FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Sports Activity Suggestions: A Visual Analytics Approach

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Software Engineering & Internet Computing

by

B.Sc. Anca Cismasiu
Registration Number 0727280

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec. Silvia Miksch
Assistance: M.Sc. Albert Amor-Amorós

Vienna, 1st September, 2018

Anca Cismasiu	Silvia Miksch

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Vorschläge für Sportaktivitäten: Ein Visual Analytics Ansatz

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

B.Sc. Anca Cismasiu
Matrikelnummer 0727280

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec. Silvia Miksch
Mitwirkung: M.Sc. Albert Amor-Amorós

Wien, 1. September 2018

Anca Cismasiu	Silvia Miksch

Erklärung zur Verfassung der Arbeit

B.Sc. Anca Cismasiu
Theresianumgasse 14/4, 1040, Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 1. September 2018

Anca Cismasiu

Abstract

As companies gather more and more data, we need to find a way to allow interested decision makers to access this data in an efficient way. In the context of sports practice, users could benefit from suggestions about new sports they could try out and the company could increase its sales. This work aims to support the analysts, simultaneously domain experts and IT laymen, in their data exploration and suggestion retrieval tasks through a user friendly interface, abstracting away the complexity of formulating expressive queries into the visual domain.

We present a characterization and task analysis for this domain, and a prototype that meets the requirements emerging from them, based on an interdisciplinary literature research.

The resulting prototype combines a visual query language with a collaborative filtering approach to render suggestions for new activities, and show multiple types of relationships in a visually compelling way. It has been implemented as a web application that handles the transformation of user input from a graphical pattern into a database query language and the results of this query into an easy to digest information representation.

We conclude with an expert interview to validate the design for analysis and exploration.

Kurzfassung

Während Unternehmen immer mehr Daten sammeln, brauchen interessierte Entscheidungsträger eine Möglichkeit, auf diese Daten effizient zuzugreifen. In dem Kontext einer Sportbuchungsplattform, könnten User über Vorschläge neuer Sportarten neue Interessen entdecken und das Unternehmen könnte den Umsatz erhöhen.

Das Ziel dieser Arbeit ist die Analysten zu unterstützen, da sie gleichzeitig Domänenexperten und IT-Laien sind. Ein benutzerfreundliches Interface, das die Komplexität der Formulierung von Abfragen wegabstrahiert, soll ihnen in ihren Datenexplorations- und Vorschlagsfindungsaufgaben unterstützen.

Wir präsentieren eine Beschreibung der vorliegenden Domäne und deren Aufgaben und ein Prototyp das die hervorgehenden Anforderungen erfüllt, basierend auf einer interdisziplinären Literaturrecherche.

Das resultierende Prototyp kombiniert eine visuelle Abfragesprache mit einem kollaborativen Filtern Ansatz für das Finden der neuen Vorschläge. Die unterschiedlichen Beziehungen zwischen den Daten werden in einer optisch ansprechenden Form angezeigt.

Das Prototyp wurde als Webapplikation entwickelt, das ein visuelles Muster als Input akzeptiert, dieses in eine Graphdatenbankabfrage umwandelt, die Ergebnisse davon dann wieder als leicht verständliche Visualisierung anzeigt.

Die Validierung des Designs für Analyse und Exploration wurde durch ein Experteninterview durchgeführt.

Contents

List of Figures

CHAPTER 1

Introduction

Many people play sports as a hobby, in a more or less organized and consistent fashion, ranging from a spontaneous game of beach volleyball between friends, to playing tennis every other week, or to attending camps, classes and workshops. The logs of an online sports venue booking platform that facilitates such activities hold information of their users' interests. This information could be leveraged to provide suggestions to the users of such platforms regarding new events they might be interested in, with the goal of sparking people's interest in trying new, different types of sport. In order to be able to make appropriate and meaningful suggestions, several aspects need to be taken into account:

Sports and Event Type
: The different sport types can be classified based on a large number of objective, inherent characteristics (racquet sports, indoor or outdoor, team or single player, winter or summer sport, etc.). By factoring in these objective properties, some sports might seem very similar. The question that needs to be answered is whether these perceived similarities translate to user preferences, and whether the users' interest can be sparked to take up new "similar" sports. For example, would an indoor, 6-person team volleyball player be more interested in beach volleyball or handball? Is it safe to assume that someone that enjoys canoeing will be interested in yachting as well, seeing as they are both boat sports?

Social Context
: With some exceptions, playing sports is a social activity. People in classes, camps and clubs play with the same teammates for extended periods of time, under the supervision of a trainer. Others still play with the same partner and/or against the same opponent(s) in less organized settings, choosing them from their friends or acquaintances. This is often the case in football, tennis or one-on-one sports like

squash. This social dimension can be used to identify groups of users and leverage the relationships among them.

Temporal Context

In addition to this relational aspect of the data, there is also a temporal one that needs to be explored. For example, do users prefer certain times of year or days of the week for different sports?

Another interesting aspect that prompts exploratory analysis is the seasonality of some sports like beach tennis or skiing, as it forces the users to either switch to available sports or to stop being active until their preferred activity is in season again. These users could be incentivized to take up a new, similar sport in the meantime.

In order to extract the information contained in these large amounts of raw data, we need to tackle the information overload problem, as to not get lost in the vast amounts of potentially disparate and conflicting data (Keim et al. [KAF+08]). Different approaches are appropriate, depending on how well the problems are defined and understood. Fully automated data analysis works well when the problems are well-defined and well-understood. As our tasks are not so well-defined, we chose a Visual Analytics approach. Visual Analytics is a young, multidisciplinary field, that Cook et al. [CT05] define as "the science of analytical reasoning facilitated by interactive visual interfaces". This semi-automated analytic approach lets humans and machines combine forces and divide labor along their respective distinct strengths.

We try to follow Keim's Visual Analytics mantra of *"Analyze first, Show the Important, Zoom, filter and analyze further, Details on demand"*. This is an adjustment to Shneiderman's *"Overview first, Filter and zoom, Details on demand"* [Shn96], that shifts the focus from visualization to analytics in the knowledge discovery process.

In order to find new activities for a cohort of users, we propose a visual query language based on a graph database, that enables analysts to construct expressive queries, without the need to write complex queries directly.

The advantage of using graphs and graph databases is that pretty much any domain can be expressed as entities and relationships. Because of this, graphs are used in a variety of domains for structural modeling as they "turn semantic proximity into topological connectivity" (Keim et al. [KAF+08]). Domains that are commonly modeled as graphs include social networks, chemical bonds, fraud detection, real-time recommendation engines, network and IT operations. Graphs are a natural way to model this type of data, because the queries usually rely on exploring and exploiting the relationships. Query performance and scalability is greatly improved as queries don't rely on joins, like in relational database systems. Graph databases store relationships as first-class entities on insert, which means that deep and complex queries are reduced to walking the graph data following these direct connections between nodes instead of computing expensive

join index lookups. This allows for the traversal of millions of nodes per second and super fast response times.

We chose Neo4J as our graph database for a number of reasons. First, it's open source. Second, Neo4J uses a native graph storage, unlike some other graph database systems that use a much slower relational or object-oriented database in the background. Third, Neo4J ensures data integrity through ACID compliance. Fourth, Neo4J comes bundled with its own query language Cypher, which is expressive, powerful and easy to use.

1.1 Main Research Question

Our main research question is

> **How can Visual Analytics support the discovery of alternative and complementary sporting activities?**

and is based on following **hypotheses**:

H1 Interactive visual querying can be an effective tool to find alternative and complementary sporting activities.

H2 More insight can be provided by taking into account both the relational, as well as the temporal aspects of the data.

H3 Additionally, interactive methods of the visualization can also help the analyst assess the value of these suggestions.

1.2 Methodology

This paper is problem driven research, that focuses on solving users' real world problems. We found that the information is mostly in the computer, while the semantics and domain knowledge is still in the analyst's head, allowing them to filter out false positives, such as sports that only seem similar when observing the category links and the structure of the classification graph, but that fail to meet other extrinsic criteria (e.g., being in same price range, having a different target group).

By examining the task clarity and information location axes in Figure 1.1, we found that the task clarity falls somewhere in the middle of the axis. Some tasks are defined, but there is an exploration part, which should support gaining new insights into the data. Consequently, we found it appropriate to follow a methodology roughly based on the nine-stage framework proposed by Sedlmair et al. [SMM12].

As described in this model, there are nine stages, grouped into three phases.

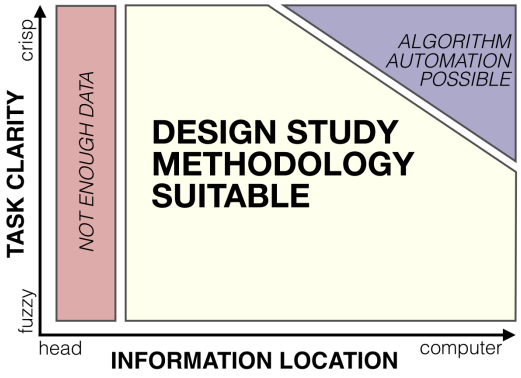


Figure 1.1: The task clarity and information location axes from Sedlmair et al. [SMM12], as a way to analyze the suitability of design study methodology.

I. Precondition Phase

1. Learn - Consult Visualization Literature
2. Winnow - Select Promising Collaborators
3. Cast - Identify collaborator roles

II. Core Phase

4. Discover
   - Characterize the problems and data of the domain at hand
   - Define the tasks
   - Determine who or what defines the "relevance of information" for a given task
   - Abstract Operations
   - Gather data from its (possibly) heterogeneous sources
   - Transform and clean the data in order to bring it to a consistent state and format, which is a precondition to performing any kind of analysis
5. Design
   - Abstract the Data
   - Design the visual encoding
   - Design the interaction
6. Implement Prototype
7. Deploy

III. Analysis Phase

8. Reflect
9. Write

As needed, there was less focus on preconditions, like winnow and cast, with more emphasis on the core phase. The thesis structure follows the methodological approach, which consists of following steps:

1. **Literature review - Chapter 2:** provide necessary background and concepts, get an overview of the topic and its many related fields and identify the possible approaches to the problem
2. **Problem characterization - Chapter 3:** requirement analysis to identify users and tasks, to determine how to support them in fulfilling their needs
3. **Design - Chapter 4:** design the interactive visualization by choosing a data abstraction, visual encodings and interactions
4. **Implement prototype - Chapter 5:** implementation of a working prototype as a web application
5. **Evaluation - Chapter 6:** validate if the chosen approach is suitable to answer the research questions

CHAPTER 2

Related Work

In this chapter we explain the background and important concepts for this thesis, as well as give an overview of the current state of the art of the fields relevant to our project like large data visualization, visual graph query languages and time and temporal event sequence visualizations.

Information Visualization, often abbreviated "InfoVis", is defined by Card et al. [CMS99] as *"the use of computer-supported, interactive visual representations of data to amplify cognition"* while Keim et al. [KMSZ06] define it as *"communication of abstract data relevant in terms of action through the use of interactive visual interfaces"*. A related field that also deals with visual representations of data is Scientific Visualization. The key difference here being that Scientific visualization handles data sets captured from the real world, from sensors, simulations or laboratory tests, that have a given spatialization. It has a limited set of application domains and a smaller design space, like medical imaging, satellite photograph processing or 3D-rendering.

Information Visualization is more general, and seeks to communicate abstract data through interaction to aid humans in task solving. Keim et al. [KMSZ06] explain the major goals of InfoVis as threefold:

- **presentation** communicate the results efficiently and effectively
- **confirmatory analysis** serves to confirm or reject an existing hypothesis regarding the data
- **exploratory analysis** search the data for structures and trends without a starting hypothesis

Visual Analytics is a young, multidisciplinary field that combines the strength of humans and computers, and it is both user-driven and data-driven. Keim et al. [KAF+08]

define it as *"Visual analytics combines automated analysis techniques with interactive visualizations for an effective understanding, reasoning and decision making on the basis of very large and complex data sets."* and Thomas et al. [TC06] outline its main focus areas as

- analytical reasoning techniques
- visual representations and interaction techniques
- data representations and transformations
- techniques to support production, presentation, and dissemination of analytical results

The paper intersects multiple research areas, each having a large body of relevant work. We briefly describe relevant work in some of the related areas.

2.1 Large data visualization

The ideas from the fundamental article by Shneiderman [Shn96] are still very much valid and relevant today, 20 years after its publication. The information overload is even more an issue now, since the volume of data produced, logged and processed only went up. To mitigate information overload, the paper introduces the Visual Information Seeking Mantra, "Overview first, zoom and filter, then details on demand", to serve as a guide for the design of graphical user interfaces for exploration. In defining a task-by-data-type taxonomy for 7 data types(1-, 2-, 3-dimensional data, temporal and multi-dimensional data, and tree and network data), the paper also identifies 7 high-level abstract user tasks needed for their exploration:

- Overview: Gain an overview of the entire collection
- Zoom: Zoom in on items of interest
- Filter: Filter out uninteresting items
- Details-on-demand: Select an item or group and get the details when needed
- Relate: View relationships among items
- History: Keep a history of actions to support undo, replay, and progressive refinement
- Extract: Allow extraction of sub-collections and of query parameters

The paper highlights the reason why translating advanced filtering queries from the user's natural language to the system is a complex matter. For a dynamic query approach, where user interface elements like sliders, buttons, table views, etc., that correspond to data properties are directly manipulated, the advantages are the rapid, incremental changes with immediate feedback and no error messages, but as the data set grows, we run into issues relating to scalability and searching of the data space.

Another important aspect of translating complex user queries into Boolean expressions is the difference in meaning between natural language "AND", "OR" and their logical equivalents. While in natural language "AND" is mostly understood as a union of sets, satisfying either of the conditions ("People that play tennis and people that play squash"), Boolean "AND" denotes the intersection, which would result only in the items satisfying all. Similarly, natural language "OR" is usually taken to mean a Boolean "XOR" ("Play tennis or squash on Monday").

A "water flow" pipes-and-filter graphical metaphor is proposed to manage this query complexity and be able to offer full Boolean expressions, together with nested parentheses and negations. "AND"-operators are filters that narrow the "water" flow, while "OR"-operators laid out in parallel merge the streams. "NOT"-operators invert the selection. One example of such a visualization is a Sankey diagram, as exemplified in Figure 2.1

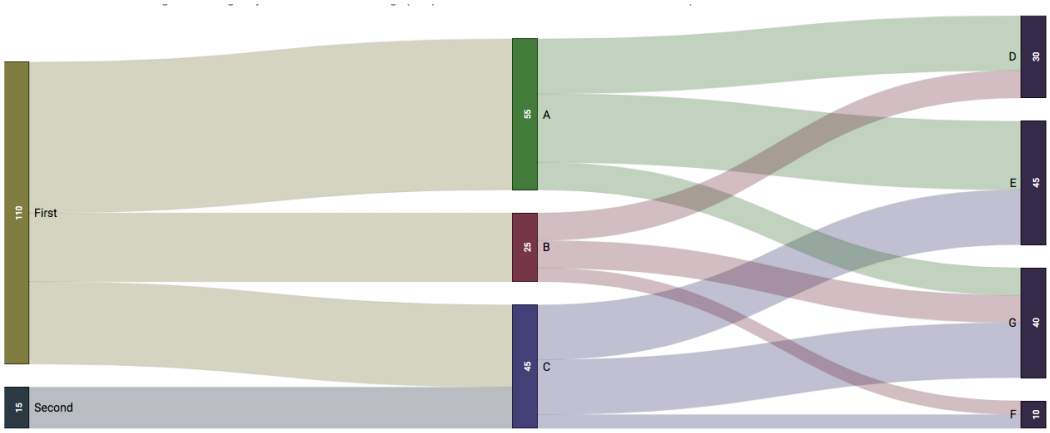


Figure 2.1: An example of a Sankey diagram, which is a flow visualization, based on a pipes-and-filter graphical metaphor

Other approaches have their merits, based on the use case, such as APOLO, depicted in Figure 2.2 and described in Chau et al. [CKHF11], which takes a different visualization approach in order to make sense of large network data. Its core idea is to start small and make the exploration user-driven, instead of data-driven, leaving the structure of the data to act solely as a guide for the exploration. This way the user can build a mental model for himself, being supported by a rich user interface and visualizations on top of machine learning.

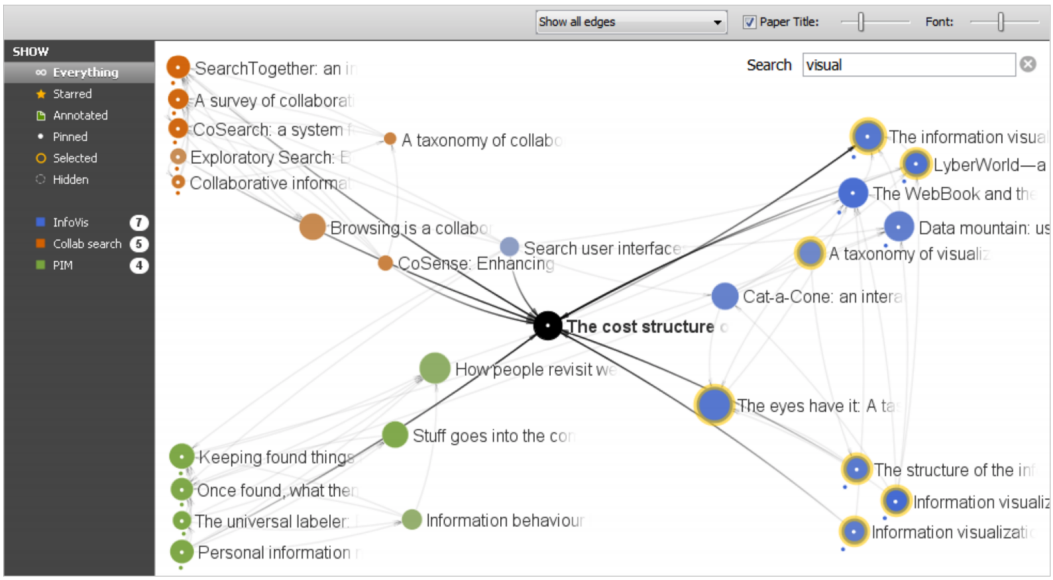


Figure 2.2: Screenshot of APOLO: User exploring the landscape of research around a the article highlighted in black, giving relevance feedback which papers he's interested in (color dot underneath) and how they should be categorized (color) [CKHF11]

2.2 Visual graph query languages

Visual Query Languages are used to extract information from databases by using a direct manipulation paradigm of visual representations of the underlying database, while striving to achieve the same semantic expressiveness as textual query languages. Graph query languages can be divided into graph traversal and graph pattern matching.

