positions of the query object may yield a significantly different result set. To reduce the number of results, this section introduces a general concept for determining representative results. Analogous to Chapter 10, dist denotes a distance measure between results. We will discuss possible *result distance functions* later in Section 11.6.2. Next, we define a representative query result as follows:

Definition 33 (Representative Result)
Let $\mathcal{DB}$ be an uncertain database and let $\mathcal{Q}$ be a query. We call a query result $\mathcal{Q}(w_i)$ a τ-ϕ-representative result, if the probability

$$P(w_i, \tau) := P(dist(\mathcal{Q}(w_i), \mathcal{Q}(\text{oracle}(\mathcal{DB}))) \leq \tau)$$

that the true (but unknown) result $\mathcal{Q}(\text{oracle}(\mathcal{DB}))$ of $\mathcal{DB}$ has a distance $dist(\mathcal{Q}(w_i), \mathcal{C}(\text{oracle}(\mathcal{DB})))$ of at most τ is at least ϕ.

We are able to approximate the probability $P(w_i, \tau)$ that the true result has at most a disatance dist of τ by using sample results from $\mathcal{DB}$, as shown in Lemma 8.

Albeit unbiased, the probability $\hat{P}(w_i, \tau)$ cannot be used directly to assess the probability $P(w_i, \tau)$ of result set w_i having a distance of at most τ to the true result $\mathcal{Q}(\text{oracle}(\mathcal{DB}))$. Thus, w_i can not simply be returned as a τ-$\phi = \hat{P}(w_i, \tau)$-representative according to Definition 33, because the estimator $\hat{P}(w_i, \tau)$ may overestimate the true probability $P(w_i, \tau)$. Again, we will aim to find a lower bound $\hat{P}(w_i, \tau, \alpha)$ such that we can guarantee $P(w_i, \tau) \geq \hat{P}(w_i, \tau, \alpha)$ with a probability of α, where α is a domain specific level of significance (typically, $\alpha = 0.95$).

We use an α-confidence probability (cf. Definition 27) to return the result $\mathcal{Q}(w_i)$ as a τ-ϕ-representative query result to the user, as it guarantees that by a user specified level

[1] Due to independent sampling, the same sample may be drawn multiple times.

of confidence α, the true probability $P(w_i, \tau)$ is guaranteed to be larger than $\hat{P}(w_i, \tau, \alpha)$. Furthermore, the probability $P(w_i, \tau)$ should not be extremely small, since a result representative having an extremely small value of $P(w_i, \tau)$ is meaningless and should not be returned to the user in the first place. In the case where all result representatives have an extremely small $P(w_i, \tau)$ value, the parameter τ should be increased to obtain meaningful representatives. Yet, we note that more accurate approximations can be obtained using Wilson Score Intervals [195] or using exact binomial confidence intervals [40].

11.5 Selection of Representative Results

Using the techniques of Section 11.4 we can estimate, for a given τ, the probability of any drawn possible world to be a τ-representative. In this section, we show how good representatives, i.e., possible results with a high confidence of having a low distance to a large fraction of possible worlds, can be extracted automatically from a set of sampled worlds. Furthermore, when more than a single representative world is returned, a requirement is to minimize redundancy between sets of worlds represented by each representative [43, 91]. This requirement is important in order to avoid overly similar result representatives. To solve this challenge, we propose two approaches. The first approach requires the user to specify a maximum τ_{max} value, to denote the maximum distance the result may have to the ground-truth to be considered good enough. For instance, for a 10-NN query, a user may wish to guarantee having at least eight out of ten objects ground-truth objects returned. Given this τ_{max} parameter, we can formulate the problem of finding the best $\tau_{max} - \hat{P}$-representatives as a maximum coverage problem on the sampled results. For this NP-hard problem, we employ a Greedy-approximation [79].

Our second, parameter free approach first derives a clustering of the set of sampled possible results PR that have been obtained by applying the any domain specific query algorithm $\mathcal{Q}$ to sampled possible worlds X. Then, a single representative result $\mathcal{R}$ is chosen from each cluster of PR such that τ is minimized while the fraction of covered drawn possible results is maximized.

11.5.1 Maximum Representative Cover

Our first approach allows a user to specify the number n of representatives as well as the maximum distance τ of a representative to the (unknown) ground truth. Thus, this approach selects representative results such that the fraction of possible worlds having a τ-similar representative is maximized.

Definition 34 (Maximum Cover Representatives) *Let PR denote the set of possible query results derived from sampled worlds $X = w_1, ..., w_n\}$, let τ be a distance threshold, and the let n be a positive integer. The set of maximum cover representatives $\mathcal{R}_{max}(\tau, n) \subseteq PR$ is the set of size n that maximizes the number of possible results $r \in PR$ having a least one representative $R \in \mathcal{R}_{max}(\tau, n)$ having a distance of at most τ to r. Formally:*

Algorithm 7: Maximum Cover Representatives

Require: Level of Significance α, Sampled Possible Results PR
Require: Similarity Threshold τ, Nr. of Representatives n

result = emptyset
for $i \in 1 : n$ **do**
  $\text{best}_r = null$
  $\text{best}_{\text{count}} = 0$
  for $r \in PR$ **do**
    covercount = 0
    for $r' \in PR$ **do**
      if $dist(r, r') < \tau$ **then**
        covercount = covercount+1
      end if
    end for
    if covercount $>$ $\text{best}_{\text{count}}$ **then**
      $\text{best}_r = r$
    end if
  end for
  result = result $\cap$ $(\text{best}_r, \tau, \hat{P}(w_i, \tau, \alpha))$
end for
return result

$$\mathcal{R}_{max}(\tau, n) = \underset{\mathcal{R} \subseteq PR, |R| \leq n}{\arg\max} \sum_{r \in PR} I(dist(r, \mathcal{R}) \leq \tau),$$

where $I(\varphi)$ is an indicator function that returns 1 if predicate φ is true and 0 otherwise, and $dist(r, \mathcal{R})$ is the minimum result distance between result r and any result in $\mathcal{R}$.

To find maximum cover representatives as defined in Definition 34 an exact solution is not viable, due to the following theoretic result.

Lemma 9 *The problem of computing $\mathcal{R}_{max}(\tau, n)$ is np-hard.*

Proof 9 *By reduction from the set covering problem, which is one of Karp's 21 NP-complete problems [106]. Given a set of elements U and a set S of n sets whose union equals U, the set cover problem is to identify the smallest subset of S whose union equals the universe. For reduction to the problem of finding maximum cover representatives, map the set $U = \{u_1, u_2, ..., u_m\}$ to $PR = \{r_1, ..., r_2, ..., r_m\}$, map each set $s \in S$ to a representative $r \in PR$ such that any other possible result $r' \in PR$ has a distance $dist(r, r_i) \leq \tau$ if and only if for corresponding element $u_i \in U$ it holds that $u_i \in s$. Now, set the parameter n of the maximum cover representative problem to the smallest number of representatives able to cover all results in PR. If the maximum cover representatives problem can be solved in polynomial time, then any set covering problem can be solved in polynomial time, following the reduction above. This leads to a contradiction assuming $P! = NP$. Thus, unless $P = NP$, the maximum cover representative problem in np-hard.*

To obtain an approximate result, we employ a simple Greedy approach, which iteratively selects the representative which covers the largest fraction of PR. Following the theoretic results of [79], this approximation guarantees to cover $1 - \frac{1}{e} \approx 0.632$ of the optimal cover. The algorithm of this Greedy approach is shown in Algorithm 7. This algorithm fetches one representative in each iteration of the loop starting in Line 2. In each iteration, Lines 5 - 15 find the possible result having the largest fraction of τ-similar results in PR. This result is returned as a $\tau - \hat{P}-$representative in Line 16, where ϕ is set to an α-significant support value according to Definition 27.

While the maximum representative cover returns useful and intuitive result representatives, the main drawback is the need for the user to specify the parameters n and τ. The next section presents a different approach, which allows to find a proper number of representatives, as well as the individual similarity τ to the ground-truth adaptively for any data set.

11.5.2 Possible Result Clustering

The approach presented in Section 11.5.1 requires the user to specify the data set specific parameter τ. If this parameter is chosen too small, then the minimal confidence ϕ of the returned representatives may be extremely small. If τ is chosen too large, only a single central representative may be returned, thus loosing information about groups of mutually

Algorithm 8: Possible Result Clustering

Require: Level of Significance α, Sampled Possible Results PR
Require: Metric Clustering Algorithm $\mathcal{C}$
1: result = emptyset
2: **for** Cluster $C \in \mathcal{C}(PR)$ **do**
3: $c = \text{median}(C)$
4: $\tau = maxDist(c, C)$
5: $\phi = \hat{P}(w_i, \tau, \alpha$
6: result $= result \cup (c, \tau, \phi)$
7: **end for**
8: **return** result

similar possible results. Our second approach drops the requirement of providing any parameter. This approach finds representative results by clustering the space of sampled results using dist and representing each cluster by a single representative result. Formally:

Definition 35 (Representative Result Clustering) *Let $\mathcal{C}$ be a metric clustering algorithm and let $\mathcal{C}(PR)$ denote the clustering returned by applying $\mathcal{C}$ to the set PR of possible results using dist as distance function. For each cluster $C \in \mathcal{C}(PR)$, a τ-ϕ-representative result $R \in C$ is obtained. R is an α-significant representative (cf. Definition 27) with a probability ϕ of at least $\hat{P}(R, \tau, \alpha)$*

In Definition 35, two parameters are undefined, the choice of the clustering algorithm $\mathcal{C}$ and a heuristic to obtain a representative from each cluster in $C \in \mathcal{C}(PR)$. For the choice of clustering algorithm $\mathcal{C}$, we propose to use the PAM algorithm [107], which is appropriate for general metric spaces [88] but any clustering algorithm which is appropriate for general metric spaces can be used [107, 90, 110]. For the problem of defining a representative R for each a cluster C, we propose to use the result that is most representative for all possible results in a cluster $C \in \mathcal{C}(PR)$.

Definition 36 (Cluster Representative)
For a cluster $C \in \mathcal{C}(PR)$, the complete representative *is the result*

$$R_{complete} := \arg\min_{R \in C} \left(\max_{R' \in C} \left(dist(R, R') \right) \right)$$

which has the minimum maximum distance

$$\tau = \max_{R' \in C} \left(dist(R, R') \right)$$

to all other results in C.

This representative R can be returned as a τ-ϕ-representative with confidence probability of $\phi = \hat{P}(R, \tau, \alpha)$ using a user specified level of confidence α as described in Section 11.4.

11.6 Experimental Evaluation

The main intent of this chapter is to mitigate the effect of uncertainty by obtaining query results that are similar to the result obtained by applying a query $\mathcal{Q}$ on the real, unknown data set oracle($\mathcal{DB}$). Thus, the main focus of this experimental study is to show the quality of our representative query results, in terms of guaranteed similarity of the ground-truth. In addition, we show how existing state-of-the-art solutions fail to consistently provide such high quality query results.

11.6.1 Data

We use two real-world data sets for our experimental evaluation. The first data set is a traditional, certain data set to which we add artificial uncertainty. The second data set has inherent uncertainty.

T-Drive: The first real data set *T-Drive* uses taxi cab positions of the city of Beijing using the T-drive data set [205, 204]. To obtain a larger data set, we project all taxi-cab trajectories to the same day. Then, for all taxi-cabs having a recent (less than one minute old) GPS signal at time 12:00 noon, we use this current position as ground-truth oracle $\mathcal{DB}$. While the resulting 4839 two dimensional (longitude, latitude)-pairs are real data, we use a synthetic way of adding uncertainty to this dataset. This approach allows to scale the degree of uncertainty of the dataset, as well the the degree of stochastic dependency between uncertain objects: In an uncertain setting we do not have access to the certain database $oracle(\mathcal{DB})$ and are rather given uncertain database $\mathcal{DB}$. Thus for each object $o \in \text{oracle}(DB)$ we draw a new object $mask(o)$ using a multivariate uniform distribution. This unform distribution is defined on a rectangular area $r(o)$ having an extent chosen uniformly in the interval $[0; ext]$ in each dimension, where ext is a parameter used to control the uncertainty of the dataset. Then, rectangle $r(o)$ is centered at $mask(o)$, and object o is described by a uniform distribution on $r(o)$. Per default, we choose $ext = 0.01$.

IIP: The first real data set *IIP* is the 2014 International Ice Patrol (IIP) Iceberg Sighting Dataset[2] which has been used in prior work on querying uncertain data [122, 95, 83]. This dataset contains a set of iceberg sighting records, each of which contains the location (latitude, longitude) of an iceberg. We use each of the these data points as the ground-truth. To obtain uncertainty, each sighting record is also associated with a confidence-level attribute according to the source of sighting: R/V (radar and visual), VIS (visual only), RAD (radar only), SAT-LOW (low earth orbit satellite), SAT-MED (medium earth orbit satellite), SAT-HIGH (high earth orbit satellite), and EST (estimated). To reflect this uncertainty, each ground-truth Iceberg sighting is masked by a Gaussian error, having a deviation σ of $1km$, $2km$, $3km$, $4km$, $5km$, $6km$ and $5km$ respectively. For each true iceberg position $oracle(i)$, a single masked position $mask(i)$ is stored in the database and considered an (uncertain) observation of an iceberg. This masking is to ensure that the ground-truth is not necessarily equal to mean of the corresponding Gaussian. The

[2] http://nsidc.org/data/g00807.html

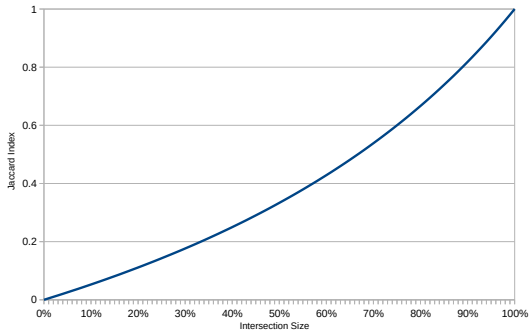


Figure 11.2: Jaccard Indices for relative size of set intersection.

uncertain position of an iceberg i is described by a Gaussian having $mask(i)$ as it's mean and σ as its deviation. The 2014 IIP dataset has a total of 17,139 uncertain ice berg positions.

11.6.2 Evaluation Metrics

For our effectiveness experiments, we evaluate the similarity of the results of all approaches to the ground-truth result. To assess similarity between result sets, we employ Jaccard similarity [87] as follows:

Definition 37 (Jaccard Similarity) *Let $Q(\mathcal{DB})$ be the result set of applying a query Q to an uncertain database $\mathcal{DB}$ and let* oracle$(\mathcal{DB})$ *be the result of applying Q to the ground-truth data* oracle$(\mathcal{DB})$*. We define the quality $J(Q(\mathcal{DB}))$ of the result $Q(\mathcal{DB})$ using Jaccard similarity (JS) as follows*

$$JS(Q(\mathcal{DB})) = \frac{Q(\mathcal{DB}) \cap \text{oracle}(\mathcal{DB})}{Q(\mathcal{DB}) \cup Q(\text{oracle}(\mathcal{DB}))}.$$

Analogously, we define define the Jaccard distance $JD(Q(\mathcal{DB}) = 1 - JS(\mathcal{DB}))$.

A Jaccard-similarity of 1 implies that the result $Q(\mathcal{DB})$ is exactly equal to the ground-truth result $Q(\text{oracle}(\mathcal{DB}))$, while a Jaccard-similarity of 0 implies that both sets are disjunctive. To gain an intuition, Figure 11.2 displays the correlation between the size of set intersection and resulting Jaccard Index. Given two sets the size of 100 objects each, the graph depicts the Jaccard Index for different intersection sizes from 0% (i.e., no intersection) to 100% (i.e., both sets are congruent).

11.6.3 Algorithms

In the remainder of this evaluation, our approach based on finding a Maximum Representative Cover (c.f. Section 11.5.1) will be denoted as *COVER*. Per default, the distance τ required by this approach is set to a Jaccard distance of 0.1. Our approach based on

Possible Result Clustering will be denoted as *PRC*. As a metric clustering approach this approach uses the PAM algorithm [107]. If note denoted otherwise, we return ten representative results and we use 1000 database samples. Any approach computing individual object probabilities, rather than possible result probabilities will be denoted as Individual Object Based (IOB), regardless of the query predicate. Note that there exists a variety of different algorithms to compute individual objects probabilities for various query predicates. However, for the same query predicate all these solutions compute the same result, and only differ in their run-times, such that we generalize these approaches for our effectiveness evaluation. For our run-time experiments, we implemented the individual object based solutions of

- a straightforward approach for ϵ range queries, which simply intersects the probability mass of each object with the query region, denoted as *NAIVE* and
- [122] for probabilistic kNN queries.

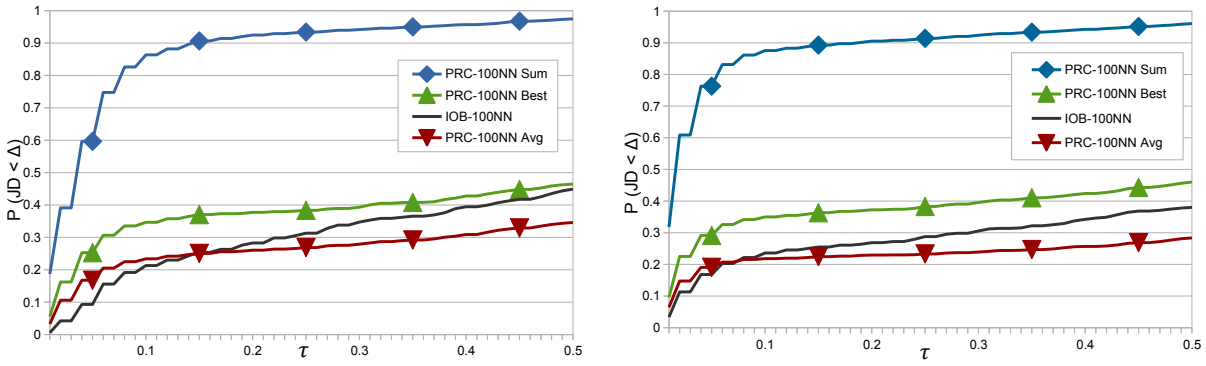
For the evaluated query predicates we select a random object as query object. For stability of results, each experiment is run 100 times, randomly choosing a new query object in each run.

11.6.4 Evaluation of Result Quality

The main contribution of this work is our claim to be able to provide useful result representatives. That is, for a $\tau - \hat{P}-$representative we can guarantee, at a level of significance α, that the ground-truth has a probability of at least $\hat{P}$ to have a distance not greater than τ. Clearly, if either $\hat{P}$ is too low or τ is too large, a representative is not useful to a user. In this section, we evaluate how useful our representatives are in terms of both $\hat{P}$ and τ. The query predicate used in per default in this section is a $k = 100$-nearest-neighbor query.

Evaluation of the Ground-Truth Distance

In this section we evaluate the distance between the results of all competitors and the (unknown) ground-truth result. Therefore, we vary a domain- and user-specific distance threshold τ that defines a "good" result, i.e., a result having a distance of at most τ to the ground-truth. For each competitor, we evaluate the probability of having a good result, depending on the choice of τ. For all our approaches, the result can be found in Figure 11.3(a) showing the fraction of of experiments where the respective result has a distance of at most τ to the ground-truth. This figure shows the average Jaccard similarity between the ground-truth and any representative, denoted as and *PRC-AVG* and the distance to the cluster representative most similar to the ground-truth, denoted as and *PRC-BEST*. We also evaluated the average (*COVER-AVG*) and best (*COVER-BEST*) representatives returned by the *COVER* approach. However, in terms of result quality, these approaches performed extremely similar to *PCR*, such that we omit these lines in Figure 11.3(a) and Figure 11.3(b).



(a) T-Drive (b) IIP

Figure 11.3: Result accuracy for different values of τ

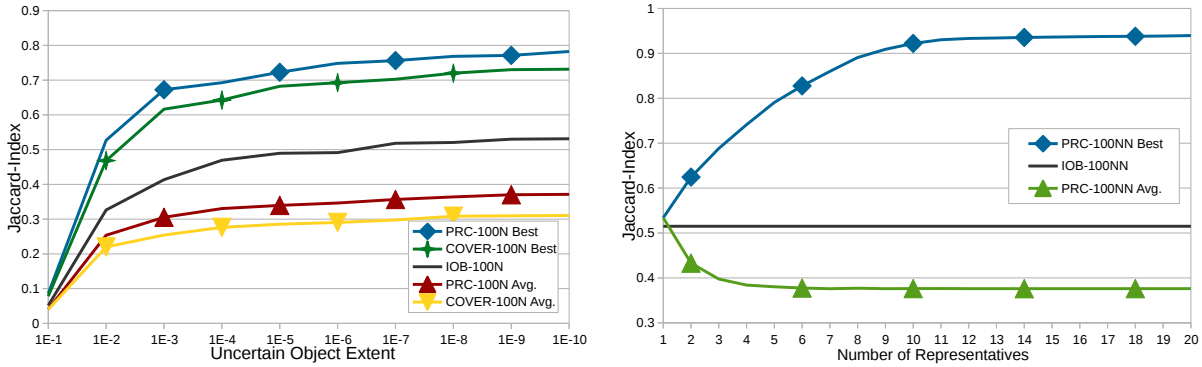
Clearly, increasing the distance threshold τ increases the probability $P(JD < \tau)$ of having a Jaccard distance of less than τ. It can be observed that the traditional *IOB* approaches have a probability of zero of having a zero-distance to the ground-truth, whereas the representative solutions have a probability greater than zero. The reason is that the *IOB* approaches do not necessarily return a result set that is a possible result. As τ increases to a Jaccard distance of 0.05, i.e., requiring the result set to have at least 95 out of the 100-nearest-neighbor in common with the ground-truth, we see that, the probability for any result to have a distance smaller than τ to the ground-truth remains low for all representatives. Yet, we see that the average representative returned by our result still performs better than the *IOB* solution. More importantly however, we can see that for $\tau = 0.05$, we already obtain a probability of 20% of being τ-similar to the best representative, and a probability of around 80% of being τ-similar to any of the 10-representatives returned by the *COVER* and *PRC* approaches in this setting. Thus, our set of representative results is capable, with a high probability, of very closely capturing the (unknown) ground-truth result.