In pattern matching, like SPARQL, we check if the sequence of tokens contains a pattern (a sequence or tree structure) and return the exact match. These patterns are defined using regular expressions and matched by backtracking.

Graph pattern matching is more common than graph traversal. GRAPHITE, in Chau et al. [CFT+08] uses G-Ray algorithm for approximate subgraph matching in attributed graphs. The user can visually construct query patterns of arbitrary shapes by drawing them and assigning values to node attributes, and the system retrieves both exact and approximate matches, ranking them by quality criteria.

QGraph, described in Blau et al. [BIJ02] is a visual language for querying and updating graph databases, in which the user can draw a query consisting of some vertices and edges with specified relations between their attributes and cardinality. The response will be the collection of all subgraphs of the database that have the desired pattern.

On the other hand, graph traversal is procedural and allows for recursion.

An example of graph traversal is visKWQL in Hartl et al. [HWB10], which is a visual alternative to the textual KWQL, used for querying semantic wikis. In this form based approach, resources, qualifiers, and operators are represented as boxes, and resource-value or qualifier-value associations are represented as nestings.

Shadoan et al. [SWS13] observe the increasing difficulty of visual exploration and analysis brought forth by increasing data dimensionality and approach visual querying of n-ary relationships by constructing them as hypergraphs, where nodes correspond to a subset of values and hyperedges to conjunctive relationships, as shown in Figure 2.3.

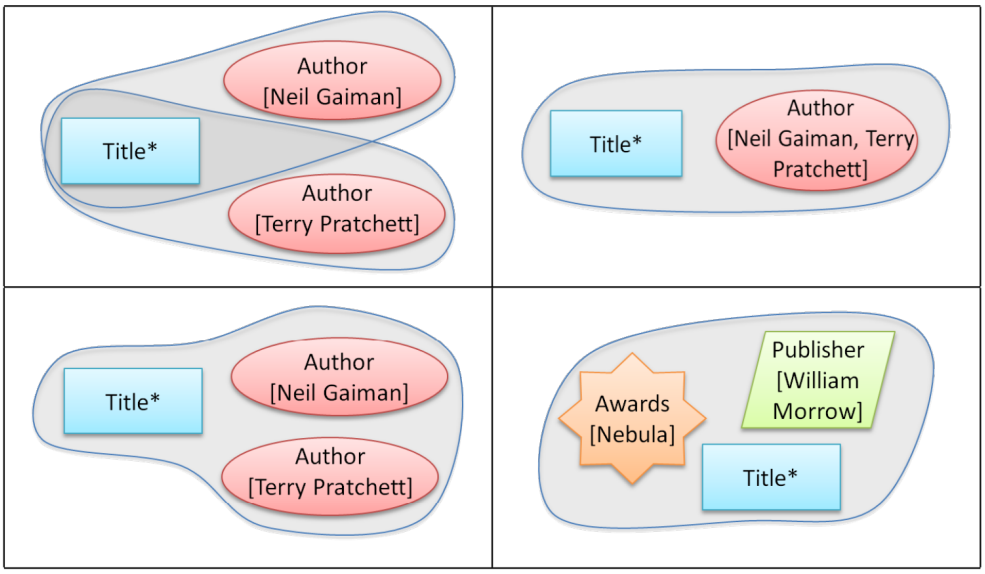


Figure 2.3: Conjunctive queries using hyperedges.
TOP: equivalent OR-queries "Which books were written by Neil Gaiman OR Terry Pratchett?".
BOTTOM LEFT: AND-query on the same attribute “Which books were written by Neil Gaiman AND Terry Pratchett?”
BOTTOM RIGHT: AND-query on different attributes “Which books published by William Morrow won Nebula awards?”

The visual query language presented is expressive, as it also supports queries with nested hyperedges, but this approach works best when the user already knows what they’re looking for, and we need a more accessible exploration technique.

The interactive visual tool that uses this query language, Candid, allows for simple interactions to build and modify queries dynamically. There is one UI window, with a panel for each relevant dimension, where the values are listed as a table. The explanation of the query building process interactions relies on Figure 2.4: a user successively selects the desired subset of values for nodes (1) and the type (2) and adds them to the “Query

graph" panel. The nodes are then connected with a hyperedge (3). When run (4), the query produces a result in graph form in the "Attribute Relationship Graph" panel. Nodes are connected if there is at least one data item that satisfies the attribute conditions. The edge width is proportional to the number of items. The edges can be adjusted (toggled) (5).

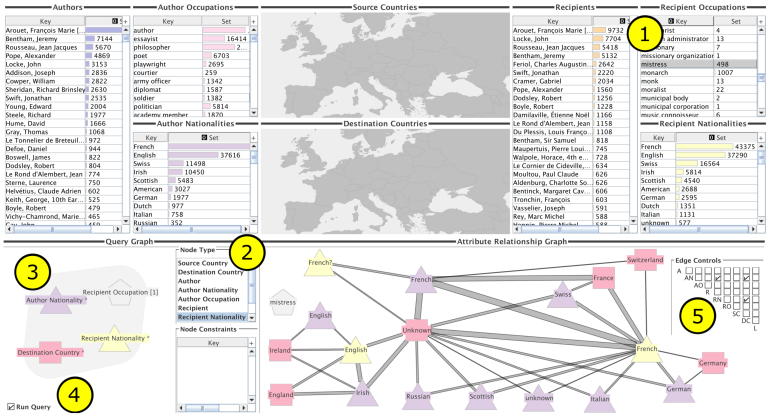


Figure 2.4: Display of Candid query results regarding the nationality and location of mistresses in the Electronic Enlightenment data set. The process is (1) user successively selects the desired subset of values for nodes (recipient occupation "mistress"), (2) then the type of nodes, so the user adds nodes representing any recipient nationality, author nationality, or destination country, (3) the nodes are then connected by a hyperedge (4) the query is run (5) when the results are shown, the user adjusts the edges
from Shadoan et al. [SWS13]

The tool combines cross filtering with an attribute-relationship graph to mitigate the disadvantages of both: cross-filtering is most useful for exploring one-to-one / one-to many relationships, and attribute-relationship graphs are visually overwhelming on their own, as they employ one node per attribute value and edges connecting them.

Cross filtering is a design pattern that uses multiple coordinated views to analyze multidimensional data. Data is partitioned based on its dimensions, and each dimension of interest gets its own view. The users can then filter and select desired values for attributes in these views, which triggers a filtering of the values for the other dimensions. Attribute relationship graphs are undirected graphs, where nodes represent unique attribute values that are connected to data points that exhibit these values. The combination approach is to cross filter the attribute relationship graph and visualize the result, so only connected nodes to data that match are shown.

Kojaph in Didimo et al. [DGM15] is a visual query language for graph databases where complex queries are defined through a mixture of interactions. As the usage scenario of the tool is querying, not data exploration, it doesn't follow the Visual Information

Seeking Mantra, but has a pattern specification approach. The web interface is split into the query window and the result window, and doesn't provide an initial overview of data. This lack of context means that the user has to know what they're looking for. The user interface has some drawbacks: it relies solely on clicks for interaction and doesn't seem to be built with the average user in mind, who is not familiar with graph databases, and adding constraints to the property tree is unintuitive, as it's unclear in which order the buttons should be clicked to add a constraint. The UI is also restricted to constructing a query by first choosing a set of criteria which is sent to the database and waiting for the result to be displayed, which makes incremental querying and data exploration impossible and makes it only accessible to advanced users that know what they are looking for.

For our visual graph querying use case, we need an "overview first, then filter" approach to specify a query, because the analyst may have no expectation of data properties. An overview gives him a chance to observe "interesting" patterns, whether that may mean, frequent, infrequent, novel, etc and then filter on demand. A dynamic, incremental querying approach is also indispensable for data exploration and analysis.

2.3 Time and Temporal event sequence visualization

There are different visualization techniques for visualizing time-oriented data, where the focus is on the representation of qualitative and quantitative data attributes with a simple time axis. But when visualizing time per se, then the focus is on the structure and characteristics of time itself, with a simple data representation.

The "pipe-and-filter" approach is often encountered in visualizing temporal event sequences as flows, like in Outflow (Wongsuphasawat et al.[WG12]), Frequence (Perer et al. [PW14]) and COQUITO (Krause et al. [KPS15]). COQUITO is a visual tool for analyzing disease progression, where the user first defines patient cohorts (groups of individuals that display some common features) visually, and can then analyze their behavior using temporal queries against an event database. In the case of disease progression patterns, the order and timing of symptom occurrence is very important. The temporal patterns can be explored through iterative querying, intermediate results are displayed, as well as hints to the database contents. In our case, the order in which events occur is less relevant, but the visual representation used here is interesting. As shown in Figure 2.5, starting from the junction on the left, which represents all patients, successive filters are applied to obtain the cohort at the rightmost junction.

The user interface provides the user with valuable feedback: the temporal queries are shown as links with labels between the junctions, whose radius and color saturation is log-scale proportional to the number of data points that satisfy the constraints. The junctions also have labels for the events and data set cardinality. Users can choose constraints from either of the complimentary visualization panels: a text search panel with real time event filtering or the treemaps showing the event distributions that are updated after each junction selection.

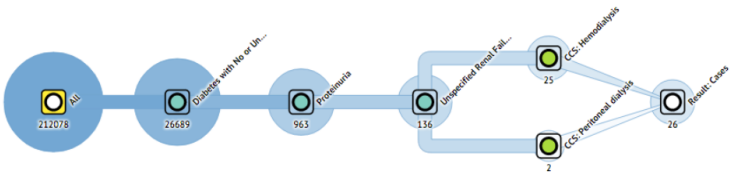


Figure 2.5: Coquito (Krause et al. [KPS15]): Defining a cohort as a set of patients with a diagnosis of Diabetes, followed by Proteinuria, then by Unspecified Renal Failure, that then had Hemodialysis and/or Peritoneal dialysis performed.

The visual queries are built using Drag&Drop of these event constraints. Clicking on a route opens an input box for adding a time window constraint for the occurrences of the events that it joins.

The queries can express AND-operators by adding several events in the same junction, OR- operators through parallel paths, and NOT-operators by a red outline, but other analyses are supported through different Drag&Drop interactions, for example, to compare junctions, which translates to a data set intersection, a junction can be dragged and dropped on top of another.

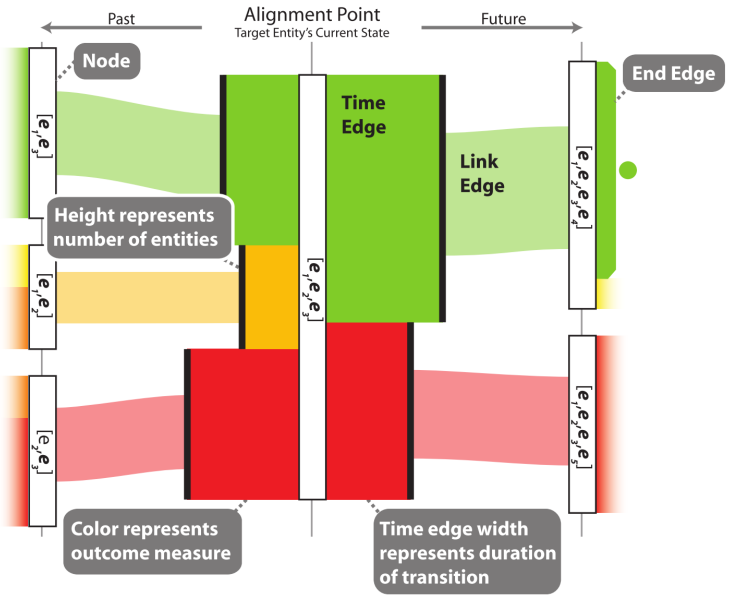


Figure 2.6: Visual encoding in Outflow (Wongsuphasawat et al.[WG12])

Outflow and Frequence are both visualizations based on Sankey diagrams (usually found in flow visualizations) but use different approaches to visualize event sequences.

Outflow is a flow-based visualization for aggregated event progression pathways. If the outcomes are measurable, like win/loss, life/death, then they will be connected to the different pathways, and the behavior is analyzed as a cumulative flow. It uses a combination of state transition graphs, modified to allow modeling of transition time, and Sankey diagrams. For an example from the medical domain, the flow symptom A —> symptom B means that the patient went from having no symptoms, to having symptom A, to having both symptom A and B.

For disease evolution patterns, the order and time of onset of different symptoms is relevant and correlates with patient outcome. For our case, the order of events is irrelevant, and timing is only partially interesting, when it shows seasonality or a certain recurrence.

Figure 2.7: Interactive exploration of the aggregate event progression pathways with associated statistics in Outflow (Wongsuphasawat et al.[WG12])

Nevertheless, the Outflow visualization and its interaction techniques are interesting. It uses a combination of state transition graphs, modified to allow modeling of transition time, and Sankey diagrams, usually found in flow visualizations. As depicted in Figure 2.6, the states are modeled as rectangles, with a height proportional to the data set size, that are aligned vertically, according to time, past (left) to present(right). Transitions are drawn as cubic Bézier curves with a time edge, where the width is proportional to the average time gap, and the height proportional to the entity count. The outcomes are color coded.

Data exploration is made possible not only through the interactions available (Panning&Zooming, Event Type Selection, Filtering, Brushing, Hovering and Tooltips, Pinning) but also because of the simplification algorithm, which hierarchically clusters similar

states in the same vertical level to reduce visual clutter (Figure 2.7) For our domain, such an aggregation approach is not ideal, as it excludes outliers and seldom occurring patterns, even if they might be of interest. Another limitation to the applicability to our domain is that this technique doesn't incorporate external factors, for example, medication given at certain stages of the disease that influence the event progression pathways. In our case, external factors are if the sport is in season, but also the social relationships (the decision to book a certain course is influenced by it being lead by a trainer the player knows beforehand).

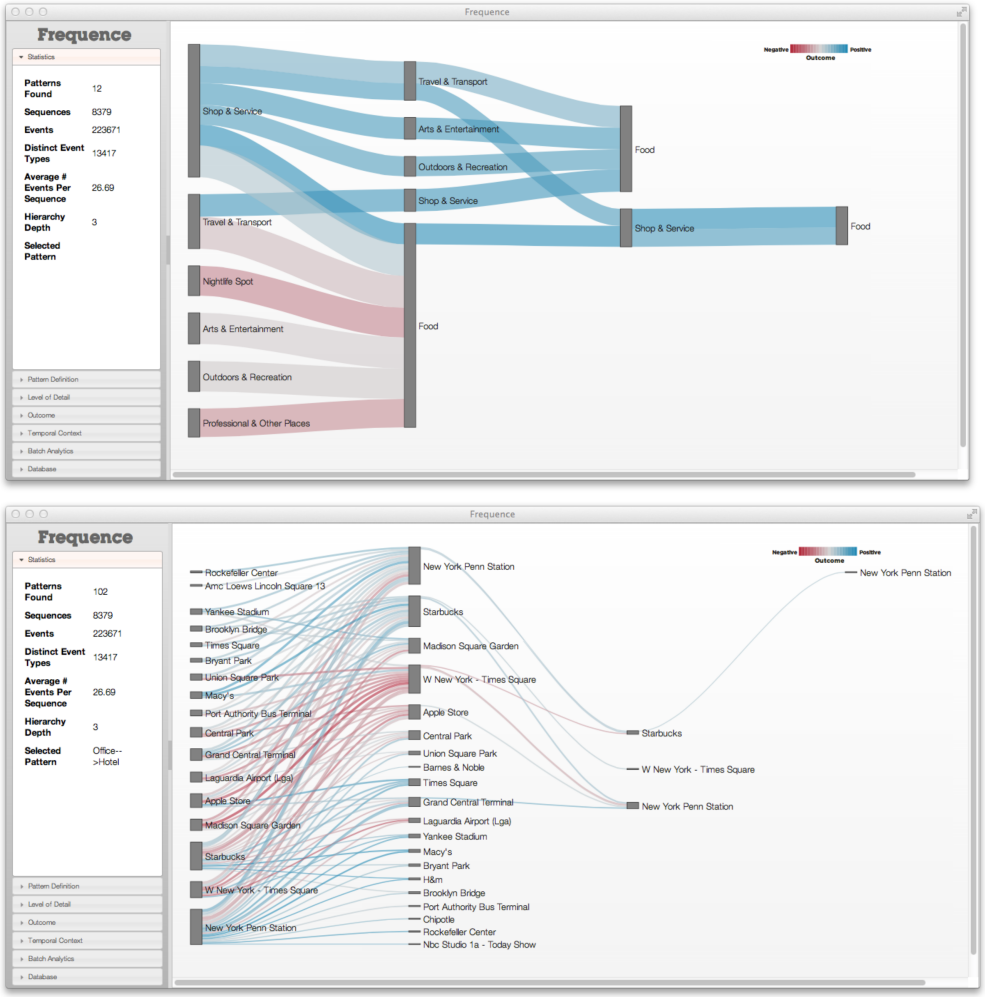


Figure 2.8: Overview vs. most detailed view in Frequence (Perer et al. [PW14]). This follows the Visual Information Seeking Mantra

Frequence, in Perer et al. [PW14] is an approach that combines frequent sequence mining with an interactive visualization to find frequent temporal event sequences, for example

for understanding progression of diseases among patients. It is a flow visualization inspired by Sankey diagrams and by alluvial diagrams. Alluvial diagrams are usually used to show how networks change over time. It follows the Visual Information Seeking Mantra, to start with an overview of frequent patterns at a coarse level of detail, and then zoom in or filter, as seen in Figure 2.8.

These approaches are interesting and could, to an extent, be extrapolated and applied to our use case, but they are missing support for the network aspect of both the users and the sport types. Another interesting approach is presented in Vrotsou et al. [VJC09] as ActiviTree, a tool for interactive visual exploration of event sequences that uses a tree like structure and a matrix similarity algorithm, although it has very limited querying expressivity.