As the allowed distance to the ground-truth τ increases to more than 0.1, the traditional *IOB* solutions gain merit: For a large τ, this approach becomes able to reach more possible results. Still, the main drawback of these traditional approaches remains: These approaches are not able to assess their similarity to the ground-truth, whereas the *COVER* and *PRC* approaches allow to return their confidence of having a τ-similar result by judging the $\hat{P}$ significance values of the obtained result representatives. We performed the same set of experiments on the *IIP* dataset in Figure 11.3(b), showing similar results.

Evaluation of the Uncertainty

In the next experiment, we scale, for each object $o \in \mathcal{DB}$ the size of the uncertainty region $r(o)$.[3] Figure 11.4(a) displays the Jaccard Index resulted for all approaches over a database

[3]see Section 11.6.1 for a description how these regions were generated



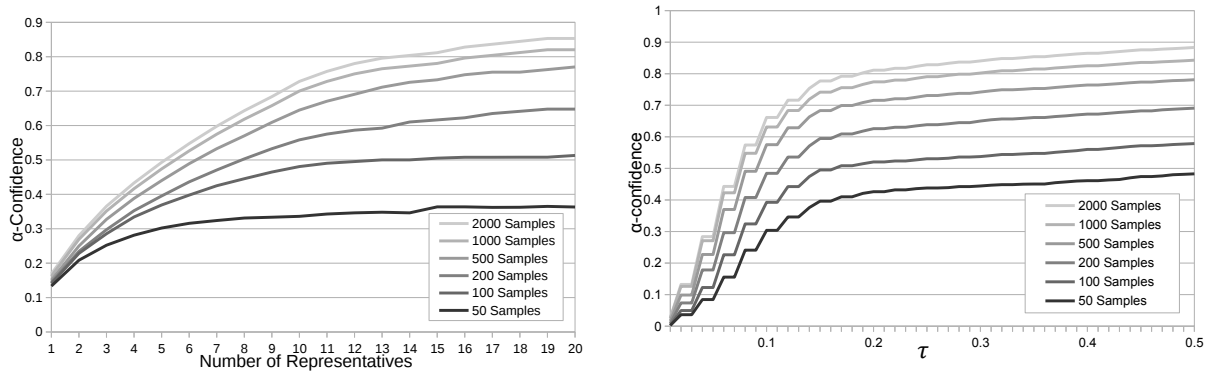
(a) Accuracy for different uncertain object extents (b) Accuracy for different numbers of representatives

Figure 11.4: Effects of uncertainty extent and number of representatives on accuracy

of differently sized uncertain regions from which the samples were drawn. It becomes apparent that result accuracy generally improves for all approaches as the uncertainty regions become smaller, however, the best results from representative approaches *COVER* and *PRC* reach high accuracy already for uncertainty regions having maximum side lengths no larger than $1E-3 = 0.001$ on the database. For the extreme case of having an uncertainty of 0.1, i.e., having each uncertainty region having cover 10% of the space of each dimension, all of the approaches fail: for such an extreme uncertainty the result converges to a uniform distribution over all database objects.

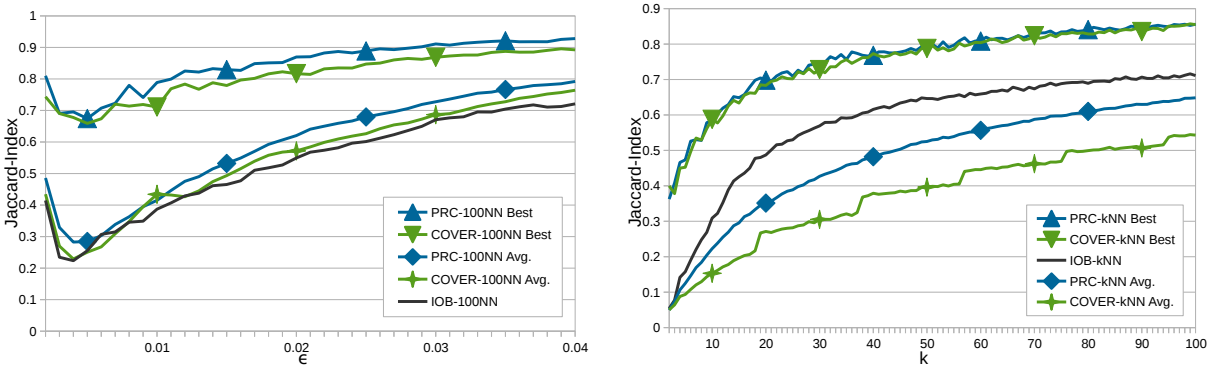
Evaluation of Number n of Representatives

In the previous experiment, both our *COVER* and *PRC* approaches used $n = 10$ representatives to describe the uncertain query result. Now, we show how this choice of n impact the result quality. Figure 11.4(b) evaluates result quality in terms of Jaccard similarity to the ground-truth, for a varying number of n. Clearly, the traditional *IOB* approach only returns a single result, such that it's quality is independent of n. We see, that increasing the number of representatives decreases the average quality of each representative. This can be explained by the diversity that increases with increasing the number of representatives. On the other hand, the distance of the representative that is closest to the ground-truth result decreases when increasing the number of representatives, as a larger number of representatives increases the probability of having a representative close to the ground-truth. On the other hand, representatives have to compete with each other to represent a fraction of the database as large as possible, thus decreases the average quality of a representative. In summary, we see that the Jaccard similarity to the most-similar representative increases very quickly, indicating we are successfully able to represent the ground-truth, even for a fairly low number of less than ten representative results.



(a) *PRC* for different numbers of representatives (b) COVER for different values of τ

Figure 11.5: Evaluation of number of samples $|\mathcal{S}|$



(a) Jaccard Index for different values of ϵ (b) Jaccard Index for different values of k

Figure 11.6: Accuracy for different query types

Evaluation of Number of Sampled Query Results $|\mathcal{S}|$

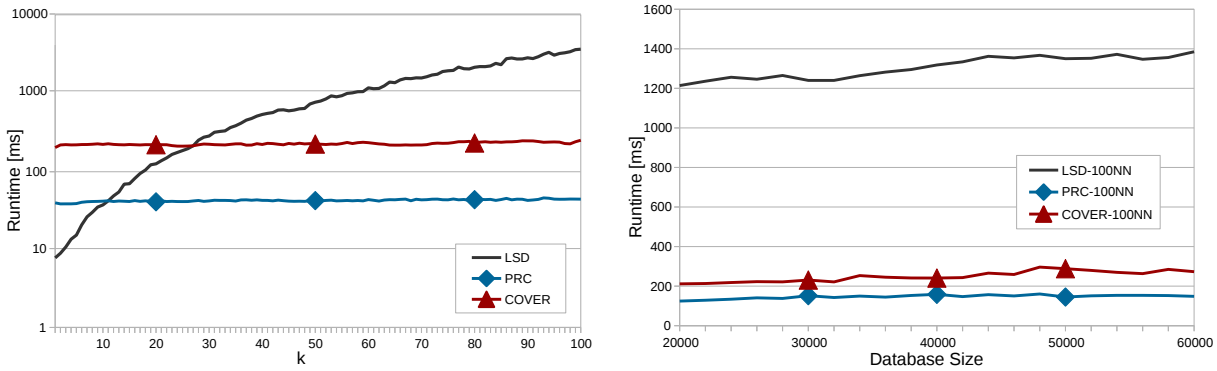
In the next experiment, we evaluate the number of samples $|\mathcal{S}|$ required in order to obtain significant results on the T-Drive data set. For this purpose, we aggregated the $\hat{P}(w_i, \tau, \alpha)$ of all representatives w_i for $\alpha = 0.99$ for different values of $|\mathcal{S}|$. For the *PRC* approach, we cut the τ range of each representative to a maximum of $\tau = 0.1$. The result is shown in Figure 11.5(a) for the *PRC* approach and in Figure 11.5(b) for the *COVER* approach. We can see that a larger sample size $|\mathcal{S}|$ increases the lower probability bounds. Also, we that after having $|\mathcal{S}| = 1000$ samples, the gain in confidence is marginal in all settings, i.e., or any number of representatives for *PRC* and any value of τ for the *COVER* approach. This is contribute to the fact that a binomial variable can be estimated highly accurately with 1000 samples. More information on obtaining confidence intervals for a binomials such as $P(w_i, \tau)$ can be found in the literature [80].

Evaluation of Query Selectivity

In all previous settings, the query predicate was a $k = 100$-nearest neighbor query. In the next set of experiments, we consider different query predicates having different selectivity. Here, we first consider ϵ-Range queries. As discussed in Section 6.2, ϵ-Range Queries have the special property of objects not influencing each other. Thus, the presence or absence of one object in a specified query range does not affect the presence or absence of another object, for a deterministic query object. Thus, the probability of two possible result object to appear together in the result can be computed simply by the product of their individual probabilities. Consequently, existing solutions which compute result probabilities of individual objects only are also able to infer the probabilities of whole result sets. Yet, this property no longer holds if the query object is also uncertain. Figure 11.6(a) shows the similarity of all solutions in terms of Jaccard similarity to the ground-truth on the *IIP* dataset, while varying the query range ϵ. First, we see that for an extremely low choice of ϵ, all approaches seem to be performing extremely well. This is contributed to the fact that for extremely small query ranges, both the ground-truth and the returned results may be empty. This trivial case results in a perfect Jaccard similarity of one. We see that the result of the object-based approach *IOB* shows the highest dissimilarity to the ground-truth. We see that for both *COVER* and *PRC*, the average result quality of all representatives is higher than the quality of the *IOB* solutions. The reason is that an *IOB* approach selects the best result objects among many different positions of the query object q. Yet, some result objects may be unlikely to appear in the same result, as they appear in possible worlds having significantly different positions of the query object, as explained in Figure 11.6(a). Furthermore we that see the best cluster representative always has a high similarity to the ground-truth. This supports our claim that our representative results, as returned by our *COVER* and *PRC* approaches, are more useful for decision making, as they allow to propagate the uncertainty of a database to the uncertainty of a query result. Finally, we see that for large values of ϵ, the quality of the *IOB* approach appears to improve. The reason is that for a large ϵ, the inhomogeneity of possible query results decreases: a large fraction of the database is guaranteed to be a result. Thus, these objects will be returned by any of the evaluated solutions, such that the Jaccard Similarity improves globally. For kNN queries, we observed a similar trend in Figure 11.7(b). In this case, the object-based approach *IOB* performs significantly better. The reason is that for kNN queries, the *IOB* approch is able to perfectly guess the number of objects in the result set, as it is always equal to k.

11.6.5 Runtime Experiments

The previous experiments all focused on evaluating the quality of the result, in terms of distance to the (unknown) ground-truth. In the next set of experiments, we evaluate the runtime required to obtain representative queries. Our runtime results are shown in Figure 11.7(a), for kNN queries and for a varying value of k. We scale the parameter k in order to simulate queries having different run-times. Overall, we can see that the *COVER*



(a) Runtime (logarithmic scale) for different k

(b) Runtime for different sizes of $\mathcal{DB}$

Figure 11.7: Runtime comparisons

approach has a significantly higher runtime than the *PRC* approach. This is contributed to the heuristic of iteratively and greedily choosing cluster representatives, which requires to compute pair-wise distances between all $|\mathcal{S}| = 1000$ sampled results. In contrast, the *PRC* requires to compute relatively few distance computations in the possible result space: this approach uses a We observe that the state-of-the-art algorithm *LSD* of [122], denoted as *LSD*, runs super-linear in k, as this algorithm is required to maintain a generating function of length k. This polynomial has to be multiplied with a generating polynomial of constant length for each database object that has a non-zero chance to be part of the result. Clearly, the number of candidate objects increases in k. We further observe that the total runtime of the *COVER* and the *PRC* approaches appear to be constant in k. Looking closely, we observe a slight increase in runtime, because runtime of kNN queries is dominated by the (constant-in-k) time required to sample a possible world to run the query on.

Furthermore, we increased the size of the *IIP* dataset in Figure 11.7, by adding iceberg sightings from years prior to 2014. We can see that the size of the database hardly influences any of the competitors. The reason is that for an increasing database size, the number of $k = 10$NN candidates increases sublinear. By using a spatial index, the *LSD* approach [122], which we use to compute the object-based *LSD* approach, scales linear in this number, independent of the database size. Our approach uses the same index, to identify database objects that are guaranteed to have a zero probability to be in the query result, thus explaining our scalability in the database size.

11.7 Conclusions

So far, existing probabilistic querying solutions could not be scaled to realistic data sets due to #P-complete nature of querying uncertain data. This work presents a new approach to query uncertain sets of data by querying possible database worlds, each resulting in a possible query result. From this result we extract a small number of representative query results, that are both diverse and have a sufficiently high probability to closely resemble the true result of the query. We believe that this result will enable users to make much more educated guesses about the true nature of the result than by using previous approaches.

Part IV

Summary

Conclusions

Main focus of this thesis is issuing queries and performing data mining in enriched geo-spatial data. After an introduction of (geo-)spatial data, an overview over different forms and applications of enrichment was given. The remainder of this thesis was subdivided in two – a part on deterministic data types used for enrichment, and one on uncertain data.

Part II: Deterministic Enriched Geo-Spatial Data.

Exemplifying deterministic data types, three use cases are given. In the first one, **Trip Planning on Touristic Data** (Chapter 5), different types of crowd-sourced and mostly qualitative data are used to enrich a road network for touristic routing with the objective of guiding tourists along paths that may be more interesting to them. To achieve this, we extracted knowledge from heterogeneous sources of user-generated spatial data and used it to either directly enrich road networks or construct meta-networks that map the qualitative user preferences. Paths resulting from applying traditional routing algorithms to these enriched networks correspond to varying crowd sentiment depending on the actual data source used. Experiments were conducted using a broad variety of real-world datasets and road networks with different characteristics. Our results clearly show that incorporating qualitative information into road networks is not only feasible but leads to enhanced solutions for the task of path computation, i.e., paths which reflect qualitative knowledge without sacrificing their quantitative aspect.

Path generation could be greatly improved in the future by incorporating techniques for the arc-orienteering problem where routing can also cope with benefits instead of cost functions only. In addition, datasets used for enrichment were mainly limited by their availability for research. Access to larger coverage areas or more data attributes would be a great addition to the existing methods and could improve results drastically.

In **Parking Queries on Road Network Data** (Chapter 6), additional knowledge about the road graph is modelled as a set of resources, along which the user is guided to maximize the probability of encountering an available resource. We aim to find a route with minimal expected cost among all routes exceeding a given probability threshold. We propose a framework in which resources are modeled as continuous-time Markov chains with two states, `available` and `consumed`, where consumed resources are allowed to reappear and short term as well as long term observations are taken into account. Since the introduced query has an unlimited search space, we propose approximate as well as optimal solutions. We employ two different search heuristics in a greedy algorithm to achieve a

trade-off between accuracy and calculation time. Furthermore, solutions using backtracking and a branch and bound approach provide optimal solutions in competitive time. We demonstrate the superiority of our model as well as the efficiency and effectiveness of our algorithms on two realistic applications – the search of a vacant parking spot, and the search for a vacant charging station for electric vehicles.

A promising future direction for this topic could on the one hand be the incorporation of a larger set of ground truth in evaluation and model training, e.g., sensors that inform whether a parking spot is consumed. Sojourn times of the model could be adjusted accordingly to provide more accurate path suggestions. On the other hand, the incorporation of other types of observations could serve to improve the model as well. This could be observations by drivers or more accurately formulated edge costs.

As a third example for deterministic data, **Trend Dissemination Mining on Social Media Data** (Chapter 7) discusses the dissemination of trends in space and time. For historic trends, the corresponding user-generated tweets including their geo-tags are incorporated into a spatio-temporal dissemination model describing their flow in spatial and temporal dimension. By applying a tensor factorization approach, we extracted latent features of trends, to which we applied a clustering approach to obtain sets of trends having a similar dissemination archetype. Our qualitative evaluation of these trend archetypes on Twitter trends show meaningful dissemination archetypes, such as political trends, celebrity trends, and disaster trends. Our quantitative analysis shows that our tensor factorization yields are high approximation quality for a low number of latent features. This result implies that a small number of latent features we derive from the flow of each trend is able to discriminate trends with a high-precision.

This topic may provide a number of questions to be addressed in the feature. Namely, to enable classification for on-line streaming environments where spatial decomposition as well as archetype classes haven been developed and researched so that automated decision-making works reliably. This allows for archetype classification of emerging trends to predict their upcoming spread.

Part III: Uncertain Enriched Geo-Spatial Data.

Primarily, an overview of uncertain data models is given (Chapter 8). Subsequently, Approximate Probabilistic Nearest Neighbor Queries (cf. Chapter 9) introduces index-supported techniques approximating the shape of Voronoi-cells to support nearest neighbor queries on uncertain data. Since object attribute values as well as their corresponding Voronoi cells are random variables, we propose to approximate, progressively and conservatively, the space which is guaranteed to be part of the Voronoi-cell of U, denoted as the guaranteed Voronoi-cell $\mathcal{V}_k^{\forall}$. In addition, we also approximate the space which has a non-zero probability to be part of the Voronoi-cell of U, which we call the possible-Voronoi cell $\mathcal{V}_k^{\exists}$. We show how our solutions can be extended to k'th-order Voronoi cells, i.e., the space which is guaranteed to (may possibly) have U as one of their k-nearest neighbors. Our approach uses an R^*-tree as a hierarchical access method to efficiently find the set of

uncertain objects that influence the possible Voronoi-cell of an uncertain object U, i.e., the set of Delaunay-neighbors of U. In addition, we propose to use a *kd*-trie as a hierarchical access method to identify regions of space which must (not) be part of a Voronoi-cell. Compared to the state-of-the-art of computing uncertain Voronoi-cells, our approach allows for much higher approximation quality, since our result approximation consists of a set of rectangular *kd*-trie nodes, rather than a single bounding rectangle.

A promising direction for the future would be a Voronoi-decomposition of the entire data space to speed up nearest neighbor queries. Although an iterative approach using the proposed methods is already feasible, there is a lot of possibility for efficiency gains.

In **Approximate Probabilistic Representative Pattern Mining** (Chapter 10) we present a general solution to cluster uncertain objects. Our challenge was to develop a framework making any clustering that has been developed for certain data applicable for the case of uncertain data. We approach this challenge by employing a sampling approach to obtain a number of possible database instances from the uncertain database. Applying a domain specific clustering algorithm to each obtained database instance yields a possibly large set of different clusterings. The new challenge is thus how to find a consensus between all these possible clusterings. For this purpose, we define the notion of representative τ-ϕ-clusterings: a representative τ-ϕ- clustering is a clustering having probability at least ϕ to have a distance of at most τ to the actual clustering of the data if the data were certain. Our solution follows a sampling approach, which returns cluster representatives that are guaranteed to be τ-ϕ-clusterings at a user specified level of significance. To the best of our knowledge, our approach is the first to yield clusterings associated with confidence, allowing the user to assess the quality of the clustering result. Furthermore, by returning multiple representative clusterings to the user, we can improve the quality (and therefore usefulness) of results, as shown by our experimental study.

Approximate Probabilistic Representative Spatial Query Processing (Chapter 11) extends the same concept to various approximate queries on uncertain data. Again, the presented framework incorporates a sampling approach to obtain possible world instances from an uncertain database, each of which being deterministic and resulting in a possible query result using a traditional query processing method. Evaluating these results with an appropriate similarity measure, the framework then returns a number of representative results, that are both diverse and have a sufficiently high probability to closely resemble the true result of the query. Interesting future directions of this topic include the incorporation and evaluation of other query types, especially SPJ-queries.

Acknowledgements

Finishing this dissertation was only possible with the continued support and belief of many. Naming every single person would greatly exceed this page, however, I wish to express my deepest gratitude to all who have encouraged me.

First and foremost, I must extend a very deeply heartfelt thank you to the people in my personal life who never ceased to support and invigorate me, in so doing they enabled me to pursue the path of a computer scientist in the first place. In moments of triumph they shared my joy, and when I needed to hold on to someone they gladly offered their shoulders. I would not be in this position without every single one of you.

To my supervisor, project P.I., and first examiner Prof. Dr. Matthias Renz I want to express my thanks as he gave me the opportunity to pursue the topics that led to this thesis and who was ready to advise me whenever there were uncertainties. I am especially thankful to Prof. Dr. Christian Jensen for agreeing to be the second examiner to this thesis, and to Prof. Dr. Hans-Peter Kriegel for first giving me the opportunity to join the research group at LMU.