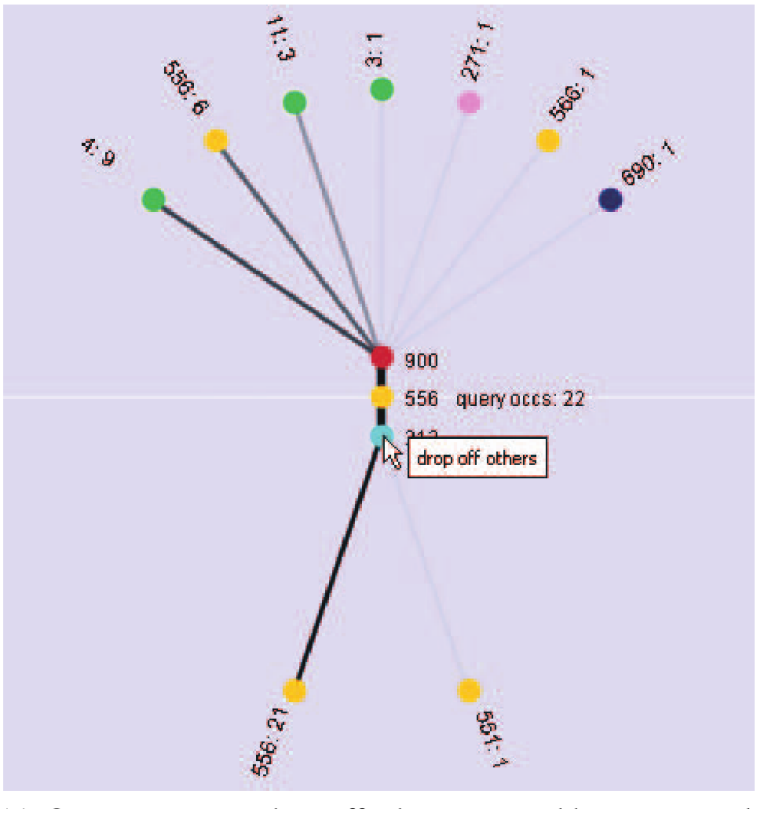


Figure 2.9: Exploration in ActiviTree (Vrotsou et al. [VJC09]) Query sequence drop off others -> travel by car -> work

The data is encoded as a directed graph, where nodes are activities and edges show the transitions between them. Users define sequences stepwise, starting from a significant event, incrementing the sequence by one event in each step, linking the next by choosing a branch. In each step, the significance of other events is calculated by using a matrix

similarity algorithm and next events are shown as in/out branches. The events have similarity scores, that can be based on connectivity, frequency and/or other factors, and users can tune the weights to reflect the sequence they consider to be important. Figure 2.9 shows the query sequence "drop off others -> travel by car -> work".

Users can also weight the significance of events in the analysis, and influence the matrix similarity algorithm in this way. Because the visualization relies on a linear temporal ordering of events to build the patterns, and the created path is the query, dynamic query modification is limited to adding and removing events to and from either end. An advantage of expanding the on user interaction is that it ensures that even unusual patterns, not just frequent ones, can be found.

When visualizing time-oriented data, we need to differentiate between different "types of time". This allows us to then choose an appropriate visualization for the given data and time characteristics, and to finally make a meaningful visual analysis. Aigner et al. [AMM+08] describe the most important criteria for types of time:

- linear time vs. cyclic time While linear time has a starting point and a linear domain (from past to future), the ordering of points of a cyclic domain is meaningless with respect to a cycle (Monday before Tuesday but also Tuesday before Monday). For cyclic time, a Spiral Graph visualization Weber et al. [WAM01], when correctly parameterized, makes the periodic pattern stand out.

- time points vs. time intervals Discrete time points are an abstraction and have no duration, and time interval data points are defined for a duration between two time points.

  TimeWheel, in Tominski et al. [TAS04] a multi-axis visualization of multivariate data over time is a good example of a visualization for point-based time. As seen in Figure 2.10, the time axis is in the center, with axes for the other attributes arranged circularly around it. For each data point, a line is drawn to connect the values on the time and variable axes.

- ordered time vs. branching time vs. time with multiple perspectives While events happen one after another in ordered time, branching time allows for description and comparison of different scenarios in multiple time threads, and time with multiple perspectives allows more than one point of view for the same situation. Most techniques are suited for ordered time only, such as ThemeRiver Havre et al. [HHWN02], which visualizes thematic variations over time using a river metaphor. A wide current in the river indicates heavy use of a topic, while changes in color distribution correlate to changes in themes as in Figure 2.11.

For time-oriented data with periodic behavior, it is important to choose a visualization that highlights patterns, so that trends and anomalies can be easily discovered. This helps users confirm or refute their hypotheses with one glance, but also lead to new insights

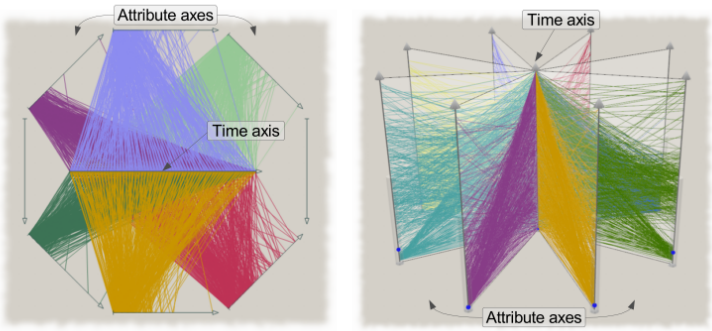


Figure 2.10: 2D and 3D view of the TimeWheel technique(Tominski et al. [TAS04]), figure from Aigner et al. [AMM+08]

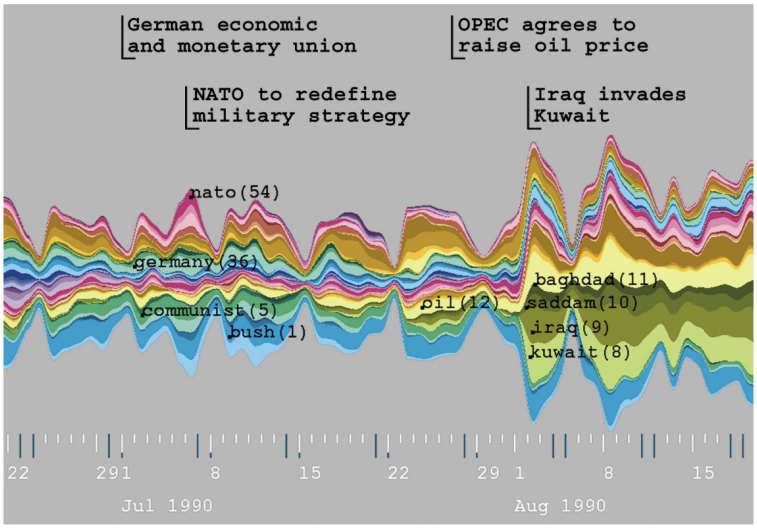


Figure 2.11: ThemeRiver Havre et al. [HHWN02], representing AP data from July - August 1990

and foster knowledge discovery. For the identification of periodic structures, Weber et al. [WAM01] propose a visualization using Spiral Graphs (Figure 2.12) The visualization metaphor is based on Archimedes spiral and is based on a spirally shaped time axis. This representation captures the continuity of data and is well suited to humans' ability to detect structures, and supports easy identification and confirmation of periodicity. To make periodic behavior apparent, the cycle length has to be chosen appropriately, so that data with the same phase have the same phase in the spiral, for example, if a daily period is expected, one cycle should represent 24 hours.

Quantitative properties of the data are usually encoded as color and thickness of the line, and this allows for comparison of data sets can be done by using intertwined spirals of the same cycle length, with different data encoding (color, texture, etc).

For very large data sets, where the spiral wouldn't fit on the screen, the data set can be mapped to a 3D helix instead. A subset of the data can then be chosen to be visualized as a 2D spiral. This approach limits the ability to identify periodic behavior, as the user can't see full cycles, it is more of a navigation tool.

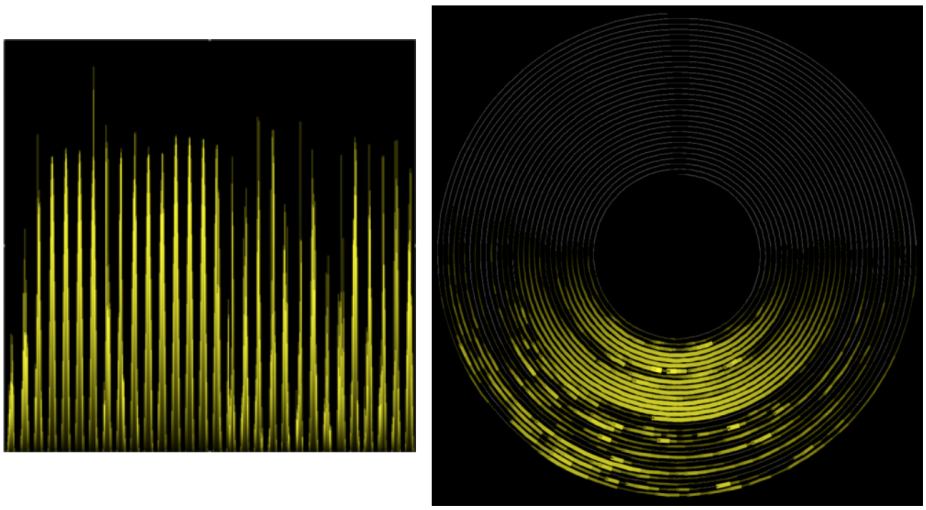


Figure 2.12: Comparison between two visualizations for sunshine intensity, a bar chart vs. Spiral Graph, which allows for much better pattern detection when parameterized with the right cycle length, 24h in this case (Weber et al. [WAM01])

CHAPTER 3

Problem characterization

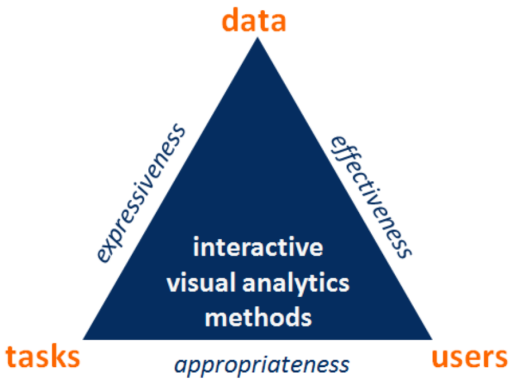


Figure 3.1: Data, Users and Tasks as major design factors for Visual Analytics, from Miksch et al. [MA14]

In this chapter we look at the major design factors for Visual Analytics and define then for our project, following the Data-Users-Tasks Design Triangle for Visual Analytics of time-oriented data proposed in Miksch et al. [MA14], depicted in Figure 3.1. It aims to answer the same questions as the first 3 steps of the Discovery phase, "Characterize the problems and data of the domain at hand", "Define the tasks" and "Determine who or what defines the 'relevance of information' for a given task", but gives more detailed guidelines for finding the major factors that determine suitable visualization and interaction methods. In order to avoid confusion, we will be referring to the "users" mentioned in the Data-Users-Tasks Design Triangle as "analysts" and to the platform's customers as "users".

Miksch et al. [MA14] also lists expressiveness, effectiveness and appropriateness as quality criteria that the Visual Analytics methods need to meet in order to be of use in accomplishing the goals. Following the corners of the design triangle we have identified:

3.1 Data

Qualitative and quantitative multivariate sport activity booking behavior data from a online booking platform. The following data entities were identified:

Users with demographic information like date of birth, gender, city

Bookings the date and time of sport activities

Facilities sports venues that offer bookings and/or events

Events sporting activities organized by the facilities, like classes, camps, workshops, etc.

Cities city and country information of users and facilities

Sports sport type name

Categories a polyarchy for classifying the sports

Other important information we have are the social relationships between users: if they have played a booking together (e.g., booked a tennis court and played against each other) or attended the same events (e.g., attended the same Yoga class)

Time is a separate dimension that needs to be considered to allow observation of trends and patterns. For this, we consider a cyclical time arrangement with an instant as the time primitive.

The problems encountered are incomplete/inconsistent data, which makes an automated approach infeasible and ineffective. Demographic data like gender, date of birth, address is only available for around half of the user entries.

3.2 Analysts

The analysts that are the primary target group of this project are non-technical people, mostly with a marketing or business background. As a rule, they will not be familiar with any programming or query language, but possess good domain knowledge on the plethora of available sports types and venues.

3.3 Tasks

Through an informal survey with the analysts two types of tasks were identified: suggestion retrieval and data exploration.

The main task is calculating and visualizing the affinity of users or groups of users for new sports, which can materialize in booking a court at a venue that offers it, or in attending an event such as a camp, class, workshop, etc., for this sport.

This potential interest of a user for a sport will result from a (weighted) combination of different factors. Users should be more affine to events

- involving sports similar to the ones they usually practice
- involving people that are similar to them

Similarity between sports is calculated as a function of the inherent (objective) characteristics of the sport. For this purpose, a hierarchical sport category graph has been set up and a similarity function defined based on it, as described in 4.1.1.

Similarity between users is based on a collaborative filtering approach, and is defined as users played a booking together, attended the same event(class, camp, workshop...), or have played the same sport.

The tasks identified for suggestion retrieval were:

- define the user target group (cohort) for which suggestions should be computed based on demographic data by drawing a graph pattern
- find users that belong to this cohort
- retrieve general suggestions for the cohort
- retrieve individual suggestions for each user from the cohort

For the data exploration part, the most common questions were:

- what other sports are closely related to a given one?
- when was a given sport most often played? what is the booking behavior for a given sport?

From the questions posed, to enable data analysis explore qualitative and quantitative data relating to sports, we derived the following tasks:

- for a given sport, show booking information like time and frequency

- for a given sport, show similar sports as defined by the similarity function
- give the analyst a way to correct the similarity function output, if the computed value seems unreasonable
- allow for node based navigation of sports, as an explorative data browsing mechanism

CHAPTER 4

Design

This chapter describes the design of visual analytics process. We started the design following the problem characterization in Chapter 3, drawing the requirements for the prototype from the data, users and tasks that we identified.

In section 4.1 we describe the structure and the domain of the data, as well as the ETL process. For the sports, we create a category tree and define our own similarity function based on the topology of this tree. Then, in section 4.2 we define the visual encoding and interaction design, based on the learnings from Chapter 2.

4.1 Data Abstraction

The domain data can be seen in Figure 4.1. We are handling two types of data in this project. First we have the qualitative, nominal data for the entities Person, City, Facility, Sport, Category, and then there is the quantitative, discrete data for Booking and Event, as these represent the discrete dates when they were held.

Data for city, facility, event, user, booking data, sports was imported from the companies relational database. Then, in order to bring it to a consistent state and format, which is a precondition to performing any kind of analysis, the data was transformed, cleaned and imported into a Neo4j graph database.

The next step in gathering the data was to build a sports classification tree, in order to augment the existing data in Neo4J and build the basis of the project. The different types of sports were pulled from Wikipedia through a script that starts from the "Sports by Type" page in Wikipedia [wik], follows each subcategory link recursively and generates a nested JSON object of the subcategories. The resulting JSON object can be seen in Appendix A and a visual representation in Figure 4.2. This data was then imported into our graph database.

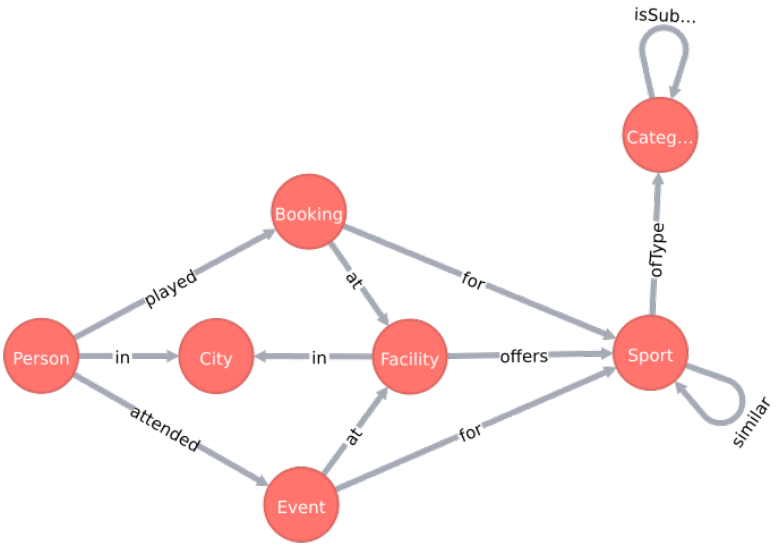


Figure 4.1: Metagraph, showing how the data entities are linked

4.1.1 Sports similarity function

The next step was finding a way to compute the intrinsic similarity of the sports, based on their characteristics, as revealed by the categorization. As sports can belong to several different (sub)categories, e.g., Tennis can be played both indoor and outdoor, so it belongs to both "Indoor Sport" and "Outdoor Sport", they were linked to all categories that they could fall under and also always connected to the most specialized one (the deepest level) Then we defined our own "sports similarity function" that made use of this topology.

By considering all sport-category links, we can have several paths between any two sports S_x and S_y, as exemplified by the pairs Tennis and Badminton in Figure 4.3 or Tennis and Gymnastics in Figure 4.4.

For each of these paths, we define C_x and C_y as the most specific categories that S_x and S_y, respectively belong to, and C_a as the branching category, the closest common ancestor of C_x and C_y.

The level of a category L_C is defined incrementally from the root, the root being level 1.

The weight function for each path is then computed as

$$weight(path) = 1 - 0.5^{L_{C_a}}$$

$$f : (S \times S) \rightarrow [0, 1]$$

where $f(S_x, S_y) = 0$ means not at all similar and $f(S_x, S_y) = 1$ means they are the same sport.