To the person ultimately responsible for my presence in science – Prof. Dr. Mario Nascimento from *UofA* in Edmonton, who supervised my Diploma thesis, first brought up the idea of pursuing a PhD, and who was always there with helpful advice, a cheerful story, and productive input for most of my research projects, thank you so very much.

Of my (former) colleagues at LMU I especially want to mention Dr. Andreas Zufle who probably had the biggest influence on my research interests and who always seemed to be able to motivate me to move on when nobody else could.

Prof. Dr. Matthias Schubert, Prof. Dr. Peer Kröger, and Dr. Tobias Emrich all from LMU advised me on many projects and were incredibly helpful with administrational concerns. Fellow PhD candidates Markus Mauder, Gregor Jossé, and Michael Weiler shared many of our research group's special moments – whiteboard discussion meetings, late-night pre deadline power sessions, and finally the well-deserved (well, usually) notifications of acceptance.

Daily life in the research group together would have been completely unimaginable without Susanne Grienberger and Franz Krojer who make sure everything runs smoothly.

Finally, I want to thank the one who accompanied me for most of my research trajectory, helped me express my ideas, and stood by me in my every moment of need – my trusted Toshiba Satellite ultrabook running on Arch Linux.

Bibliography

[1] Abdelhaq, H., Sengstock, C., and Gertz, M. EvenTweet: Online localized event detection from Twitter. *Proceedings of the VLDB Endowment 6*, 12 (2013), 1326–1329.

[2] Achtert, E., Goldhofer, S., Kriegel, H.-P., Schubert, E., and Zimek, A. Evaluation of clusterings – metrics and visual support. In *Proceedings of the 28th International Conference on Data Engineering (ICDE), Washington, DC* (2012), pp. 1285–1288.

[3] Achtert, E., Kriegel, H.-P., Schubert, E., and Zimek, A. Interactive data mining with 3D-Parallel-Coordinate-Trees. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), New York City, NY* (2013), pp. 1009–1012.

[4] Agrawal, P., Benjelloun, O., Sarma, A. D., Hayworth, C., Nabar, S., Sugihara, T., and Widom, J. Trio: A system for data, uncertainty, and lineage. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB), Seoul, Korea* (2006).

[5] Agrawal, P., Sarma, A. D., Ullman, J., and Widom, J. Foundations of uncertain-data integration. *Proceedings of the VLDB Endowment 3*, 1-2 (2010), 1080–1090.

[6] Akdogan, A., Demiryurek, U., Banaei-Kashani, F., and Shahabi, C. Voronoi-based geospatial query processing with mapreduce. In *IEEE CloudCom* (2010), IEEE, pp. 9–16.

[7] Ali, M. E., Tanin, E., Zhang, R., and Kotagiri, R. Probabilistic voronoi diagrams for probabilistic moving nearest neighbor queries. *DKE 75* (2012), 1–33.

[8] Alvares, L. O., Bogorny, V., Kuijpers, B., de Macedo, J. A. F., Moelans, B., and Vaisman, A. A model for enriching trajectories with semantic geographical information. In *Proc. of the 15th Annual ACM Int'l Symp. on Advances in Geographic Information Systems* (2007), ACM, pp. 22:1–22:8.

[9] Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. OPTICS: Ordering points to identify the clustering structure. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Philadelphia, PA* (1999), pp. 49–60.

[10] Antova, L., Jansen, T., Koch, C., and Olteanu, D. Fast and simple relational processing of uncertain data. In *Proceedings of the 24th International Conference on Data Engineering (ICDE), Cancun, Mexico* (2008).

[11] Aurenhammer, F. Voronoi diagrams-a survey of a fundamental geometric data structure. *ACM CSUR 23*, 3 (1991), 345–405.

[12] Bacchus, F., Grove, A. J., Halpern, J. Y., and Koller, D. From statistical knowledge bases to degrees of belief. *Artificial intelligence 87*, 1 (1996), 75–143.

[13] Barbará, D., Garcia-Molina, H., and Porter, D. The management of probabilistic data. *IEEE Transactions on knowledge and data engineering 4*, 5 (1992), 487–502.

[14] Beckmann, N., Kriegel, H.-P., Schneider, R., and Seeger, B. The R*-Tree: An efficient and robust access method for points and rectangles. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Atlantic City, NJ* (1990), pp. 322–331.

[15] Benjelloun, O., Sarma, A. D., Halevy, A., and Widom, J. ULDBs: Databases with uncertainty and lineage. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB), Seoul, Korea* (2006), pp. 1249–1264.

[16] Bernecker, T., Emrich, T., Kriegel, H.-P., Mamoulis, N., Renz, M., and Züfle, A. A novel probabilistic pruning approach to speed up similarity queries in uncertain databases. In *Proceedings of the 27th International Conference on Data Engineering (ICDE), Hannover, Germany* (2011), pp. 339–350.

[17] Bernecker, T., Kriegel, H.-P., Mamoulis, N., Renz, M., and Züfle, A. Scalable probabilistic similarity ranking in uncertain databases. *IEEE Transactions on Knowledge and Data Engineering 22*, 9 (2010), 1234–1246.

[18] Bernecker, T., Kriegel, H.-P., Renz, M., Verhein, F., and Züfle, A. Probabilistic frequent itemset mining in uncertain databases. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Paris, France* (2009), pp. 119–128.

[19] Bernecker, T., Kriegel, H.-P., Renz, M., and Züfle, A. Hot item detection in uncertain data. In *Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Bangkok, Thailand* (2009), pp. 673–680.

[20] Beskales, G., Soliman, M. A., and Ilyas, I. F. Efficient search for the top-k probable nearest neighbors in uncertain databases. *Proceedings of the VLDB Endowment 1*, 1 (2008), 326–339.

[21] Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006.

[22] Böhm, C., Pryakhin, A., and Schubert, M. The Gauss-tree: Efficient object identification of probabilistic feature vectors. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE), Atlanta, GA* (2006), p. 9.

[23] Boulos, J., Dalvi, N., Mandhani, B., Mathur, S., Re, C., and Suciu, D. Mystiq: a system for finding more answers by using probabilities. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Baltimore, ML* (2005), pp. 891–893.

[24] Brilhante, I., Macedo, J. A., Nardini, F. M., Perego, R., and Renso, C. Where shall we go today?: Planning touristic tours with tripbuilder. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management* (New York, NY, USA, 2013), CIKM '13, ACM, pp. 757–762.

[25] Brown, L. D., Cai, T., and DasGupta, A. Interval estimation for a binomial proportion. *Stat. Sci. 16*, 2 (2001), 101–133.

[26] Campello, R. J. G. B., Moulavi, D., and Sander, J. Density-based clustering based on hierarchical density estimates. In *Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Gold Coast, Australia.* 2013, pp. 160–172.

[27] Carroll, J. D., and Chang, J.-J. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika 35*, 3 (1970), 283–319.

[28] Cavallo, R., and Pittarelli, M. The theory of probabilistic databases. In *VLDB* (1987), vol. 87, pp. 1–4.

[29] Cheema, M. A., Lin, X., Wang, W., Zhang, W., and Pei, J. Probabilistic reverse nearest neighbor queries on uncertain data. *IEEE Transactions on Knowledge and Data Engineering 22*, 4 (2010), 550–564.

[30] Chen, C., Zhang, D., Guo, B., Ma, X., Pan, G., and Wu, Z. Tripplanner: Personalized trip planning leveraging heterogeneous crowdsourced digital footprints. *IEEE Transactions on Intelligent Transportation Systems 16*, 3 (2015), 1259–1273.

[31] Chen, H., Ku, W.-S., Sun, M.-T., and Zimmermann, R. The multi-rule partial sequenced route query. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACMGIS'08)* (2008), pp. 10:1–10:10.

[32] Chen, H., Ku, W.-S., Sun, M.-T., and Zimmermann, R. The partial sequenced route query with traveling rules in road networks. *Geoinformatica 15*, 3 (2011), 541–569.

[33] Chen, Z., Shen, H. T., and Zhou, X. Discovering popular routes from trajectories. In *ICDE* (2011), pp. 900–911.

[34] Cheng, R., Chen, J., Mokbel, M., and Chow, C. Probabilistic verifiers: Evaluating constrained nearest-neighbor queries over uncertain data. In *Proceedings of the 24th International Conference on Data Engineering (ICDE), Cancun, Mexico* (2008).

[35] Cheng, R., Chen, L., Chen, J., and Xie, X. Evaluating probability threshold k-nearest-neighbor queries over uncertain data. In *Proceedings of the 12th International Conference on Extending Database Technology (EDBT), Saint-Petersburg, Russia* (2009), pp. 672–683.

[36] Cheng, R., Kalashnikov, D. V., and Prabhakar, S. Evaluating probabilistic queries over imprecise data. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), San Diego, CA* (2003), pp. 551–562.

[37] Cheng, R., Kalashnikov, D. V., and Prabhakar, S. Querying imprecise data in moving object environments. In *IEEE Transactions on Knowledge and Data Engineering* (2004).

[38] Cheng, R., Xia, Y., Prabhakar, S., Shah, R., and Vitter, J. S. Efficient indexing methods for probabilistic threshold queries over uncertain data. In *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB), Toronto, Canada* (2004), pp. 876–887.

[39] Cheng, R., Xie, X., Yiu, M. L., Chen, J., and Sun, L. Uv-diagram: A voronoi diagram for uncertain data. In *Proceedings of the 26th International Conference on Data Engineering, ICDE 2010, March 1-6, 2010, Long Beach, California, USA* (2010), pp. 796–807.

[40] Clopper, C., and Pearson, E. S. Probable inference, the law of succession, and statistical inference. *Biometrika 26* (1934), 404–413.

[41] Cormode, G., Li, F., and Yi, K. Semantics of ranking queries for probabilistic data and expected results. In *Proceedings of the 25th International Conference on Data Engineering (ICDE), Shanghai, China* (2009), pp. 305–316.

[42] Cormode, G., and Muthukrishnan, S. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms 55*, 1 (2005), 58–75.

[43] Cui, Y., Fern, X. Z., and Dy, J. G. Non-redundant multi-view clustering via orthogonalization. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM), Omaha, NE* (2007), pp. 133–142.

[44] Dalvi, N., and Suciu, D. Efficient query evaluation on probabilistic databases. *The VLDB Journal 16*, 4 (2007), 523–544.

[45] Dalvi, N. N., Ré, C., and Suciu, D. Probabilistic databases: diamonds in the dirt. *Communications of the ACM 52*, 7 (2009), 86–94.

[46] De Choudhury, M., Feldman, M., Amer-Yahia, S., Golbandi, N., Lempel, R., and Yu, C. Automatic construction of travel itineraries using social breadcrumbs. In *Proceedings of the 21st ACM Conference on Hypertext and Hypermedia* (New York, NY, USA, 2010), HT '10, ACM, pp. 35–44.

[47] Delis, A., Efstathiou, V., and Verroios, V. Reaching available public parking spaces in urban environments using ad hoc networking. In *Proceedings of the 12th International Conference on Mobile Data Management (MDM'11)* (2011), pp. 141–151.

[48] Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 39*, 1 (1977), 1–31.

[49] Deshpande, A., Guestrin, C., Madden, S. R., Hellerstein, J. M., and Hong, W. Model-based approximate querying in sensor networks. *The VLDB Journal 14*, 4 (2005), 417–443.

[50] Dijkstra, E. W. A note on two problems in connection with graphs. *Numerische Mathematik 1* (1959), 269–271.

[51] Dolev, N., Doytsher, Y., Kanza, Y., Safra, E., and Sagiv, Y. Computing a k-route over uncertain geographical data. In *Proceedings of the 10th International Conference on Advances in Spatial and Temporal Databases (SSTD'07)* (2007), pp. 276–293.

[52] Dong, X., Halevy, A. Y., and Yu, C. Data integration with uncertainty. In *VLDB Endowment* (2007), pp. 687–698.

[53] Duckham, M., and Kulik, L. "simplest" paths: Automated route selection for navigation". 169–185.

[54] Duckham, M., Winter, S., and Robinson, M. Including landmarks in routing instructions. *Journal on Location Based Services Vol. 4 4* (2010), 28–52.

[55] Emrich, T., Kriegel, H.-P., Kröger, P., Renz, M., and Züfle, A. Incremental reverse nearest neighbor ranking in vector spaces. In *Proceedings of the 11th International Symposium on Spatial and Temporal Databases (SSTD), Aalborg, Denmark* (2009), pp. 265–282.

[56] Emrich, T., Kriegel, H.-P., Kröger, P., Renz, M., and Züfle, A. Boosting spatial pruning: On optimal pruning of MBRs. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Indianapolis, IN* (2010), pp. 39–50.

[57] Emrich, T., Kriegel, H.-P., Mamoulis, N., Renz, M., and Züfle, A. Querying uncertain spatio-temporal data. In *Proceedings of the 28th International Conference on Data Engineering (ICDE), Washington, DC* (2012).

[58] Emrich, T., Schmid, K. A., Züfle, A., Renz, M., and Cheng, R. Approximate uv computation based on space decomposition. In *Proceedings of the 14th International Symposium on Spatial and Temporal Databases (SSTD), Hong Kong, China* (2015).

[59] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (KDD), Portland, OR* (1996), pp. 226–231.

[60] Feldman, D., Sugaya, A., Sung, C., and Rus, D. idiary: From gps signals to a text-searchable diary. In *Proc. of the 11th ACM Conf. on Embedded Networked Sensor Systems* (2013), ACM, pp. 6:1–6:12.

[61] Finch, T. Incremental calculation of weighted mean and variance. Tech. rep., University of Cambridge, 2009.

[62] Fuhr, N., and Rölleke, T. A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Systems (TOIS) 15*, 1 (1997), 32–66.

[63] Fuller, R., and Kuromiya, K. *Critical Path*. St. Martin's Press, 1982.

[64] Garcia, A., Arbelaitz, O., Linaza, M. T., Vansteenwegen, P., and Souffriau, W. *Personalized tourist route generation*. Springer, 2010.

[65] Gavalas, Konstantopoulos, C., Mastakas, K., and Pantziou, G. A survey on algorithmic approaches for solving tourist trip design problems. *Journal of Heuristics 20*, 3 (2014), 291–328.

[66] Ghinita, G., Karras, P., Kalnis, P., and Mamoulis, N. Fast data anonymization with low information loss. In *VLDB* (2007), pp. 758–769.

[67] Gidofalvi, G., Pedersen, T. B., Risch, T., and Zeitler, E. Highly scalable trip grouping for large-scale collective transportation systems. In *Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology* (New York, NY, USA, 2008), EDBT '08, ACM, pp. 678–689.

[68] Graf, F., Kriegel, H.-P., Renz, M., and Schubert, M. MARiO: Multi attribute routing in open street map. In *Proceedings of the 12th International Symposium on Spatial and Temporal Databases (SSTD), Minneapolis, MN* (2011), pp. 486–490.

[69] Gullo, F., Ponti, G., and Tagarelli, A. Clustering uncertain data via k-medoids. In *Scalable Uncertainty Management* (2008), pp. 229–242.

[70] Gullo, F., Ponti, G., and Tagarelli, A. Minimizing the variance of cluster mixture models for clustering uncertain objects. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on* (2010), pp. 839–844.

[71] Gullo, F., Ponti, G., and Tagarelli, A. Minimizing the variance of cluster mixture models for clustering uncertain objects. *Statistical Analysis and Data Mining 6*, 2 (2013), 116–135.

[72] Gullo, F., Ponti, G., Tagarelli, A., Tradigo, G., and Veltri, P. Hierarchical clustering of microarray data with probe-level uncertainty. In *CBMS* (2009), pp. 1–6.

[73] Gullo, F., and Tagarelli, A. Uncertain centroid based partitional clustering of uncertain data. *Proceedings of the VLDB Endowment 5*, 7 (2012), 610–621.

[74] Güting, R. H. An introduction to spatial database systems. *The VLDB JournalThe International Journal on Very Large Data Bases 3*, 4 (1994), 357–399.

[75] Harshman, R. A. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics 16*, 1 (1970), 84.

[76] Hart, P. E., Nilsson, N. J., and Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics 4*, 2 (July 1968), 100–107.

[77] Hey, T., Tansley, S., and Tolle, K., Eds. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, Redmond, Washington, 2009.

[78] Hjaltason, G. R., and Samet, H. Ranking in spatial databases. In *Proceedings of the 4th International Symposium on Large Spatial Databases (SSD), Portland, ME* (1995), pp. 83–95.

[79] Hochbaum, D. S. Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems. In *Approximation algorithms for NP-hard problems*, D. S. Hochbaum, Ed. PWS Publishing Co., Boston, MA, USA, 1997, pp. 94–143.

[80] Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association 58*, 301 (1963), 13–30.

[81] Hsieh, H., and Li, C. Mining and planning time-aware routes from check-in data. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014* (2014), pp. 481–490.

[82] Hsieh, H. P., Li, C. T., and Lin, S. D. Exploiting large-scale check-in data to recommend time-sensitive routes. In *Proceedings of the ACM SIGKDD International Workshop on Urban Computing* (2012), ACM.

[83] Hua, M., Pei, J., Zhang, W., and Lin, X. Ranking queries on uncertain data: a probabilistic threshold approach. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Vancouver, BC, Canada* (2008), pp. 673–686.

[84] Huang, J., Antova, L., Koch, C., and Olteanu, D. Maybms: A probabilistic database management system. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Providence, RI* (2009), pp. 1071–1074.

[85] Hubert, L., and Arabie, P. Comparing partitions. *Journal of Classification 2*, 1 (1985), 193–218.

[86] Iijima, Y., and Ishikawa, Y. Finding probabilistic nearest neighbors for query objects with imprecise locations. In *Proceedings of the 10th International Conference on Mobile Data Management (MDM), Taipei, Taiwan* (2009).

[87] Jaccard, P. Distribution de la florine alpine dans la bassin de dranses et dans quelques regiones voisines. *Bulletin de la Societe Vaudoise des Sciences Naturelles 37* (1901), 241–272.

[88] Jain, A. K. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters 31*, 8 (2010), 651–666.

[89] Jain, A. K., and Dubes, R. C. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, 1988.

[90] Jain, A. K., Murty, M. N., and Flynn, P. J. Data clustering: A review. *ACM Computing Surveys 31*, 3 (1999), 264–323.

[91] Jain, P., Meka, R., and Dhillon, I. S. Simultaneous unsupervised learning of disparate clusterings. *Statistical Analysis and Data Mining 1*, 3 (2008), 195–210.

[92] James, J. Data never sleeps 4.0, jun 2016.

[93] Jampani, R., Xu, F., Wu, M., Perez, L., Jermaine, C., and Haas, P. J. The monte carlo database system: Stochastic analysis close to the data. *ACM Trans. Database Syst. 36*, 3 (2011), 18:1–18:41.

[94] Jiang, B., Pei, J., Tao, Y., and Lin, X. Clustering uncertain data based on probability distribution similarity. *IEEE Transactions on Knowledge and Data Engineering 25*, 4 (2013), 751–763.

[95] Jin, C., Yi, K., Chen, L., Yu, J. X., and Lin, X. Sliding-window top-k queries on uncertain streams. *Proceedings of the VLDB Endowment 1*, 1 (2008), 301–312.

[96] Jossé, G., Franzke, M., Skoumas, G., Züfle, A., Nascimento, M. A., and Renz, M. A framework for computation of popular paths from crowdsourced data. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015* (2015), pp. 1428–1431.

[97] Jossé, G., Schmid, K. A., and Schubert, M. Probabilistic resource route queries with reappearance. In *Proceedings of the 18th International Conference on Extending Database Technology (EDBT), Brussels, Belgium* (2015), pp. 445–456.

[98] Jossé, G., Schmid, K. A., Züfle, A., Skoumas, G., Schubert, M., and Pfoser, D. Tourismo: A user-preference tourist trip search engine. In *Proceedings of the 14th International Symposium on Spatial and Temporal Databases (SSTD), Hong Kong, China* (2015), pp. 514–519.

[99] Jossé, G., Schubert, M., and Kriegel, H.-P. Probabilistic parking queries using aging functions. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS), Orlando, FL* (2013), p. 4.

[100] Kalyanam, J., Mantrach, A., Saez-Trumper, D., Vahabi, H., and Lanckriet, G. Leveraging social context for modeling topic evolution. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2015), ACM, pp. 517–526.

[101] Kanza, Y., Levin, R., Safra, E., and Sagiv, Y. An interactive approach to route search. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2009), GIS '09, ACM, pp. 408–411.

[102] Kanza, Y., Levin, R., Safra, E., and Sagiv, Y. Interactive route search in the presence of order constraints. *The International Journal on Very Large Data Bases (VLDB'10)* (2010), 117–128.

[103] Kanza, Y., Safra, E., and Sagiv, Y. Route search over probabilistic geospatial data. In *Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases (SSTD'09)* (2009), pp. 153–170.

[104] Kanza, Y., Safra, E., Sagiv, Y., and Doytsher, Y. Heuristic algorithms for route-search queries over geographical data. In *ACM SIGSPATIAL GIS* (2008), p. 11.

[105] Kao, B., Lee, S. D., Cheung, D. W., Ho, W. S., and Chan, K. F. Clustering uncertain data using voronoi diagrams. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM), Pisa, Italy* (2008), pp. 333–342.

[106] Karp, R. M. *Reducibility among combinatorial problems*. Springer, 1972.

[107] Kaufman, L., and Rousseeuw, P. J. *Finding Groups in Data: An Introduction to Cluster Analyis*. John Wiley&Sons, 1990.