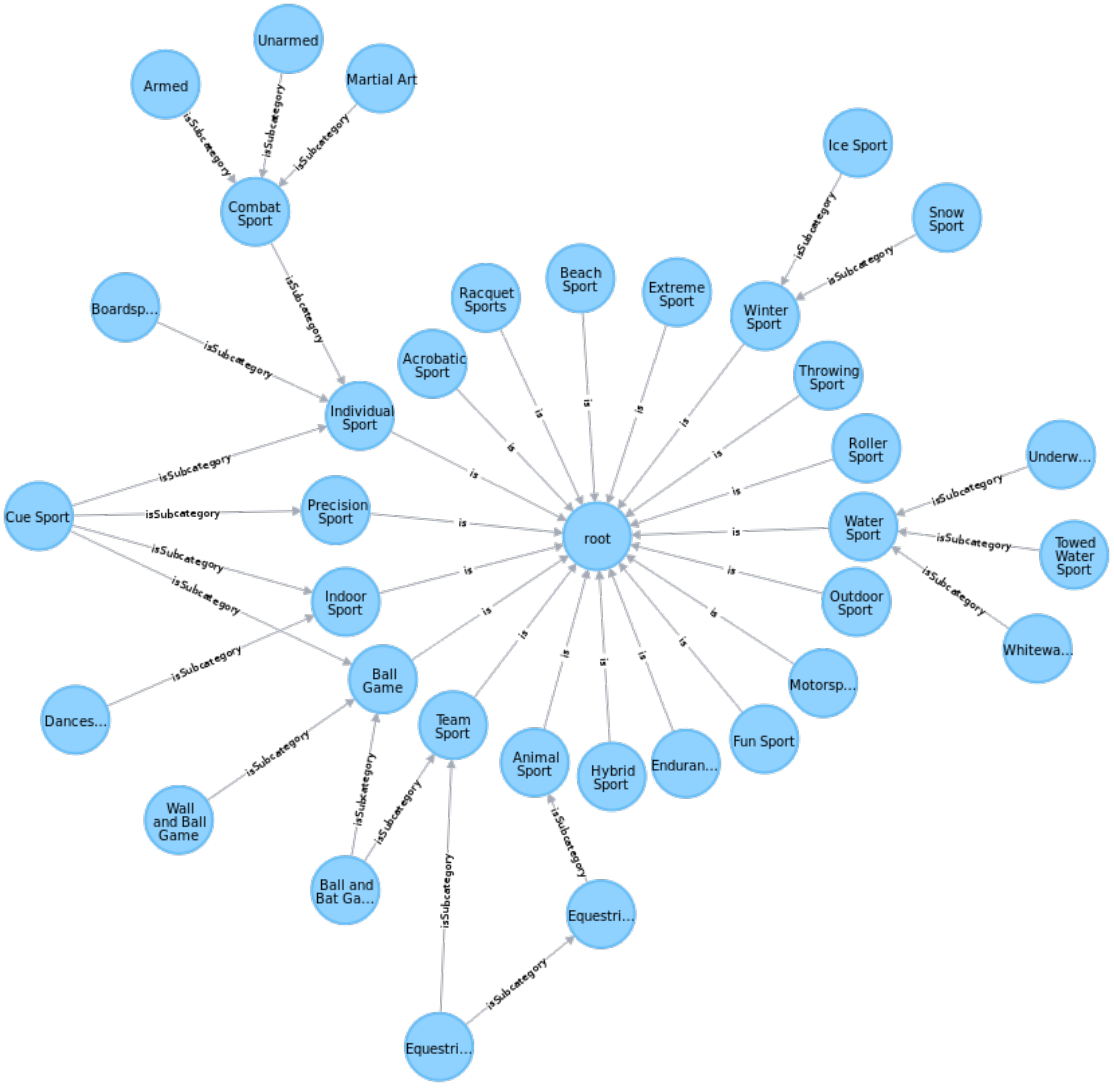


Figure 4.2: Sports Category Polyarchy - A visual representation of the JSON object in Appendix A - Sports categories. The edges show the *–isSubcategoryOf–>* relationships

$$f(S_x, S_y) = \frac{\sum_{i=0}^{n} weight(path_i)}{n}$$

where the weight of each path between the two sports is defined as a function of the category levels C_x of S_x, C_y of S_y, and C_a of the closest common ancestor of the sports on this path.

After defining this function, a script was run on all sports pairs and edges were added to Neo4J between the sports with the resulting values properties.

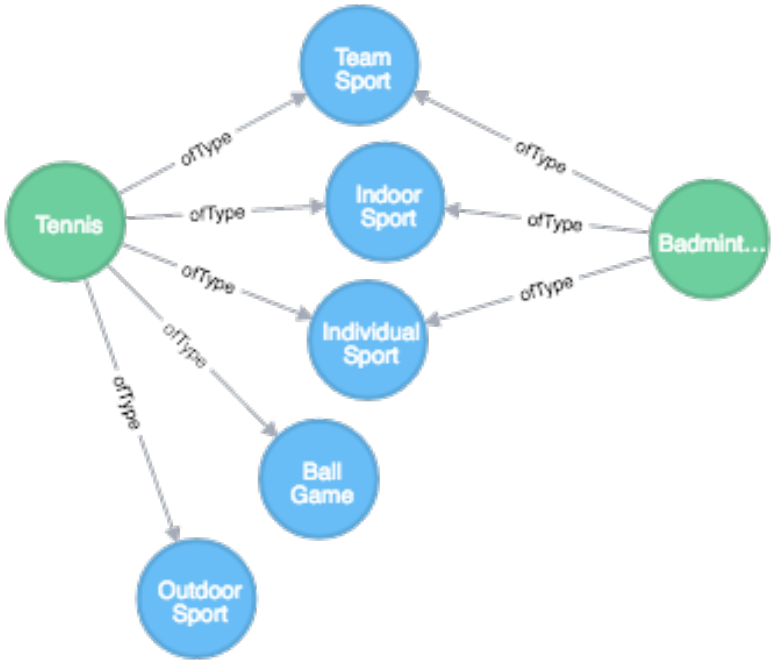


Figure 4.3: Categories and paths for Tennis and Badminton, $f(Tennis, Badminton) = 0.45$

4.1.2 Modeling time

For fast querying of time data, a tree structure as described in [MB] was chosen to represent the years, months and days, exemplified in Figure 4.5.

The time tree structure has a node for each year, and each year is connected to its separate set of Month nodes, labeled by the month number, by [HAS_MONTH] relationship edges. Each month is in turn connected to its own Day nodes by [HAS_DAY] edges.

The leaf nodes of the tree represent days, and they each have a pointer to the next day, materialized as [NEXT] edges.

Each of the Booking and Event nodes is then connected to the Day, Month and Year node that they belong to. As each Day node is only connected to one Month node, which in turn is connected to one Year node, the full date can be determined by traversing three edges.

The advantage of having [NEXT] pointers on the day leaf nodes is that one can quickly perform time queries. For range queries one must find the beginning of the time interval, follow the [NEXT] pointers until the end date and retrieve for each the nodes (Bookings or Events in this case) that are connected. This approach is much faster than doing index lookups on a text date fields stored in the Bookings and Events entities.

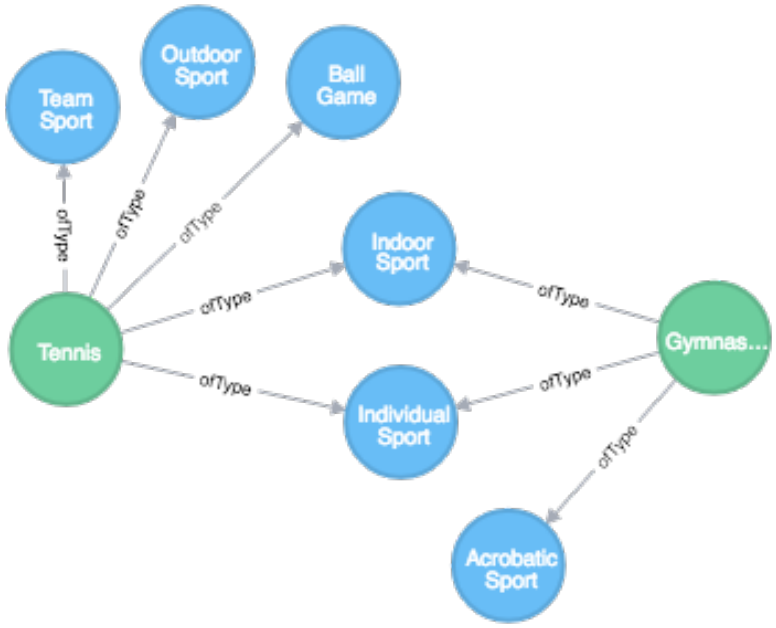


Figure 4.4: Categories and paths for Tennis and Gymnastics, $f(Tennis, Gymnastics) = 0.25$

4.2 Visual Encoding and Interaction Design

Following Keim's definition of Visual Analytics in Keim et al. [KAF+08]

> *Visual analytics combines automated analysis techniques with interactive visualizations for an effective understanding, reasoning and decision making on the basis of very large and complex data sets.*

we present the information in a task-oriented way and provide the analysts with interactions that facilitate problem solving. Interactive techniques are important for the visual exploration of data.

Based on the design requirements that we have determined through the task analysis we decided we should have two types of interactive visualizations for our graph data as they address different aspects of the problem and have different challenges.

- the **Sports Overview** page and the **Sports Detail** page, for exploratory purposes
- the **Visual Query** canvas, for information extraction

In this section, we briefly describe purpose and characteristics of the proposed views.

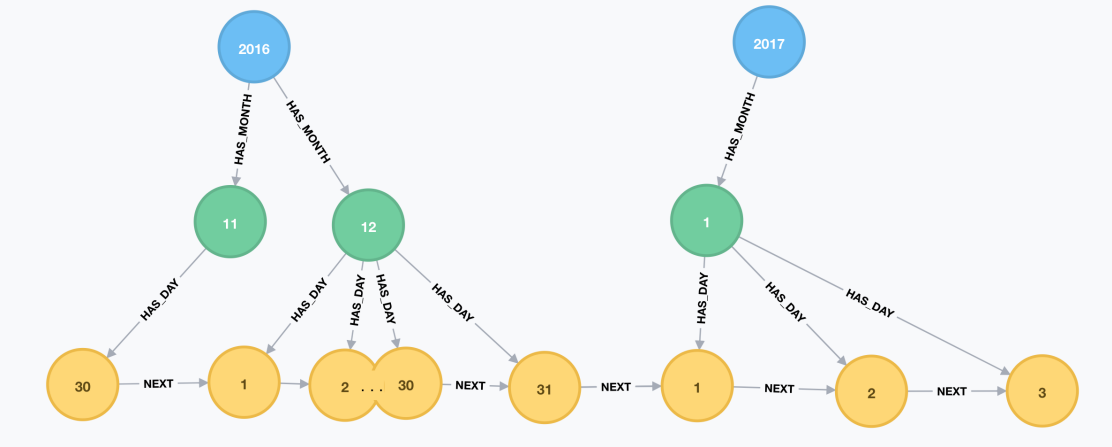


Figure 4.5: Time tree diagram, showing the connections for transitions between months(30^{th} of November - 1^{st} of December) and years (31^{st} of December 2016 - 1^{st} of January 2017)

4.2.1 Sports Overview

The Sports Overview page is a force-directed graph layout of all sports that uses the similarity values as separation constraints and serves to give the analyst a mental map of what's available.

Both the sports overview and the closely related sports detail page are driven primarily by the exploratory and analytic needs of the analysts. They are node-link representations of a view of the graph, where the sports nodes are represented as circles and classic straight lines are used to visualize connectivity, as seen in Figure 4.6.

The size of the nodes encodes booking information, as it corresponds to the total number of bookings and event participations for that sport. It it a proportional encoding using a logarithmic scale, as we have a very large domain (0 to 500000). The scale information panel is displayed underneath the fold to save space for the visualization.

The overview shows all sport nodes but, in order to reduce visual clutter, only those links are shown which represent a similarity function value >=0.5 (50%).

The overview allows for several observations:

- clusters of sports form, like the highly connected martial arts at the top left or the ball sports at the bottom right
- which sports are very popular, based on the node size (Tennis, Soccer, Squash, Beach Volleyball, etc)
- which sports don't have closely related sports (Paintball, Kart, Ice Stock Sport, Freestyle Jump)

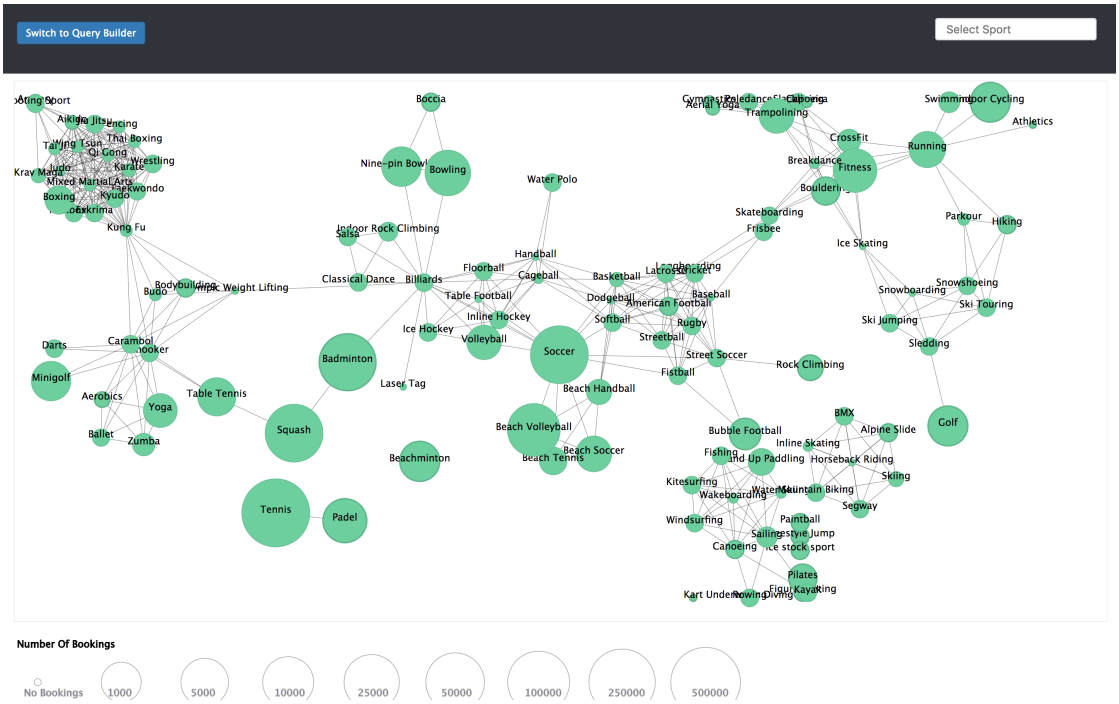


Figure 4.6: Sports overview graph with scale information at the bottom

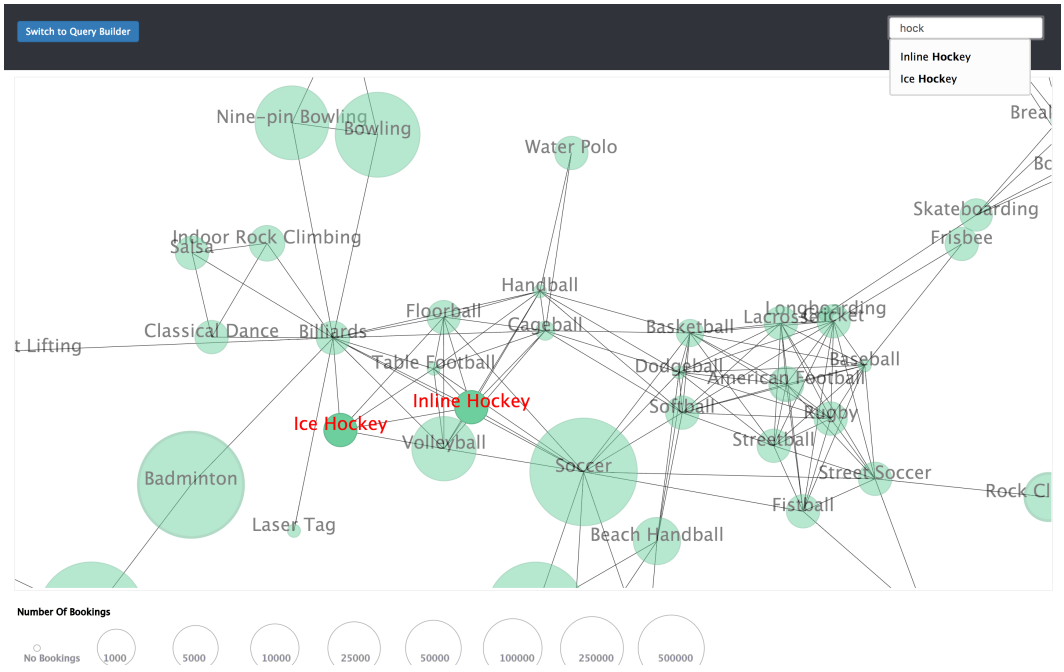
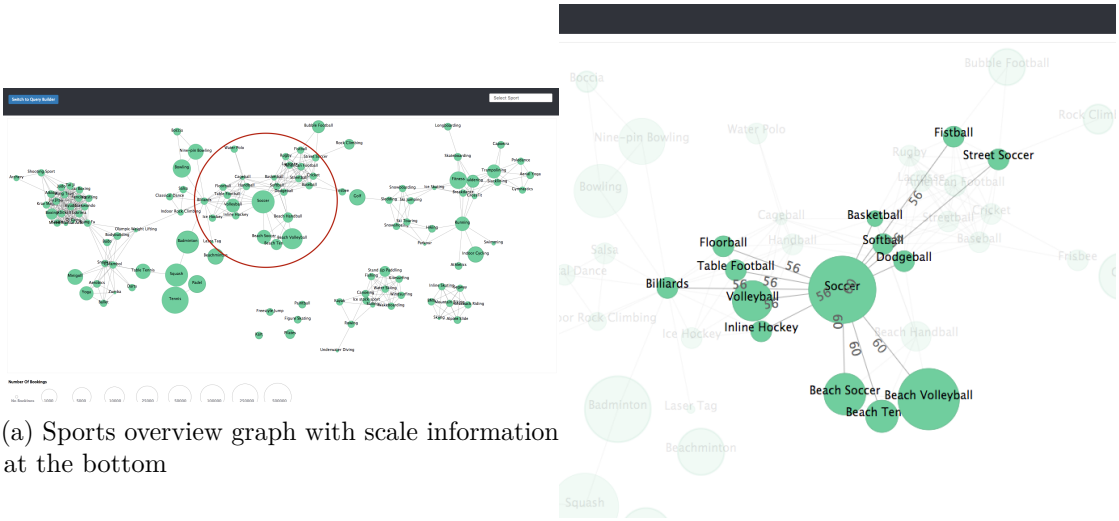


Figure 4.7: Sports overview filtering with autocomplete, for the string "hock", zoomed in view



(a) Sports overview graph with scale information at the bottom

(b) Edge labels display similarity information

Figure 4.8: Node click interaction on the sport Soccer highlights it and its connections

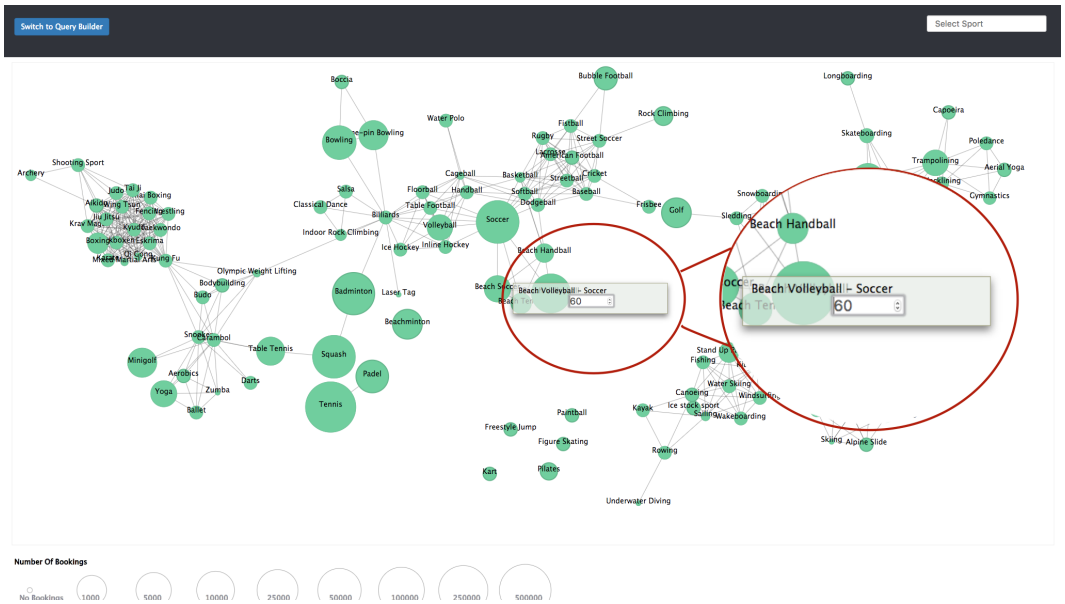


Figure 4.9: Editing the similarity value between Soccer and Beach Volleyball

This view enables following interactions:

Drag node to rearrange The layout isn't fixed, it's a dynamic layout that uses the similarity information on graph links as constraints, the analyst can drag nodes to

slightly rearrange the layout for better visibility in case the edges overlap.