[108] Kaufman, L., and Rousseeuw, P. J. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley&Sons, 1990, ch. Partitioning Around Medoids (Program PAM), pp. 68–125.

[109] Kolahdouzan, M., and Shahabi, C. Voronoi-based k nearest neighbor search for spatial network databases. In *VLDB Endowment* (2004), VLDB Endowment, pp. 840–851.

[110] Kriegel, H.-P., Kröger, P., Sander, J., and Zimek, A. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1*, 3 (2011), 231–240.

[111] Kriegel, H.-P., Kröger, P., and Zimek, A. Clustering high dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD) 3*, 1 (2009), 1–58.

[112] Kriegel, H.-P., Kunath, P., and Renz, M. Probabilistic nearest-neighbor query on uncertain objects. In *Proceedings of the 12th International Conference on Database Systems for Advanced Applications (DASFAA), Bangkok, Thailand* (2007), pp. 337–348.

[113] Kriegel, H.-P., and Pfeifle, M. Density-based clustering of uncertain data. In *Proceedings of the 11th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Chicago, IL* (2005), pp. 672–677.

[114] Kriegel, H.-P., and Pfeifle, M. Hierarchical density-based clustering of uncertain data. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM), Houston, TX* (2005), pp. 689–692.

[115] Lakshmanan, L. V., Leone, N., Ross, R., and Subrahmanian, V. S. Probview: A flexible probabilistic database system. *ACM Transactions on Database Systems (TODS) 22*, 3 (1997), 419–469.

[116] Lappas, T., Vieira, M. R., Gunopulos, D., and Tsotras, V. J. On the spatiotemporal burstiness of terms. *Proceedings of the VLDB Endowment 5*, 9 (2012), 836–847.

[117] Lee, S. D., Kao, B., and Cheng, R. Reducing uk-means to k-means. In *ICDM Workshops* (2007), pp. 483–488.

[118] Li, F., Cheng, D., Hadjieleftheriou, M., Kollios, G., and Teng, S.-H. On trip planning queries in spatial databases. In *Proceedings of the 9th International Conference on Advances in Spatial and Temporal Databases* (Berlin, Heidelberg, 2005), SSTD'05, Springer-Verlag, pp. 273–290.

[119] Li, H., Yu, B., and Zhou, D. Error rate analysis of labeling by crowdsourcing. In *Proc. Machine Learning meets Crowdsourcing, Workshop at the Int Conference on Machine Learning (ICML-2013)* (2013).

[120] Li, J., and Deshpande, A. Consensus answers for queries over probabilistic databases. In *Proceedings of the Twenty-eighth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems* (New York, NY, USA, 2009), PODS '09, ACM, pp. 259–268.

[121] Li, J., and Deshpande, A. Ranking continuous probabilistic datasets. *Proceedings of the VLDB Endowment 3*, 1–2 (2010), 638–649.

[122] Li, J., Saha, B., and Deshpande, A. A unified approach to ranking in probabilistic databases. *Proceedings of the VLDB Endowment 2*, 1 (2009), 502–513.

[123] Lian, X., and Chen, L. Probabilistic ranked queries in uncertain databases. In *Proceedings of the 11th International Conference on Extending Database Technology (EDBT), Nantes, France* (2008), pp. 511–522.

[124] Lin, C. X., Zhao, B., Mei, Q., and Han, J. Pet: a statistical model for popular events tracking in social communities. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), ACM, pp. 929–938.

[125] Liu, X., Lin, K. K., Andersen, B., and Rattray, M. Including probe-level uncertainty in model-based gene expression clustering. *Bioinformatics, 8:98* (2007).

[126] Loper, E., and Bird, S. Nltk: The natural language toolkit. In *Proc. of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1* (2002), pp. 63–70.

[127] Lu, Y., and Shahabi, C. An arc orienteering algorithm to find the most scenic path on a large-scale road network. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2015), GIS '15, ACM, pp. 46:1–46:10.

[128] Lv, M., Chen, L., and Chen, G. Discovering personally semantic places from gps trajectories. In *ACM CIKM12* (2012), pp. 1552–1556.

[129] Ma, S., Wolfson, O., and Zheng, Y. T-share: A large-scale dynamic taxi ridesharing service. In *Proceeding of International Conference on Data Engineering (ICDE'13)* (2013), pp. 410 – 421.

[130] MacQueen, J. Some methods for classification and analysis of multivariate observations. In *5th Berkeley Symposium on Mathematics, Statistics, and Probabilistics* (1967), vol. 1, pp. 281–297.

[131] Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., and Byers, A. H. Big data: The next frontier for innovation, competition, and productivity. Tech. rep., McKinsey Global Institute, June 2011.

[132] Matos, L., Nune, J., and Trigo, A. Taxi pick-ups route optimization using genetic algorithms. In *Proceedings of the 10th International Conference on Adaptive and Natural Computing Algorithms (ICANNGA'11)* (2011), pp. 410–419.

[133] Mousselly-Sergieh, H., Watzinger, D., Huber, B., Döller, M., Egyed-Zsigmond, E., and Kosch, H. World-wide scale geotagged image dataset for automatic image annotation and reverse geotagging. In *Proc. of the 5th ACM Multimedia Systems Conf.* (2014), pp. 47–52.

[134] Ngai, W. K., Kao, B., Chui, C. K., Cheng, R., Chau, M., and Yip, K. Y. Efficient clustering of uncertain data. In *Data Mining, 2006. ICDM'06. Sixth International Conference on* (2006), IEEE, pp. 436–445.

[135] Ngai, W. K., Kao, B., Chui, C. K., Cheng, R., Chau, M., and Yip, K. Y. Efficient clustering of uncertain data. In *ICDM 2006* (2006), pp. 436–445.

[136] Niedermayer, J., Züfle, A., Emrich, T., Renz, M., Mamoulis, N., Chen, L., and Kriegel, H.-P. Probabilistic nearest neighbor queries on uncertain moving object trajectories. *Proceedings of the VLDB Endowment 7*, 3 (2013), 205–216.

[137] Niedermayer, J., Züfle, A., Emrich, T., Renz, M., Mamoulis, N., Chen, L., and Kriegel, H.-P. Probabilistic nearest neighbor queries on uncertain moving object trajectories. *Proceedings of the VLDB Endowment 7*, 3 (2013), 205–216.

[138] Norris, J. *Markov Chains*. Cambridge Univ. Press, Cambridge, 1998.

[139] Nutanong, S., Zhang, R., Tanin, E., and Kulik, L. The v*-diagram: a query-dependent approach to moving knn queries. *VLDB Endowment 1*, 1 (2008), 1095–1106.

[140] Orenstein, J. A., and Merrett, T. H. A class of data structures for associative searching. In *ACM SIGACT-SIGMOD* (1984), ACM, pp. 181–190.

[141] Palma, A. T., Bogorny, V., Kuijpers, B., and Alvares, L. O. A clustering-based approach for discovering interesting places in trajectories. In *Proc. of the ACM Symp. on Applied Computing* (2008), pp. 863–868.

[142] Parent, C., Spaccapietra, S., Renso, C., Andrienko, G., Andrienko, N., Bogorny, V., Damiani, M. L., Gkoulalas-Divanis, A., Macedo, J., Pelekis, N., Theodoridis, Y., and Yan, Z. Semantic trajectories modeling and analysis. *ACM Comput. Surv. '13 45*, 4 (Aug. 2013), 42:1–42:32.

[143] Patroumpas, K., Papamichalis, M., and Sellis, T. Probabilistic range monitoring of streaming uncertain positions in geosocial networks. In *Proceedings of the 24th International Conference on Scientific and Statistical Database Management* (2012), SSDBM'12, Springer-Verlag, pp. 20–37.

[144] Pei, J., Hua, M., Tao, Y., and Lin, X. Query answering techniques on uncertain and probabilistic data: tutorial summary. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD), Vancouver, BC, Canada* (2008), pp. 1357–1364.

[145] Pfitzner, D., Leibbrandt, R., and Powers, D. Characterization and evaluation of similarity measures for pairs of clusterings. *Knowledge and Information Systems (KAIS) 19*, 3 (2009), 361–394.

[146] Quercia, D., Schifanella, R., and Aiello, L. M. The shortest path to happiness: Recommending beautiful, quiet, and happy routes in the city. *CoRR (abs/1407.1031)* (2014).

[147] Rand, W. M. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association 66*, 336 (1971), 846–850.

[148] Ré, C., Dalvi, N. N., and Suciu, D. Query evaluation on probabilistic databases. *IEEE Data Eng. Bull. 29*, 1 (2006), 25–31.

[149] Renz, M., Cheng, R., Kriegel, H.-P., Züfle, A., and Bernecker, T. Similarity search and mining in uncertain databases. *Proceedings of the VLDB Endowment 3*, 1-2 (2010), 1653–1654.

[150] Richter, K.-F., and Duckham, M. Simplest instructions: Finding easy-to-describe routes for navigation. 274–289.

[151] Ritter, A., Etzioni, O., Clark, S., et al. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (2012), ACM, pp. 1104–1112.

[152] Roussopoulos, N., Kelley, S., and Vincent, F. Nearest neighbor queries. In *ACM SIGMOD* (1995), vol. 24, ACM, pp. 71–79.

[153] Sacharidis, D., and Bouros, P. Routing directions: Keeping it fast and simple. In *Proc. of the 21st ACM Int'l Conf. on Advances in Geographic Information Systems* (2013), pp. 164–173.

[154] Sakaki, T., Okazaki, M., and Matsuo, Y. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web (WWW), Raleigh, NC* (2010), pp. 851–860.

[155] Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., and Sperling, J. Twitterstand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* (2009), ACM, pp. 42–51.

[156] Santos, D. O., and Xavier, E. C. Dynamic taxi and ridesharing: A framework and heuristics for the optimization problem. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)* (2013), pp. 2885–2891.

[157] Sarma, A. D., Benjelloun, O., Halevy, A., and Widom, J. Working models for uncertain data. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE), Atlanta, GA* (2006), p. 7.

[158] Sayyadi, H., Hurst, M., and Maykov, A. Event detection and tracking in social streams. In *ICWSM* (2009).

[159] Schubert, E., Weiler, M., and Kriegel, H.-P. SigniTrend: Scalable detection of emerging topics in textual streams by hashed significance thresholds. In *Proceedings of the 20th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), New York, NY* (2014), pp. 871–880.

[160] Schubert, M., Renz, M., and Kriegel, H.-P. Route skyline queries: a multi-preference path planning approach. In *Proceedings of the 26th International Conference on Data Engineering (ICDE), Long Beach, CA* (2010), pp. 261–272.

[161] Sen, R., and Deshpande, A. Representing and querying correlated tuples in probabilistic databases. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE), Istanbul, Turkey* (2007).

[162] Services, I. G. T. The toxic terabyte: How data-dumping threatens business efficiency. Tech. rep., International Business Machines Corporation, 2006.

[163] Sharifzadeh, M., Kolahdouzan, M., and Shahabi, C. The optimal sequenced route query. *The VLDB Journal 17*, 4 (July 2008), 765–787.

[164] Sharifzadeh, M., and Shahabi, C. Approximate voronoi cell computation on spatial data streams. *VLDB Journal 18*, 1 (2009), 57–75.

[165] Sharifzadeh, M., and Shahabi, C. Vor-tree: R-trees with voronoi diagrams for efficient processing of spatial nearest neighbor queries. *VLDB Endowment 3*, 1-2 (2010), 1231–1242.

[166] Shekelyan, M., Jossé, G., and Schubert, M. Paretoprep: Efficient lower bounds for path skylines and fast path computation. In *Proceedings of the 14th International Symposium on Spatial and Temporal Databases (SSTD), Hong Kong, China* (2015), pp. 40–58.

[167] Shekelyan, M., Jossé, G., Schubert, M., and Kriegel, H.-P. Linear path skyline computation in bicriteria networks. In *Proceedings of the 19th International Conference on Database Systems for Advanced Applications (DASFAA), Bali, Indonesia* (2014), pp. 173–187.

[168] Sim, K., Gopalkrishnan, V., Zimek, A., and Cong, G. A survey on enhanced subspace clustering. *Data Mining and Knowledge Discovery 26*, 2 (2013), 332–397.

[169] Skoumas, G., Pfoser, D., and Kyrillidis, A. On quantifying qualitative geospatial data: A probabilistic approach. In *Proc. of the Second ACM Int'l Workshop on Crowdsourced and Volunteered Geographic Information* (2013), pp. 71–78.

[170] Skoumas, G., Pfoser, D., and Kyrillidis, A. T. Location estimation using crowdsourced geospatial narratives. *CoRR abs/1408.5894* (2014).

[171] Skoumas, G., Schmid, K. A., Jossé, G., Schubert, M., Nascimento, M., Züfle, A., Renz, M., and Pfoser, D. Knowledge-enriched route computation. In *Proceedings of the 14th International Symposium on Spatial and Temporal Databases (SSTD), Hong Kong, China* (2015), pp. 157–176.

[172] Skoumas, G., Schmid, K. A., Jossé, G., Züfle, A., Nascimento, M., Renz, M., and Pfoser, D. Towards knowledge-enriched path computation. In *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM GIS), Dallas, TX* (2014), pp. 485–488.

[173] Skoumas, G., Schmid, K. A., Jossé, G., Züfle, A., Nascimento, M., Renz, M., and Pfoser, D. Towards knowledge-enriched path computation. Tech. rep., LMU, 2014.

[174] Soliman, M. A., and Ilyas, I. F. Ranking with uncertain scores. In *Proceedings of the 25th International Conference on Data Engineering (ICDE), Shanghai, China* (2009), pp. 317–328.

[175] Soliman, M. A., Ilyas, I. F., and Chang, K. C.-C. Top-k query processing in uncertain databases. In *Proceedings of the 23rd International Conference on Data Engineering (ICDE), Istanbul, Turkey* (2007), pp. 896–905.

[176] Souffriau, W., Vansteenwegen, P., Berghe, G. V., and Oudheusden, D. V. The planning of cycle trips in the province of east flanders. *Omega 39*, 2 (2011), 209 – 213.

[177] Spaccapietra, S., and Parent, C. Adding meaning to your steps. In *Proc. of the 30th Int'l Conf. on Conceptual Modeling* (2011), pp. 13–31.

[178] Struyf, A., Hubert, M., and Rousseeuw, P. Clustering in an object-oriented environment. *Journal of Statistical Software 1*, 4 (1997), 1–30.

[179] Sun, L., Cheng, R., Cheung, D. W., and Cheng, J. Mining uncertain data with probabilistic guarantees. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Washington, DC* (2010), pp. 273–282.

[180] Tao, Y., Cheng, R., Xiao, X., Ngai, W. K., Kao, B., and Prabhakar, S. Indexing multi-dimensional uncertain data with arbitrary probability density functions. In *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB), Trondheim, Norway* (2005), pp. 922–933.

[181] Tao, Y., Xiao, X., and Cheng, R. Range search on multidimensional uncertain data. *ACM Transactions on Database Systems (TODS) 32*, 3 (2007), 15.

[182] Trajcevski, G., Tamassia, R., Scheuermann, P., Hartglass, D., and Zamierowski, C. Ranking continuous nearest neighbors for uncertain trajectories. *The VLDB Journal 20*, 5 (2011), 767–791.

[183] Tran, T. T., Peng, L., Li, B., Diao, Y., and Liu, A. Pods: A new model and processing algorithms for uncertain data streams. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 2010), SIGMOD '10, ACM, pp. 159–170.

[184] Unankard, S., Li, X., and Sharaf, M. A. Emerging event detection in social networks with location sensitivity. *World Wide Web 18*, 5 (2015), 1393–1417.

[185] Valiant, L. G. The complexity of enumeration and reliability problems. *SIAM Journal on Computing 8*, 3 (1979), 410–421.

[186] Vansteenwegen, P., Souffriau, W., and Oudheusden, D. V. The orienteering problem: a survey. *Europe Journal of Operational Research 209*, 1 (2011), 1–10.

[187] Verbeeck, C., Vansteenwegen, P., and Aghezzaf, E.-H. An extension of the arc orienteering problem and its application to cycle trip planning. *Transportation Research Part E: Logistics and Transportation Review 68* (2014), 64 – 78.

[188] Wallis, S. Binomial confidence intervals and contingency tests: Mathematical fundamentals and the evaluation of alternative methods. *Journal of Quantitative Linguistics 20*, 3 (2013), 178–208.

[189] Wang, D. Z., Michelakis, E., Garofalakis, M., and Hellerstein, J. M. Bayesstore: managing large, uncertain data repositories with probabilistic graphical models. *Proceedings of the VLDB Endowment 1*, 1 (2008), 340–351.

[190] Weiler, M., Schmid, K. A., Renz, M., and Mamoulis, N. Geo-social co-location mining. In *GeoRich: 2nd International ACM SIGMOD Workshop on Managing and Mining Enriched Geo-Spatial Data Held in Conjunction with SIGMOD 2015, Melbourne, Australia* (2015).

[191] Welford, B. P. Note on a method for calculating corrected sums of squares and products. *Technometrics 4*, 3 (1962), 419–420.

[192] Weng, J., and Lee, B.-S. Event detection in twitter. *ICWSM 11* (2011), 401–408.

[193] West, D. H. D. Updating mean and variance estimates: an improved method. *Communications of the ACM 22*, 9 (1979), 532–535.

[194] Westphal, M., and Renz, J. Evaluating and minimizing ambiguities in qualitative route instructions. In *Proc. of the 19th ACM Int'l Conf. on Advances in Geographic Information Systems* (2011), pp. 171–180.

[195] Wilson, E. B. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association 22* (1927), 209–212.

[196] Yan, Z., Chakraborty, D., Parent, C., Spaccapietra, S., and Aberer, K. Semitri: A framework for semantic annotation of heterogeneous trajectories. In *Proc. of the 14th Int'l Conf. on Extending Database Technology* (2011), pp. 259–270.

[197] Yan, Z., Chakraborty, D., Parent, C., Spaccapietra, S., and Aberer, K. Semantic trajectories: Mobility data computation and annotation. *ACM Trans. Intell. Syst. Technol. 4*, 3 (July 2013), 49:1–49:38.

[198] Yan, Z., Parent, C., Spaccapietra, S., and Chakraborty, D. A hybrid model and computing platform for spatio-semantic trajectories. In *Proc. of the 7th Int'l Conf. on The Semantic Web: Research and Applications (ESWC) '10* (2010), ESWC'10, Springer-Verlag, pp. 60–75.

[199] Yan, Z., Spremic, L., Chakraborty, D., Parent, C., Spaccapietra, S., and Aberer, K. Automatic construction and multi-level visualization of semantic trajectories. In *Proc. of the 18th Int'l Conf. on Advances in Geographic Information Systems* (2010), pp. 524–525.

[200] Yang, D., Zhang, D., Chen, L., and Qu, B. Nationtelescope: Monitoring and visualizing large-scale collective behavior in lbsns. *Journal of Network and Computer Applications 55* (2015), 170–180.

[201] Yang, D., Zhang, D., and Qu, B. Participatory cultural mapping based on collective behavior in location based social networks. *ACM Transactions on Intelligent Systems and Technology* (2015). in press.

[202] Yi, K., Li, F., Kollios, G., and Srivastava, D. Efficient processing of top-k queries in uncertain databases with x-relations. *IEEE Transactions on Knowledge and Data Engineering 20*, 12 (2008), 1669–1682.

[203] Yiu, M. L., Mamoulis, N., Dai, X., Tao, Y., and Vaitis, M. Efficient evaluation of probabilistic advanced spatial queries on existentially uncertain data. *IEEE Transactions on Knowledge and Data Engineering 21*, 1 (2009), 108–122.

[204] Yuan, J., Zheng, Y., Xie, X., and Sun, G. Driving with knowledge from the physical world. In *SIGKDD* (2011), pp. 316–324.

[205] Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G., and Huang, Y. T-drive: Driving directions based on taxi trajectories. In *SIGSPATIAL* (2010), pp. 99–108.

[206] Zhang, P., Cheng, R., Mamoulis, N., Renz, M., Züfle, A., Tang, Y., and Emrich, T. Voronoi-based nearest neighbor search for multi-dimensional uncertain databases. In *Proceedings of the 29th International Conference on Data Engineering (ICDE), Brisbane, Australia* (2013), pp. 158–169.

[207] Zheng, B., Xu, J., Lee, W.-C., and Lee, L. Grid-partition index: a hybrid method for nearest-neighbor queries in wireless location-based services. *VLDB Journal 15*, 1 (2006), 21–39.

[208] Zhou, X., and Chen, L. Event detection over twitter social media streams. *The VLDB journal 23*, 3 (2014), 381–400.

[209] Zimányi, E. Query evaluation in probabilistic relational databases. *Theor. Comput. Sci. 171*, 1-2 (Jan. 1997), 179–219.

[210] Zimek, A., and Vreeken, J. The blind men and the elephant: On meeting the problem of multiple truths in data from clustering and pattern mining perspectives. *Machine Learning 98*, 1–2 (2015), 121–155.

[211] Züfle, A., Emrich, T., Schmid, K. A., Mamoulis, N., Zimek, A., and Renz, M. Representative clustering of uncertain data. In *Proceedings of the 20th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), New York, NY* (2014), pp. 243–252.

[212] Zwillinger, D., and Kokoska, S. *CRC Standard Probability and Statistics Tables and Formulae*. CRC Press, 2000.