Pan&Zoom Different areas of the view can be zoomed in and explored using the mouse wheel.

Filter In order to enable fast lookup of a sport in the overview, a search box has been added to the top right of the view. First, it highlights the sport names containing the input string in real time in red, as seen in Figure 4.7. The desired sport can then be explored in the overview, or, alternatively, in the detail view. The sports detail view can be accessed by choosing it from the suggestions, or by double clicking the sport in the graph and is discussed in greater depth in Section 4.2.2.

Autocomplete The sport filter has autocomplete, to eliminate typos and errors caused by different spellings.

Details-on-demand If an analyst wants more information on a sport, they can click on it in order to highlight it and its connections, which retain their full opacity, unconnected elements fading out. String labels on the edges provide textual information about the similarity between sports as a percentage (Figure 4.8).

Direct Manipulation of similarity value The analyst is the ultimate authority making use of the interaction techniques provided to correct system behavior. If they come across a value that they consider to be inaccurate, they can change it by double clicking the edge. This renders a popup with the names of the sports and the current value. The new value is saved on ENTER and the new value is visible in the graph. Pressing ESC discards the unsaved changes and closes the popup (Figure 4.9).

Node-based navigation Node-based navigation by double clicking on a node to open the desired sport in the detail view

4.2.2 Sport Detail

The Sport Detail visualization is an extracted graph macro-view, small enough to fit on the available screen. The detail view focuses on one sport and its connections. The selected sport is in the middle and all others that have a similarity >=10% are arranged radially around it. There are mechanisms to filter nodes and edges via range sliders.

Here, both color and size of the nodes are used to encode numerical attributes. The node colors, together with the link lengths encode the similarity to the central node, with more similar sports being darker and displayed closer to it. As in the overview, the size reflects the booking numbers.

If booking information is available, it is visualized in two time spiral diagrams, which can be used to show recurrent patterns in booking behavior. A winding of each spiral corresponds to a year. For the top visualization, the breakdown is monthly, and the bottom one, daily. The color saturation for each sector corresponds linearly to the number

of bookings for that time interval. By hovering over points in the diagrams, we get absolute booking numbers for the corresponding time interval.

In Figure 4.10, the detail view for Tennis, we can observe that the total monthly booking frequency has steadily increased since November 2015. In the daily breakdown, we see a recurring pattern of weekdays being busier than weekends.

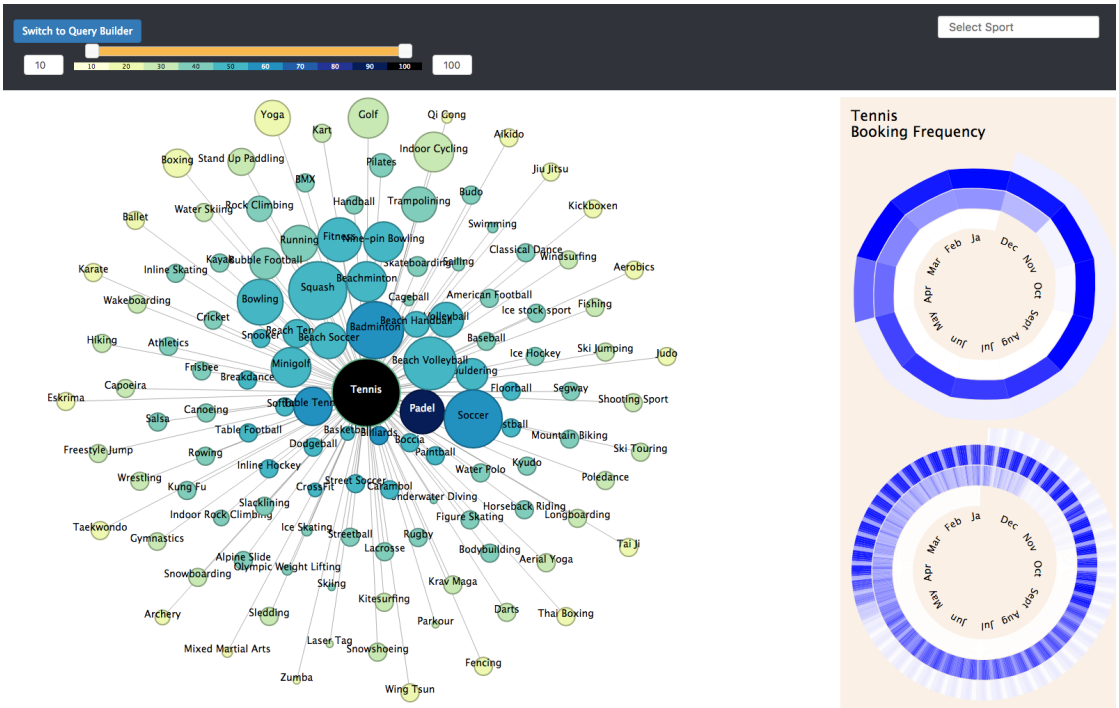


Figure 4.10: Detail view for Tennis, with monthly and daily booking information encoded in two time spirals on the right

Figure 4.11 shows that Running has a very different booking behavior, only September 2016, January and February 2017 have bookings on single days. This is a case that can benefit from the analyst's domain knowledge that this sport can be done freely, without needing a booking. However, people take part in events, like races, that do need booking, which explains the data points visualized.

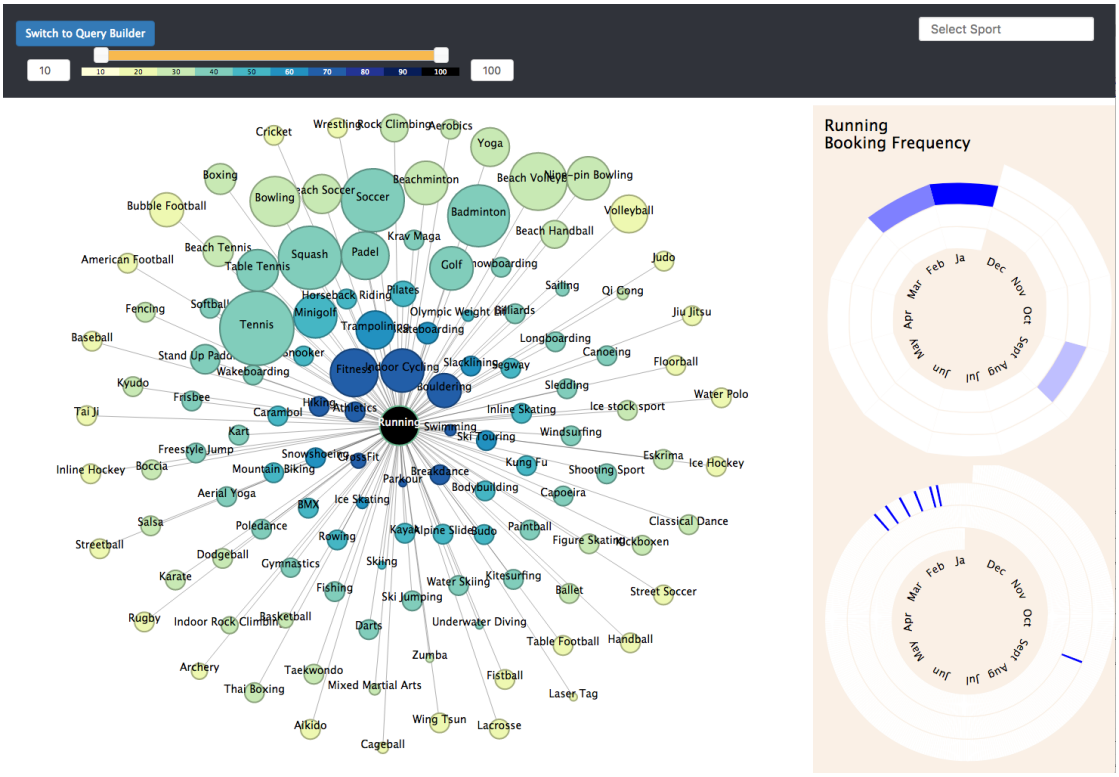


Figure 4.11: Detail view for Running, with monthly and daily booking information encoded in two time spirals on the right

According to our task definition, analysts are interested in the booking behavior of a given sport. In order to support them in fulfilling this task and identifying patterns and relationships between time and the booking numbers, we consider a cyclical time arrangement and instants as time primitives. The Spiral Graph described in Section 2.3 is an appropriate visualization for quantitative data, in this case, the number of bookings per time interval. Spiral Graphs support the comparison of values both in a neighborhood, so we can observe how the number of bookings changes from one month to the next, and in cycles, e.g., same month in different years. Although we only have data for three years, this approach is scalable, and can visualize large data sets.

In addition to supporting the interactions from the Sports Overview, the Sport Detail view also allows following interactions:

Filter Because there can be a lot of sports that have a similarity of at least 10% with the currently chosen node, the connected sports can be filtered by the similarity value to the central node by using the double slider, as in Figure 4.12. The value range can be selected by sliding or by inputting the values manually. The analyst sees the changes in real time.

Details-on-demand By hovering over the time spiral sectors, the time interval identifier (month-year or day-month-year) and the aggregated number of bookings for that interval is shown.

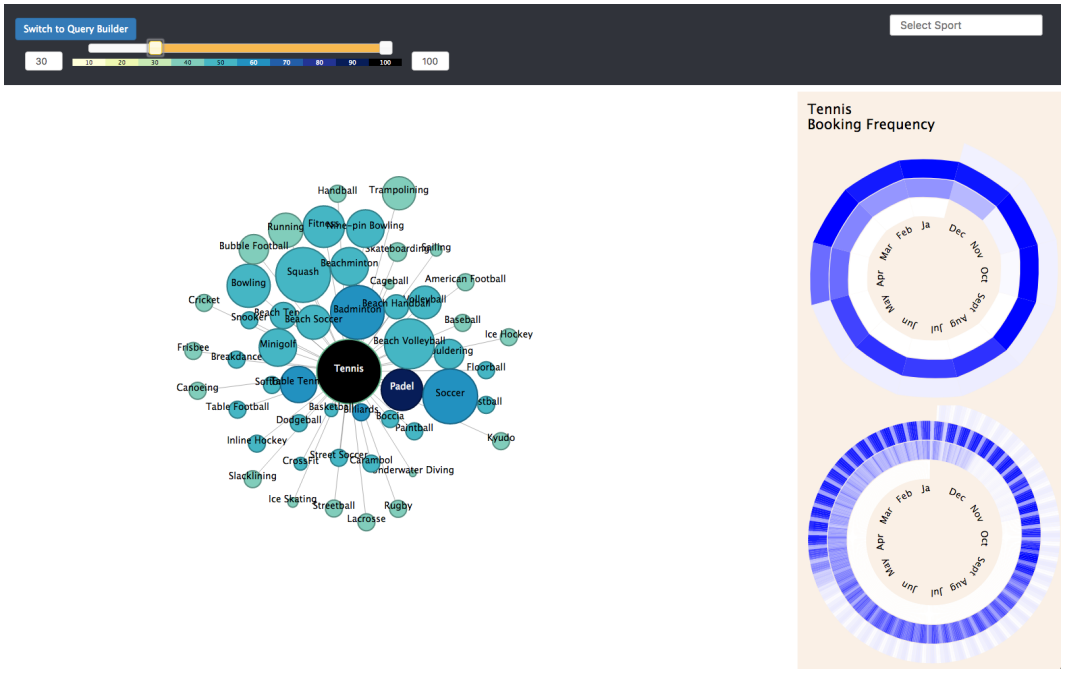


Figure 4.12: Filtered detail view for Tennis, showing sports with a similarity >= 30%

In both the Sports Overview and the Sports Detail visualization, exploration is made possible by the node-based navigation.

4.2.3 Visual Query

The Visual Query canvas consists of several views:

- the query canvas
- the cohort results visualization
- individual results table

The first step in suggesting new sporting activities to users is defining which subset of the users the analyst wants to target. As the common analyst will most likely not have a background in IT, they will have difficulties in directly describing the data they want as a correct text query. To facilitate the specification of the sought data set, we developed an abstraction layer for the graph database in the form of a visual query language.

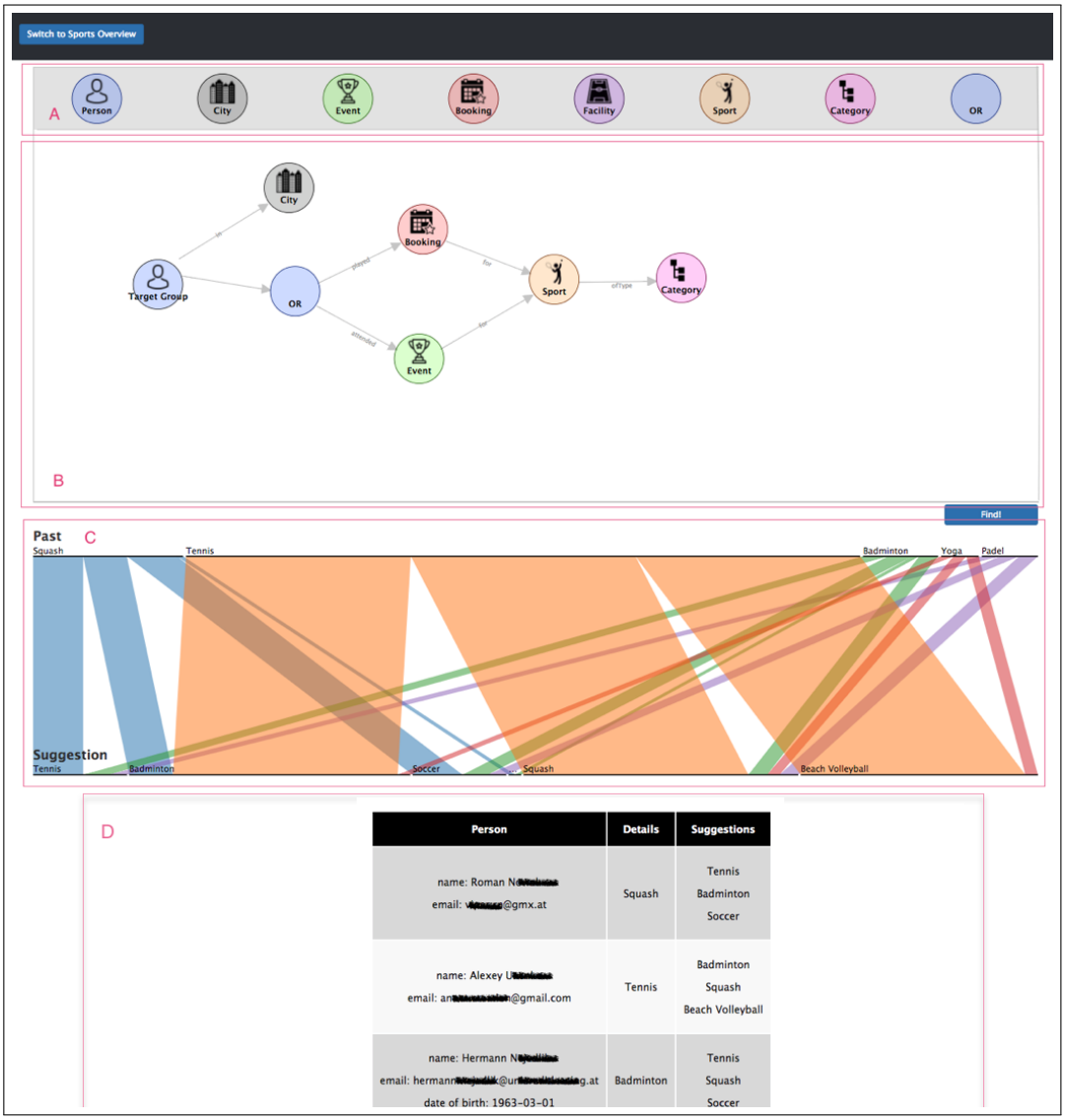


Figure 4.13: Overview of the Visual Query view. The query pattern is drawn in the query canvas (B) by Drag&Drop from the entity menu (A), with the results visualized as parallel sets in the results area (C) and the individual results as a table (D)

To formulate their queries, the analyst uses the query panel to define the user target group that they want to find suggestions for. These target groups, or cohorts, are defined as a graph pattern that is created by using Drag&Drop to pull their chosen data entities from the menu to the query panel and connect them by clicking and dragging the mouse with depressed left mouse button between nodes. The edge types and possible connections are predefined to only allow relationships that are valid.

Each data entity type is encoded as a colored circle with an icon and a text label to make recognition easy at a glance. The relationships between entities are also labeled and the edges are directed, to indicate the direction of reading the label, e.g., Person –[in]–> City, or Booking –[for]–> Sport.

A special entity that has been added is the OR-node. The Cypher query language does not support this kind of alternative paths, so it is our extension to the Cypher query language and is used here to build queries where at least one of the two outgoing paths yields results.

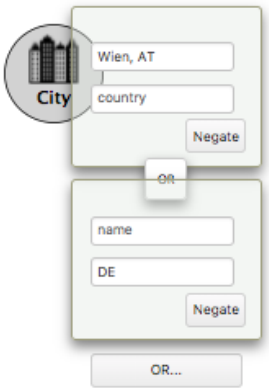


Figure 4.14: Filter for City is Vienna OR City is in Germany

The analyst can add constraints to the nodes through filters, like in Figure 4.14, which are accessed on right click on the node. This opens a popup context menu with initially one filter group that contains each property once. These filters are all optional and AND-connected. Each such group can be negated using the "Negate" button, and any number of further groups can be added to be OR connected by clicking the "Add Filter" button.

For ease of use, the input fields for the name filters for most entities(Sport, Category, Facility, City) are autocomplete search fields that allows the analyst to select the desired entity name using only a few keystrokes.

After drawing the pattern that corresponds to the target group definition, the analyst can run the query by pressing the "Find" button located at the bottom of the screen.

The cohort query results are then visualized using the Parallel Sets technique, described in Kosara et al. [KBH06]. There are two dimensions, which hold information as to what sports the users in the cohort have engaged in in the past and which should be suggested to them. The boxes in each dimension represent the different categories (sports in this case) and are scaled according to their corresponding frequency, so their size

is relative to all the data samples. The colored parallelograms show the relationship between categories.

As we can see in Figure 4.13, for the query *People that live in Vienna or Berlin and have bookings or events for a Racquet Sport* the most played sport in the past is unsurprisingly *Tennis*, followed by *Squash, Badminton, Padel* but that some of these users also attended *Yoga*. It's also easy to see that the suggestions for this cohort are *Badminton, Squash, Beach Volleyball, Soccer, Tennis* and *Table Tennis*.

This overview of the results can be augmented with details on demand by hovering the mouse over a category or a connection to get quantitative information, like absolute numbers and percentages, as shown in Figure 4.15. For example, by selecting *Squash*, we learn that has been played by 24 (15%) of cohort users in the past, and for these, *Badminton, Soccer, Tennis* and *Table Tennis* should be suggested. By selecting only the *Squash -> Badminton* relationship, we see that this particular suggestion occurs in 7 (4%) of cohort users.

The categories are by default sorted randomly, but can also be sorted either alphabetically or by size.

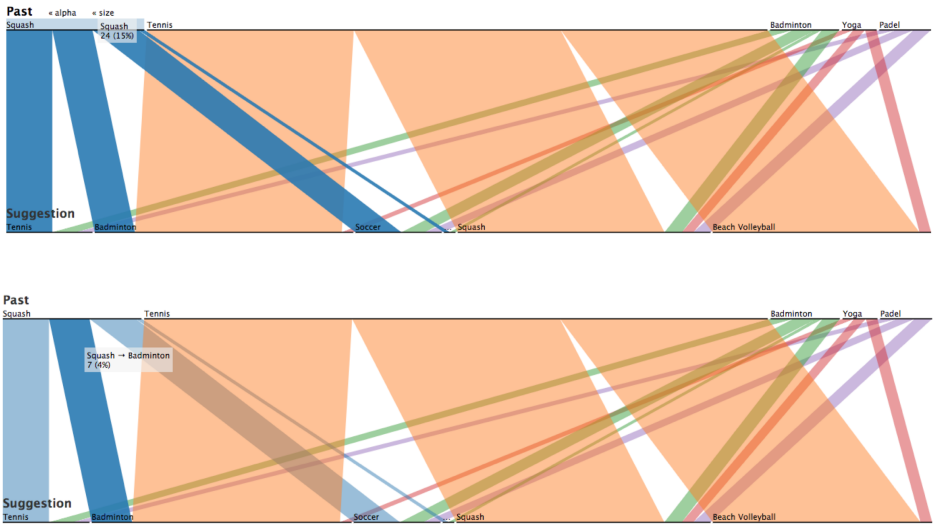


Figure 4.15: Results for the query *People that live in Vienna or Berlin and have bookings or events for a Racquet Sport* visualized as Parallel Sets. Highlighting of the *Squash* category in the dimension *Past* (top) and the relationship *Squash-Badminton* (bottom)

For better insight, the individual results are shown underneath the cohort result visualization as a table view. The analyst can scroll through the user list, where past activities and the suggestions are displayed and can be analyzed. In Figure 4.16 we see the first five users with their top three suggestions. As per our query, all have at least one Raquet

Person	Details	Suggestions
name: Roman N[redacted] email: v[redacted]@gmx.at	Squash	Tennis Badminton Soccer
name: Alexey U[redacted] email: an[redacted]@gmail.com	Tennis	Badminton Squash Beach Volleyball
name: Hermann N[redacted] email: hermann[redacted]@ur[redacted]g.at date of birth: 1963-03-01	Badminton	Tennis Squash Soccer
name: Klaus St[redacted] email: klaus.st[redacted]@gmx.net date of birth: 1985-07-15	Yoga Tennis	Badminton Squash Beach Volleyball
name: Markus D[redacted] email: markus.d[redacted]@hotmail.com	Badminton Squash	Tennis Soccer Table Tennis

Figure 4.16: The individual results of the query as a table, with past activities and top suggestions

Sport among their past activities (Details column), but some have other interests as well (*Yoga* for Klaus).

Although each user only receives suggestions for sports they haven't tried out yet, the cohort suggestions will contain the union of these individual sets.

After the iterative creation of this design, the prototype was built and evaluated, which we detail in the next chapters.

CHAPTER 5

Prototype Architecture & Implementation

In this chapter we describe the prototype and its inner workings. First, in Section 5.1 we give a short overview of the technologies used and a high-level architecture description. Then, in Section 5.2, we describe all the steps that were necessary to get from drawing a pattern in the browser to finding matching results in the graph database.

5.1 Implementation

Our prototype has a client/server architecture. The client side is responsible for requesting the data from the server using JSON-RPC over HTTP and WebSocket calls based on user input, visualizing it and handling user interactions. It it a JavaScript web client that relies on D3.js as the visualization library, with JQuery support for DOM manipulation.

The server side handles data processing, by transforming query data to the query language of Neo4J, Cypher, querying the database and extracting the results to a usable format. The web server is built on top of the NodeJS and Express frameworks and communicates with a Neo4J database server using Neo4J's new binary network protocol Bolt, which works over a TCP connection or WebSocket. Is a highly efficient, lightweight client-server protocol designed for database applications.

This decoupled setup allows for system distribution and scaling of the backend if very large data sets need to be processed, but for this proof of concept a single machine with 2,4 GHz Intel Core i5 CPU and 8GB RAM was sufficient for a graph of >855.000 and >6.000.000 edges.

5.2 Visual Query Language

The first task in the process of retrieving suggestions for users is to define which users are of interest. In this sense, the analyst can describe the target group by drawing a pattern by Drag&Drop in the query panel. This pattern then undergoes parsing and transformations in a series of steps, so we can translate it to a Cypher query. Cypher is the declarative graph query language that comes bundled with Neo4J. It is expressive and powerful, while still being human-readable.

Client-side visual pattern encoding

After the analyst has drawn the visual pattern and clicked the "Find!" button, the pattern is encoded on the client side as a JSON object, with an array of nodes with filters and an array of links, as seen in Appendix B. A query request containing this structure is then sent to a server endpoint.

Server-side paths data structure

On the server, the query request is transformed to a new JSON data structure that describes the paths and the subpaths that need to be matched, filtered and disjunctively connected in the graph. The terminology used in this structure (match, path, where) was chosen intentionally to map to the Cypher keywords used in the query language. A formal description of the server side data structure can be found in Appendix C.

The structure is an array of paths, where each path is an object containing a array of matches and an optional so called disjunct-field. This is simply the ID of the OR-node, if applicable. This is needed to be able to correctly assign the paths that need to be OR-connected. Each match contains an array of nodes, and an array of conditions imposed on them, as well as a "negated"-flag, which, if set, determines that the results matching that subpath are to be filtered out of the results.

Translation to Cypher

The last transformation step was translating the paths data structure described above into a Cypher query, which is run against the Neo4J database. This returns an array that is limited to a maximum 1000 IDs of Persons that fall into the defined target group.

Suggestion Retrieval

The previous step returned the cohort as an array of Person IDs. For these IDs, suggestions are then computed by using a collaborative filtering approach, which we explain using the example in Figure 5.1.

For all users in the cohort, *Florentin and Tanja*, we find all sports for which they bookings or event participations, in this case *Badminton, Squash, Tennis and Table Tennis*. Then we find all other users with similar interests, that have also partaken in these sports. For

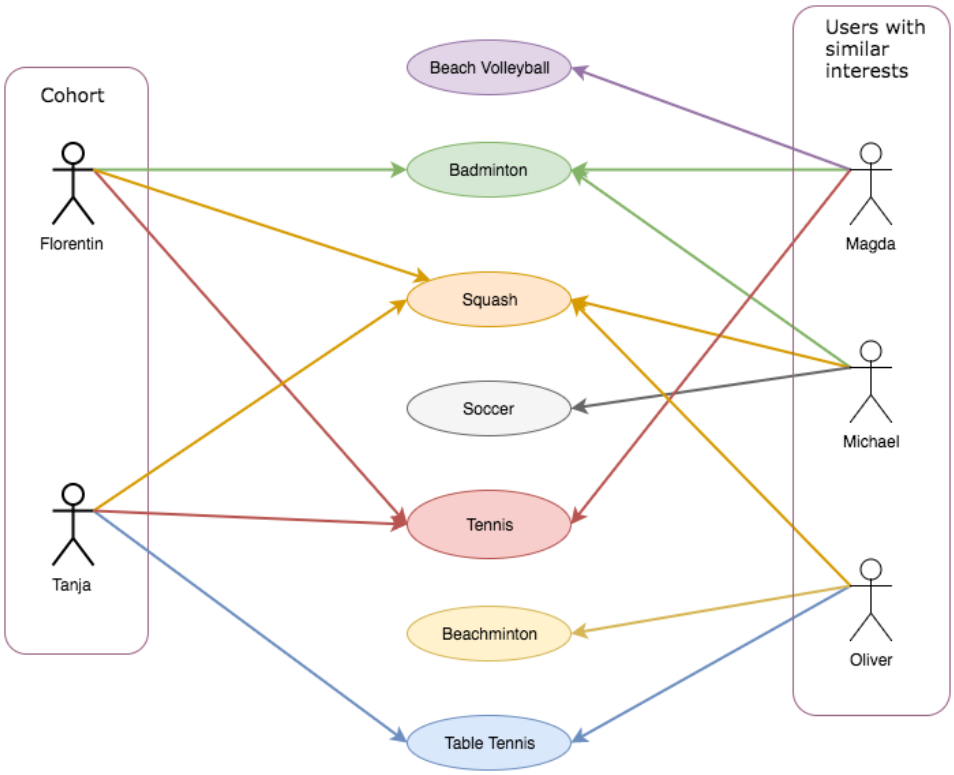


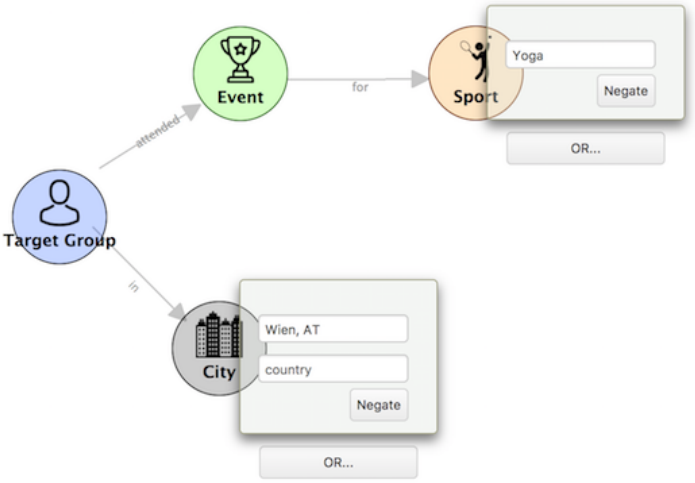
Figure 5.1: Collaborative filtering used for suggestion retrieval

these new users, *Magda, Michael, and Oliver*, we get all new activities and the associated sports. This is how we find our suggestions. We have 3 sports that are new for and should be suggested to all cohort users, *Beach Volleyball, Soccer, and Beachminton*, but also some that are only relevant for some *Table Tennis for Florentin, Beach Volleyball for Tanja*.

5.3 Use case scenarios

We present some use case scenarios that align with our requirements and show how our prototype can support analysts in their tasks.

People that live in Vienna and have attended events for yoga



Visual Query Pattern

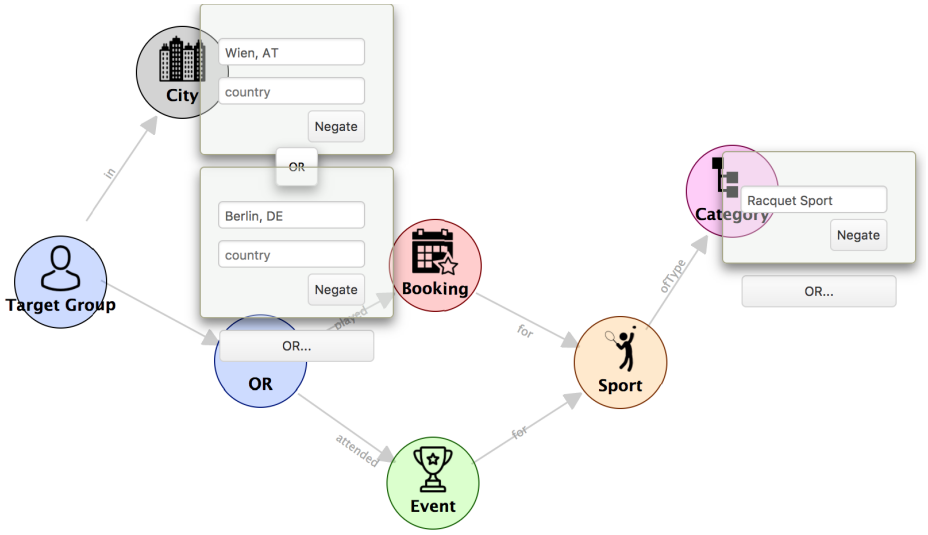
```
1               MATCH (P8:Person)--(C1:City)
2                 WITH P8,C1
3               MATCH (P8:Person)--(E2:Event)--(S5:Sport
    )
4                 WHERE (S5.name = 'Yoga')
5               RETURN distinct P8.id as PersonId
6               LIMIT 1000
7
```

Generated Cypher Query

Figure 5.2: Comparing the complexity of querying for *"People that live in Vienna and have attended events for yoga"*

The corresponding client-side and server-side data structures for this pattern can be seen in Appendix D.1.

People that live in Vienna or Berlin and have bookings or events for a Racquet Sport



Visual Query Pattern

```
        MATCH (P8:Person)
          WITH P8
        OPTIONAL MATCH (P8:Person)--(B3:Booking)--(S5x7_1:Sport)--(
C6x7_1:Category)
          WHERE (C6x7_1.name = 'Racquet Sport')
          WITH P8,B3,S5x7_1,C6x7_1
        OPTIONAL MATCH (P8:Person)--(E2:Event)--(S5x7_2:Sport)--(
C6x7_2:Category)
          WHERE (C6x7_2.name = 'Racquet Sport')
          WITH P8,B3,S5x7_1,C6x7_1,E2,S5x7_2,C6x7_2,
            (S5x7_1 IS NOT NULL AND S5x7_2 S NOT NULL) AS S5x7_both
,
            (S5x7_1.id = S5x7_2.id) AS S5x7_same,
            (C6x7_1 IS NOT NULL AND C6x7_2 IS NOT NULL) AS
C6x7_both,
            (C6x7_1.id = C6x7_2.id) AS C6x7_same
        MATCH (P8:Person)--(C1:City)
          WHERE (C1.name = 'Wien, AT') OR (C1.name = 'Berlin, DE')
          WHERE (NOT S5x7_both OR S5x7_same)
            AND (NOT C6x7_both OR C6x7_same)
            AND (B3 IS NOT NULL OR E2 IS NOT NULL )
        RETURN distinct P8.id AS PersonId LIMIT 1000

```

Generated Cypher Query

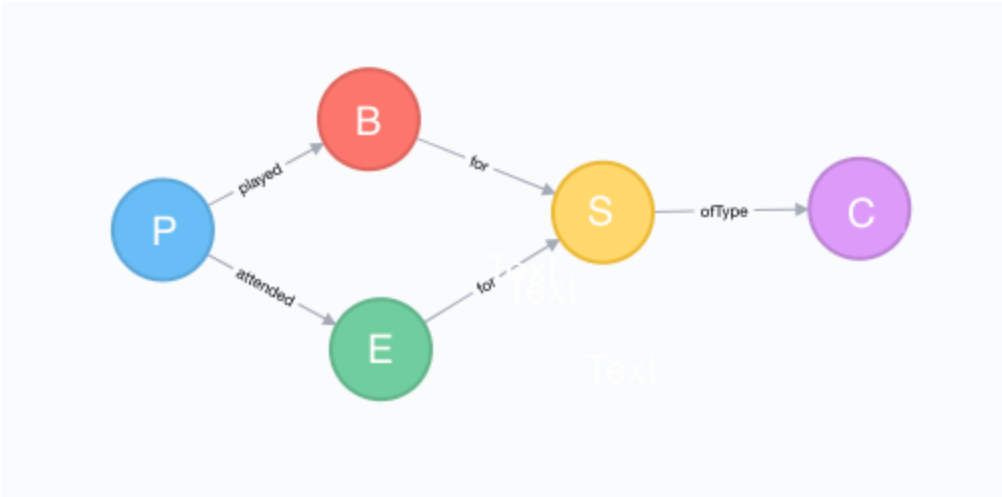
Figure 5.3: Comparing the complexity of querying for *"People that live in Vienna or Berlin and have bookings or events for a Racquet Sport"*

The corresponding client-side and server-side data structures for this pattern can be seen in Appendix D.2.

5.4 Limitations

In order to be able to describe queries containing disjunctive paths (e.g., get users that have played a booking or attended an event for a Racquet Sport that isn't Tennis) we added an OR node. This does not belong to the standard set of Cypher features.

Cypher only supports optional paths, which are of the form



Users that have played a booking or attended an event for a Racquet Sport that isn't Tennis

```
1 MATCH (P:Person) WITH P
2  OPTIONAL MATCH (P)-[:played]->(B:Booking)-[:for]-(S:Sport)-[:ofType
     ]->(C:Category)
3   WHERE NOT(S.name = 'Tennis') AND (C.name = 'Racquet Sport')
4   WITH P
5 OPTIONAL MATCH (P)-[:attended]->(E:Event)-[:for]-(S:Sport)-[:ofType
     ]->(C:Category)
6     WHERE NOT(S.name = 'Tennis') AND (C.name = 'Racquet Sport')
7 RETURN P
8
```

Cypher query

Figure 5.4: Limitation of OR-connected paths

As the "OPTIONAL MATCH" clauses don't need to be fulfilled for a valid result, the query this will return

- persons that have played at least one booking but have no events

- persons that have attended at least one event but have no bookings
- persons that have at least one of both
- persons that have none

To narrow down the search result to that of a boolean *OR*, constraints must be added to make sure that the *s:Sport*-nodes on both paths reference the same node in the database. The same goes for any further common nodes on the paths, like Category, in this case. Cypher doesn't allow node alias reuse on different paths, so we must make sure that if both paths exist, and so, two *s:Sport*-nodes are present, that they are the same node. This means that we give each node on the common path a different alias (sport S becomes S_1 and S_2, Category becomes C_1 and C_2), and then constrain them to point to the same node in the database if both exist.

Using the Sports nodes as an example, this is achieved by introducing two new variables

$$S_{both} := (S_1 \text{ is not null AND } S_2 \text{ is not null})$$
$$S_{same} := (S_1.\text{id} = S_2.\text{id})$$

and the condition

$$(\text{NOT } S_{both} \text{ OR } S_{same})$$

which means that S_1 and S_2 can't both be defined or, if they are, that they must point to the same node.

Following this reflection, the query becomes

```
1  MATCH (P:Person) WITH P
2    OPTIONAL MATCH (P:Person)-[:played]->(B:Booking)-[:for]-(S_1:Sport)--(C_1:
       Category)
3      WHERE NOT(S_1.name = 'Tennis') AND (C_1.name = 'Racquet Sport') WITH P8,
       B3,S_1,C_1
4    OPTIONAL MATCH (P:Person)-[:attended]->(E:Event)-[:for]->(S_2:Sport)-[:in
       ]->(C_2:Category)
5      WHERE NOT(S_2.name = 'Tennis') AND (C_2.name = 'Racquet Sport')
6        WITH P8,B3,S_1,C_1,E2,S_2,C_2,
7        (S_1 is not null AND S_2 is not null) AS S_both,
8        (S_1.id = S_2.id) AS S_same,
9        (C_1 is not null AND C_2 is not null) AS C_both,
10       (C_1.id = C_2.id) AS C_same
11 WHERE (NOT S_both OR S_same) AND (NOT C_both OR C_same) AND (B3 IS NOT NULL
       OR E2 IS NOT NULL )
12 RETURN P
```

As we can see, this technique works reasonably well for one OR-node, but introducing more than one OR node per path would lead to an exponential increase in variable number and query complexity, which is infeasible.

CHAPTER 6

Evaluation

With the growing interest and amount of research in the field of Information Visualization, it is becoming increasingly important to validate the research results, to see if a visualization does or does not support users in their information tasks. This is no easy feat, and the challenges in evaluating Information Visualization have been discussed extensively, i.e. by Carpendale [Car08], Plaisant [Pla04], Fekete et al. [FWSN08]. It is hard to evaluate Information Visualization techniques, because it's difficult to quantify how effective and efficient they are.

Carpendale in [Car08] explains how Information Visualization has some of the same challenges as other empirical research fields, like HCI, perceptual psychology or cognitive reasoning. First, the usability of a system is conditioned by the appropriateness of the visual representations and by the interactions with the interface, as this provides access to the data. Second, the results might be skewed by a number of factors. The user's motivation, domain knowledge or familiarity with other, older solutions are among the things which might influence the perception of new visualizations. Third, the level of insight gained is difficult to quantify and varies from person to person, especially when handling complex tasks that are not well defined. Also, insight triggered by a visualization might occur to a user with a substantial time delay and might not be traced back to the system. Lastly, decomposing a system in an attempt to evaluate it's components might not be useful, as the total might be greater than the sum of it's parts. It's difficult to tell if some results come from a specific visualization, or if it's an effect of the overall system solution.

Plaisant [Pla04] summarizes current evaluation practices, reviews challenges specific to information visualization and proposes refined evaluation methodologies. Based on an extensive literature survey, they identified four types of evaluation for information visualization

1. Controlled experiments comparing design elements

2. Usability evaluation of a tool
3. Controlled experiments comparing two or more tools
4. Case studies of tools in realistic settings

Fekete et al. [FWSN08] discusses the value of Information Visualization and the challenges behind identifying and communicating it. First of all, the role of Information Visualization is to amplify cognition, but measuring the level of amplification is not straightforward, so the value of InfoVis needs to be communicated differently. Second, the problems in this field don't have a ground truth, unlike algorithms that solve or don't solve a problem. This is why InfoVis is best suited for exploratory tasks, as is ours, where the user examines the data without a specific goal in mind, just to gain potential new insight into it. The user is being guided through the data to the parts that are potentially interesting by the InfoVis process itself and is also influenced by it, so measuring value of a InfoVis prototype as an exploratory aid is not easily quantifiable. Third, as a system incorporates several visualization and interaction techniques, it becomes more difficult to evaluate.

Some of the ways in which InfoVis can amplify cognition are

- Increasing memory and processing resources available
- Reducing search for information
- Enhancing the recognition of patterns
- Enabling perceptual inference operations
- Using perceptual attention mechanisms for monitoring
- Encoding info in a manipulable medium

Fekete et al. [FWSN08] also name some scenarios when browsing is useful and the value would be most evident

- When there is a good underlying structure so that items close to one another can be inferred to be similar - for example, similar sports being shown closer together in the sports overview and detail page
- When users are unfamiliar with a collection's contents - the users are familiar with the object types (Sports, Categories, etc) but might not be familiar with all the relations between the different sports, or what the cross-booking behavior is
- When users have limited understanding of how a system is organized and prefer a less cognitively loaded method of exploration

- When users have difficulty verbalizing the underlying information need - drawing patterns for quering, adding/removing filters iteratively is much easier than writing a text query
- When information is easier to recognize than describe - for example, similarity between sports and the booking patterns

We chose to focus early on the users, get feedback and design iteratively and the evaluation method was to conduct expert interviews in real world application scenarios.

6.1 Expert evaluation

In oder to validate our research question, we have tested our prototype on a domain expert from the business development department for feedback on the user interface and interaction techniques.

6.1.1 Evaluation method

- Present user with prototype
- Short explanation
- Give user tasks
- Observe while user tries to complete tasks
- Give the user time to explore the data on their own
- Note findings

The user is familiar with the domain, but received a short explanation on the structure of the prototype, on the different views with the corresponding interaction techniques and on the definition of the similarity function used.

To verify that the prototype actually fulfilled its intended purpose, we asked the analyst to try and complete instances of the tasks identified in section 3.3, split into suggestion retrieval and data exploration.

Suggestion Retrieval Tasks

Task 1 Define the cohort "People that live in Vienna or Berlin and have attended events for Yoga"

Task 2 Identify the users that belong to this cohort

Task 3 Retrieve cohort suggestions

Task 4 Retrieve individual suggestions

For the querying and suggestion retrieval, the analyst reported that the Drag&Drop for the query builder was intuitive to use. Connecting nodes and adding filters was done without error on the first try, fulfilling Task 1.

He suggested that, with each node addition to the query pattern, the next potential connecting node types could be made more prominent in the menu. This would make the underlying data structure more transparent and enable users that are less familiar with the domain to use the tool as well. For himself though, the constraint that an edge can't be drawn between unrelated node types was enough.

Running the query drawn in Task 1 displayed the results, leading to Tasks 2-4.

Identifying the users that belong to this cohort and the individual suggestions consists of reading the results table view.

The parallel set view of the cohort suggestion was greatly appreciated, as these flows and intersections are of great interest and were previously difficult to grasp using spreadsheets. Task 3 is fulfilled by reducing the cognitive load. The analyst remarked that the interactions on the parallel set visualization help a lot in digesting the information and understanding the users' preferences and partitioning.

Data Exploration Tasks

Task 5 Identify booking time and frequency for the sport *Padel*

Task 6 Identify most similar sports to *Padel*

Task 7 Correct similarity function between *Padel* and *Tennis*

Task 8 Navigate from *Padel* to *Tennis*

Starting in the sports overview, the user didn't need the search bar on the top right, letting his knowledge of the sport types guide him to the one he sought. When asked about it, he answered he knew it had to be very close to Tennis, which happens to have a large number of booking and is, subsequently, displayed as a very large bubble. He mentioned the search bar might prove useful for less connected and/or less often booked sports.

Highlighting the connected sports and fading the others in this view was very useful to reduce the visual clutter that the multitude of connections can cause, and see at a glance what sports are most similar. In this case, only Tennis has a similarity score of over 60% and is therefore connected, completing Task 6.

The user then selected Padel with a double click, opening the sport detail view. Observing the spiral graph, the user said it confirmed his intuition on the booking patterns of the

sport, with the vast majority happening between April and September, and some outliers in February and March, completing Task 5.

For Task 7, the user managed to update the similarity value after filtering the similar sports using the slider to reduce the visual clutter and more easily be able to click on the edge connecting Padel and Tennis.

The user completed Task 8 by double clicking the Tennis-node in the sports detail view, navigating to the Tennis detail view.

Free exploration

Given time to freely browse the data, the analyst mentioned new insight they gained on the similarity of certain sports, that hadn't occurred to him, for example Bowling and Boccia (54%), which are both target sports that can be played individually or in a team.

On the other hand, the analyst felt that some of the computed values are off, like the 75% similarity between Budo and Carambol, which he was easily able to change.

The color encoding for the similarity in the detail view was considered meaningful, especially as the colors are mirrored in the slider, which made it intuitive to use.

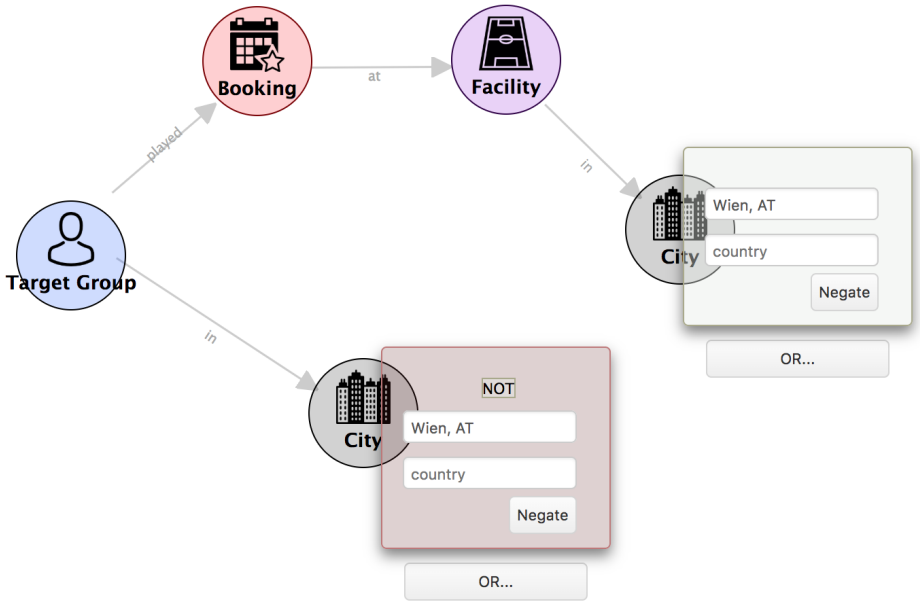
The user observed an interesting trend in the booking pattern for Bouldering, where every second month has around 3 times as many bookings as the month before, which he then explained as one of the largest venues offers courses in these months.

The user also tried out some more use case scenarios for the visual querying part.

People that don't live in Vienna, but have booking for facilities in Vienna The analyst wanted to see if the prototype could answer questions like "Are people that don't live in Vienna booking facilities in the city?", "Which sports are booked most often in this scenario?". The visual pattern drawn and the Cypher query can be seen in Figure 6.1. Results show that this is very strongly represented by Bowling, as can be seen in Figure 6.2.

What should be suggested to People that have attended events for "Bouldering" or "Indoor Rock Climbing"? Another use case the analyst wanted to try was to see what should be suggested to users that have attended events for Bouldering or Indoor Rock Climbing. The visual pattern drawn and the Cypher query can be seen in Figure 6.3.

According to the results in in Figure 6.4, the best matches, excluding Bouldering and Rock Climbing, would be Yoga and Soccer.



Visual Query Pattern

```
MATCH (P8:Person)--(C11:City)
  WITH P8,C11
MATCH (P8:Person)--(B3:Booking)--(F4:Facility)--(C1:City)
  WHERE (C1.name = 'Wien, AT')
RETURN distinct P8.id as PersonId LIMIT 1000
```

Generated Cypher Query

Figure 6.1: Comparing the complexity of querying for "People that don't live in Vienna, but have bookings for facilities in Vienna"

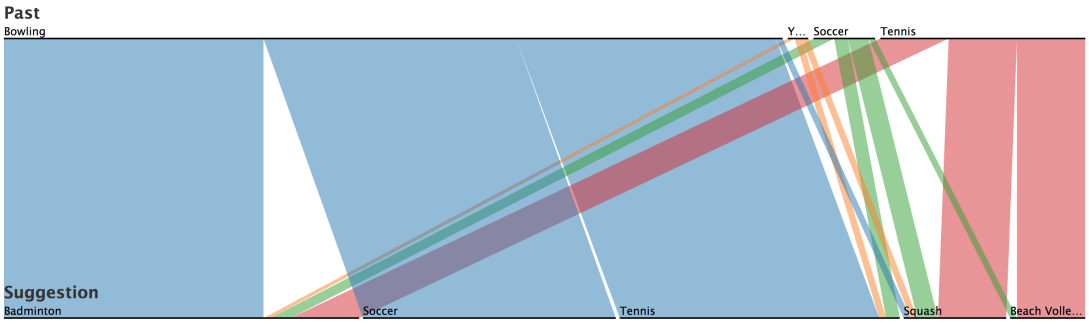
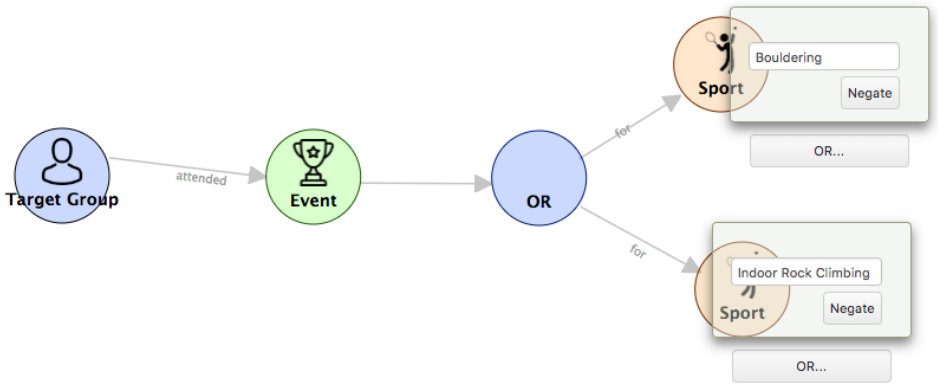


Figure 6.2: Parallel Sets result: The sport played most often in Vienna by people that don't live in Vienna is Bowling



Visual Query Pattern

```
MATCH (P8:Person)--(E2:Event)
  WITH P8,E2
OPTIONAL MATCH (E2:Event)--(S5:Sport)
  WHERE (S5.name = 'Bouldering') WITH P8,E2,S5
OPTIONAL MATCH (E2:Event)--(S11:Sport)
  WHERE (S11.name = 'Indoor Rock Climbing')
  WITH P8,E2,S5,S11
WHERE (S5 IS NOT NULL  OR S11 IS NOT NULL )
RETURN distinct P8.id as PersonId LIMIT 1000

```

Generated Cypher Query

Figure 6.3: Comparing the complexity of querying for "People that have attended events for "Bouldering" or "Indoor Rock Climbing"

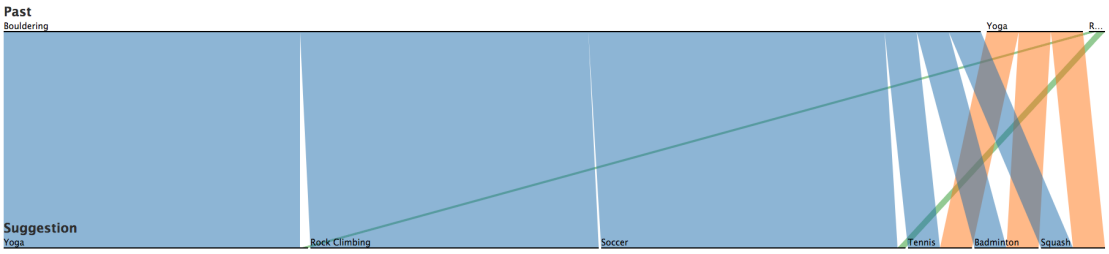


Figure 6.4: Parallel Sets result: Suggestions for People that have attended events for "Bouldering" or "Indoor Rock Climbing"

6.1.2 Summary

All in all, the domain expert found it intuitive and useful as both an exploratory aid and as a querying tool.

Some improvements were suggested, such as storing frequently used queries for later reuse, to allow having a repository of queries that the domain expert can simply select, maybe add a filter parameter and run; or highlighting the next viable connections in the node menu for drawing the patterns.

These improvements might be included in a future iteration, if the target audience of the prototype is expanded to include non-domain experts. These are great nice-to-have improvements and something we want to follow up on in future work.

CHAPTER 7

Conclusion

During this thesis, we tried to create a tool that enables analysts to synthesize information and derive insights from their company's large data set. The data was modeled as a graph and stored in a graph database, Neo4J, which is able to handle huge data sets.

The problem characterization revealed the data, users and tasks, and led to extensive multidisciplinary literature research from Information Visualization, human-computer interaction, Visual Analytics, graph databases and graph visualization. From the literature research we identified the need for a new tool for the domain of retrieving suggestions for sports activities, that allows for both visual querying as well as exploration and editing of the underlying sports similarity graph, as this combination was not found during our research.

From our main research question **How can Visual Analytics support the discovery of alternative and complementary sporting activities?**, we derived the hypotheses that **H1** Interactive visual querying can be an effective tool to find alternative and complementary sporting activities, that **H2** more insight can be provided by taking into account both the relational, as well as the temporal aspects of the data, and that **H3** interactive methods of the visualization can also help the analyst assess the value of these suggestions.

A prototype application that includes a visual query language was built based on the learnings from the literature research and on the aforementioned hypotheses. This prototype seeks to support users in fulfilling their exploration and querying tasks, to detect the expected and discover the unexpected.

The evaluation in Chapter 6 show that these requirements are met with the implementation of the three related data views, which communicate information effectively.The Sports Overview and Sports Detail show how closely related different sports are, and what the booking behavior was, also allowing the manual editing of similarity values. The Visual Query View makes it possible to define a cohort of users graphically by Drag&Drop and

retrieves suggestions for the cohort and also the individual users, without the analyst having to write complex queries themselves.

As the defined tasks can be neatly split into two categories, suggestion retrieval and exploration, we devised two types of views to be able to support both. The visualization design was validated by a domain expert who solved instances of these tasks.

According to the evaluation results, the query view allowed the user to intuitively define cohorts and read suggestions, but also to better understand which sports co-occur in users' booking patterns, positively answering the first hypothesis of our research question, **H1**, that interactive visual querying can be an effective tool to find alternative and complementary sporting activities. Also, the value of these suggestions can be assessed using the interaction techniques of the parallel set view of the cohort suggestions, where the cognitive load is reduced, confirming the third hypothesis **H3**.

For the exploratory tasks, much insight came from the similarity function that decided the layout of the sports bubbles, as well as the Spiral graph, which helped reveal expected and detect unexpected patterns in booking times for a sport, confirming the second hypothesis **H2**, that more insight can be provided by taking into account both the relational, as well as the temporal aspects of the data.

From this we can conclude that Visual Analytics supports the discovery of alternative and complementary sporting activities, answering the main research question.

We see our main contribution from a Visual Analytics perspective to be in line with the goals outlined in Keim et al. [KAF+08] of the creation of tools and techniques that enable people to

- Synthesize information and derive insight from massive, dynamic, ambiguous, and often conflicting data.
- Detect the expected and discover the unexpected.
- Provide timely, defensible, and understandable assessments.
- Communicate assessment effectively for action.

As a design study we bring contributions in the areas of problem characterization and abstraction and validated visualization design, in line with Sedlmair et al. [SMM12]

The benefits this paper brings are the development of a visual query language for the relevant subset of Neo4J's Cypher, which demonstrates the ability to construct graph queries significantly faster than using a conventional query language. This part is domain agnostic, and can be transferred to any problem if the data is transformed to a graph structure. As any real life problem can be described as a set of entities with relationships, this should be straightforward, although possibly time consuming.

During this thesis, a working prototype was developed. We had to limit the supported complexity of queries, like queries with several disjunctive paths sets, as explained in Section 5.4.

During the implementation and evaluation of the prototype, following suggestions for future work were made. From usability evaluation, we learned that some users would like more transparency regarding the "hidden" part of the query, the collaborative filtering pattern, to gain more insight in the data being processed to achieve the outputted result. Some also would like to make this configurable to gain more control. Another improvement was to be able to label and save the queries, so they can easily be run at a later time again. From a functional perspective, we wish to focus on making drawing more complex queries possible, with filtering on date ranges and allowing edges themselves to be negated. These improvements are great opportunities for future work.

Bibliography

[AMM+08] W. Aigner, S. Miksch, W. Müller, H. Schumann, and C. Tominski. Visual methods for analyzing time-oriented data. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):47–60, Jan 2008.

[BIJ02] Hannah Blau, Neil Immerman, and David D. Jensen. A visual language for relational knowledge discovery. Technical Report UM-CS-2002-37, Department of Computer Science, University of Massachusetts, Amherst, MA, 2002.

[Car08] Sheelagh Carpendale. *Evaluating Information Visualizations*, pages 19–45. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[CFT+08] Duen Horng Chau, Christos Faloutsos, Hanghang Tong, Jason I. Hong, Brian Gallagher, and Tina Eliassi-Rad. Graphite: A visual query system for large graphs. *2008 IEEE International Conference on Data Mining Workshops*, pages 963–966, 2008.

[CKHF11] Duen Horng Chau, Aniket Kittur, Jason I. Hong, and Christos Faloutsos. Apolo: Making sense of large network data by combining rich user interaction and machine learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 167–176, New York, NY, USA, 2011. ACM.

[CMS99] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[CT05] Kristin A. Cook and James J. Thomas. Illuminating the path: The research and development agenda for visual analytics. 2005.

[DGM15] Walter Didimo, Francesco Giacchè, and Fabrizio Montecchiani. Kojaph: Visual definition and exploration of patterns in graph databases. In Emilio Di Giacomo and Anna Lubiw, editors, *Graph Drawing and Network Visualization*, pages 272–278, Cham, 2015. Springer International Publishing.

[FWSN08] Jean-Daniel Fekete, Jarke J. Wijk, John T. Stasko, and Chris North. Information visualization. chapter The Value of Information Visualization, pages 1–18. Springer-Verlag, Berlin, Heidelberg, 2008.

[HHWN02] Susan Havre, Elizabeth Hetzler, Paul Whitney, and Lucy Nowell. Themeriver: Visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, January 2002.

[HWB10] Andreas Hartl, Klara Weiand, and François Bry. visKQWL, a visual renderer for a semantic web query language. *Proceedings of the 19th international conference on World wide web - WWW '10*, page 1253, 2010.

[KAF+08] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. *Visual Analytics: Definition, Process, and Challenges*, pages 154–175. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[KBH06] Robert Kosara, Fabian Bendix, and Helwig Hauser. Parallel sets: Interactive exploration and visual analysis of categorical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):558–568, July 2006.

[KMSZ06] D. A. Keim, F. Mansmann, J. Schneidewind, and H. Ziegler. Challenges in visual data analysis. In *Tenth International Conference on Information Visualisation (IV'06)*, pages 9–16, July 2006.

[KPS15] Josua Krause, Adam Perer, and Harry Stavropoulos. Supporting Iterative Cohort Construction with Visual Temporal Queries. 2626(c), 2015.

[MA14] S. Miksch and Wolfgang Aigner. A matter of time: Applying a data-users-tasks design triangle to visual analytics of time-oriented data. *Computers & Graphics, Special Section on Visual Analytics*, 38:286–290, 2014.

[MB] Luanne Misquitta and Michal Bachman. Graphaware neo4j timetree. `https://graphaware.com/neo4j/2014/08/20/graphaware-neo4j-timetree.html`. Accessed: 03 June 2017.

[Pla04] Catherine Plaisant. The challenge of information visualization evaluation. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, pages 109–116, New York, NY, USA, 2004. ACM.

[PW14] Adam Perer and Fei Wang. Frequence. *Proceedings of the 19th international conference on Intelligent User Interfaces - IUI '14*, pages 153–162, 2014.

[Shn96] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. *Proceedings 1996 IEEE Symposium on Visual Languages*, pages 336–343, 1996.

[SMM12] Michael Sedlmair, Miriah Meyer, and Tamara Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, 2012.

[SWS13] Rachel Shadoan, Chris Weaver, and Ieee Computer Society. Visual Analysis of Higher-Order Conjunctive Relationships in Multidimensional Data Using a Hypergraph Query System. 19(12):2070–2079, 2013.

[TAS04] Christian Tominski, James Abello, and Heidrun Schumann. Axes-based visualizations with radial layouts. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, SAC '04, pages 1242–1247, New York, NY, USA, 2004. ACM.

[TC06] James J. Thomas and Kristin A. Cook. A visual analytics agenda. *IEEE Computer Graphics and Applications*, 26:10–13, 2006.

[VJC09] Katerina Vrotsou, Jimmy Johansson, and Matthew Cooper. ActiviTree: interactive visual exploration of sequences in event-based data using graph similarity. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):945–52, 2009.

[WAM01] M. Weber, M. Alexa, and W. Muller. Visualizing time-series on spirals. *IEEE Symposium on Information Visualization, 2001. INFOVIS 2001.*, pages 7–13, 2001.

[WG12] Krist Wongsuphasawat and David Gotz. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2659–2668, 2012.

[wik] Category:sports by type. `https://en.wikipedia.org/w/index.php?title=Category:Sports_by_type`. Accessed: 10 December 2016.

‘

APPENDIX A

Sports categories

```
{ "name": "root",
  "subcategories": [
    { "name": "Acrobatic Sport" },
    { "name": "Ball Game",
      "subcategories": [
        { "name": "Ball and Bat Game" },
        { "name": "Cue Sport" },
        { "name": "Wall and Ball Game" }
      ]
    },
    { "name": "Beach Sport" },
    { "name": "Endurance Sport" },
    { "name": "Hybrid Sport" },
    { "name": "Fun Sport" },
    { "name": "Individual Sport",
      "subcategories": [
        { "name": "Boardsport" },
        { "name": "Combat Sport",
          "subcategories": [
            { "name": "Martial Art" },
            { "name": "Unarmed" },
            { "name": "Armed" }
          ]
        },
        { "name": "Cue Sport" }
      ]
    },
    { "name": "Indoor Sport",
      "subcategories": [
        { "name": "Cue Sport" },
        { "name": "Dancesport" }
      ]
    },
    { "name": "Outdoor Sport" },
```

```
    { "name": "Racquet Sport" },
    { "name": "Roller Sport" },
    { "name": "Team Sport",
      "subcategories": [
        { "name": "Ball and Bat Game" },
        { "name": "Equestrian Team Sport" }
      ]
    },
    { "name": "Throwing Sport" },
    { "name": "Water Sport",
      "subcategories": [
        { "name": "Towed Water Sport" },
        { "name": "Underwater Sport" },
        { "name": "Whitewater Sport" }
      ]
    },
    { "name": "Winter Sport",
      "subcategories": [
        { "name": "Ice Sport" },
        { "name": "Snow Sport" }
      ]
    },
    { "name": "Animal Sport",
      "subcategories": [
        {
          "name": "Equestrian Sport",
          "subcategories": [
            { "name": "Equestrian Team Sport" }
          ]
        }
      ]
    },
    { "name": "Extreme Sport" },
    { "name": "Motorsport" },
    { "name": "Precision Sport",
      "subcategories": [
        { "name": "Cue Sport" }
      ]
    }
  ]
}

```

APPENDIX B

Client-side visual pattern encoding structure

```
{
 "nodes": [
   {
     "id": Integer,
     "name":"Target Group",
     "type": "Person"
     "filters":[ FILTER_0, FILTER_1, ..., FILTER_n],
     "cnt":0
   },
   NODE_1, NODE_2, ...
 ],

 "links":[ LINK_0, LINK_1, ...LINK_n ]
}
```

where

```
NODE = {
       "id": Integer,
       "name": String [Booking|Category|City|Event|Facility|Person|Sport|OR],
       "type": String [Booking|Category|City|Event|Facility|Person|Sport|OR],
       "filters":[ FILTER_0, FILTER_1, ..., FILTER_n],
       "cnt": Integer,
},

FILTER = {
   "group": [ FILTERGROUP_0, FILTERGROUP_1, ..., FILTERGROUP_n]
   "negated": Boolean
},

FILTERGROUP = {
   "property":String,
   "operator": String,
   "value":[ String, String, ...]
},

LINK = {
       "source": Integer,
       "target": Integer
}
```

APPENDIX C

Server-side visual pattern encoding structure

```
paths = [ PATH_1 , PATH_2 , ... , PATH_n ]
```

where each PATH is of the form

```
PATH = {
   "matches": [MATCH_1 , MATCH_2 , ....MATCH_n ] ,
   "disjunct": int
}
```

where disjunct is the ID of the OR node, if there is one, so we can correctly assign the paths that need to be OR-connected.

Each MATCH has the form

```
MATCH = {
   nodes: [NODE_1 , NODE_2 , ... , NODE_n ] ,
   where:[ String , String , ... ] ,
   negated: Boolean
}
```

and each NODE is

```
NODE = {
   id: Integer ,
   alias: String ,
   type: String
}
```

APPENDIX D

Use case scenarios: data structures

D.1 People that live in Vienna and have attended events for the Sport Yoga

Client-side structure

```
{ "nodes": [
    { "id": 1, "name": "City", "filters": [], "type": "City", "cnt": 0 },
    { "id": 2, "name": "Event", "filters": [], "type": "Event", "cnt": 0 },
    { "id": 5,
      "name": 'Sport',
      "filters": [
        { "group": [
            { "property": "name",
              "operator": "= '{val}'",
              "value": [ "Yoga" ]
            } ],
          "negated": false
        } ],
      "type": 'Sport',
      "cnt": 0
    },
    { "id": 8, "name": "Target Group", filters: [], "type": "Person", "cnt": 0 } ],

  "links":
   [ { "source": 8, "target": 1 },
     { "source": 8, "target": 2 },
     { "source": 2, "target": 5 } ] }
```

Server-side structure

```
{ "paths":[
  { "matches": [
    { "nodes": [
      { "id": 8, "alias": "P8","type": "Person" },
      { "id": 1, "alias": "C1","type": "City" } ],
      "where": []
    } ]
  },
```

```
  { "matches": [
    { "nodes": [
      { "id": 8, "alias": "P8", "type": "Person" },
      { "id": 2, "alias": "E2", "type": "Event" },
      { "id": 5, "alias": "S5", "type": "Sport" } ],
      "where": [ "(S5.name = 'Yoga')"]
    } ]
  } ],
  "orConnectedAlias":[ ],
  "orConnectedIndex":[ ]
}
```

D.2 People that live in Vienna or Berlin and have bookings or events for a Racquet Sport

Client side structure

```
{ "nodes": [
  { "id": 1,
    "name": "City",
    "filters": [
      { "group": [
          { "property": "name",
            "operator": "= '{val}'",
            "value": [ "Wien, AT" ]
          } ],
        "negated": false
      },
      { "group": [
        { "property": "name",
          "operator": "= '{val}'",
          "value": [ "Berlin, DE" ]
        } ],
        "negated": false
      } ],
    "type": "City",
    "cnt": 0 },
   { "id": 2, "name": "Event", "filters": [], "type": 'Event', "cnt": 0 },
   { "id": 3, "name": "Booking", "filters": [], "type": "Booking", "cnt": 0 },
   { "id": 5, "name": "Sport", "filters": [], "type": "Sport", "cnt": 0 },
   { "id": 6,
     "name": "Category",
     "filters": [
        { "group": [
          { "property": "name",
            "operator": "= '{val}'",
            "value": [ "Racquet Sport" ]
          }],
          "negated": false
        } ],
    "type": "Category",
    "cnt": 0 },
   { "id": 7, "name": "OR", "filters": [], "type": 'OR', "cnt": 0 },
   { "id": 8, "name": "Target Group", "filters": [], "type": "Person", "cnt": 0 } ],
  "links":
   [ { "source": 8, "target": 7 },
     { "source": 7, "target": 3 },
     { "source": 7, "target": 2 },
     { "source": 3, "target": 5 },
     { "source": 2, "target": 5 },
     { "source": 5, "target": 6 },
     { "source": 8, "target": 1 } ] }
```

Server-side data structure

```
{ "paths": [
```

```
    { "matches": [
      { "nodes": [
        { "id": 8, "alias": "P8", "type": "Person" },
        { "id": 7, "alias": "O7", "type": "OR"} ],
        "where": []
      } ]
    },
    { "matches": [
        { "nodes": [
          { "id": 8, "alias": "P8", "type": "Person" },
          { "id": 7, "alias": "O7", "type": "OR" },
          { "id": 3, "alias": "B3", "type": "Booking" },
          { "id": 5, "alias": "S5x7_1", "type": "Sport" },
          { "id": 6, "alias": "C6x7_1", "type": "Category" } ],
          "where": [ "(C6x7_1.name = 'Racquet Sport')" ]
        },
        { "nodes": [
          { "id": 8, "alias": "P8", "type": "Person"},
          { "id": 7, "alias": "O7", "type": "OR"},
          { "id": 2, "alias": "E2", "type": "Event"},
          { "id": 5, "alias": "S5x7_2", "type": "Sport"},
          { "id": 6, "alias": "C6x7_2", "type": "Category"} ],
          "where": [ "(C6x7_2.name = 'Racquet Sport')"]
        } ],
      "disjunct": 7
    },
    { "matches": [
        { "nodes": [
          { "id": 8, "alias": "P8", "type": "Person" },
          { "id": 1, "alias": "C1", "type": "City" } ],
          "where": [ "(C1.name = 'Wien, AT') OR (C1.name = 'Berlin, DE')" ]
        } ]
    } ],
  "orConnectedAlias": [
    { "orId": 7,
      "nodeAlias": [
        { "nodeId": 5, "alias": [ "S5x7_1", "S5x7_2" ], "type": "Sport" },
        { "nodeId": 6, "alias": [ "C6x7_1", "C6x7_2" ], "type": "Category" } ]
    } ],
  "orConnectedIndex": [ 7 ]
}
```