FAKULTÄT FÜR !NFORMATIK

Interactive Visual Analysis of Relational Data and Applications in Event-Based Business Analytics

MASTERARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

im Rahmen des Studiums

Computergraphik/Digitale Bildverarbeitung

eingereicht von

Bilal Alsallakh
Matrikelnummer 0627567

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung:
Betreuer: Ao.Univ.Prof. Dr.techn. Eduard Gröller
Mitwirkung: Dipl.-Ing. Martin Suntinger

Wien, 20.07.2009

(Unterschrift Verfasser/in) (Unterschrift Betreuer/in)

Abstract

In this work, a framework for interactive visual analysis of attributed graphs has been developed. An attributed graph is an extension of the standard graph of a binary relation, which attaches a set of attributes to the nodes and edges. The implemented visual analysis techniques aim at the local level at enabling an intuitive navigation in the graph which reveals both the structure of the selected part of the graph and the attributes of the nodes and edges in this part. At the global level these techniques aim at understanding the distributions of the attributes in the graph as a whole or in specific parts in it and at spotting meaningful associations between the attributes and the relations.

The work presents several extensions to the attributes such as graph-theoretic features, values aggregated over the relations, and hierarchical grouping. All attributes are treated in a unified manner which helps performing elaborate analysis tasks using the existing tools.

Additionally, novel graph drawing techniques are proposed. They are designed to understand attribute distributions and associations in the graph. These techniques can be additionally used to visualize results of queries in the data, which can be also visually defined using the attribute analysis tools.

Finally, the work addresses several types of association analysis in relational data, along with visual analysis methods for them. It presents a perceptual enhancement for the well-known parallel sets technique for association analysis in categorical data, and proposes extensions for employing it in relational data. Also, novels methods for other types of association analysis are introduced.

The relational data in this work were defined upon typed events in an event-based system, which offers a flexible architecture for real-time analysis. Nevertheless, the presented analysis methods are generic and have been tested on two real-world datasets. In the first dataset, entities for customers and products are derived from the purchase events, and various meaningful associations were found between the attributes and the relation (for example, which types of products the female customers bought more frequently, or at which age customers have higher interest for books). In the second dataset, events in an issue-tracking system are analyzed to find out ticket assignment patterns and forwarding patterns between the support offices.

Keywords

Graph Visualization, Relational Data Analysis, Multidimensional Data Analysis.

Kurzfassung

Im Zuge der hier präsentierten Arbeit wurde ein System entwickelt, um attributierte Graphen zu analysieren. In diesen Graphen werden Attribute zu den Knoten und den Kanten zugeordnet. Das System bietet mehrere Dartellungen für die visuelle Analyse im Graph an, sowohl auf einem lokalen als auch einem globalen Niveau. Die lokale Analyse erlaubt eine Navigation im Graph um die Struktur und die Attribute von einem Teil des Graphs zu erfoschen. Die globale Analyse erlaubt, die Verteilungen der Attribute in dem graph (oder im ausgewälten Teil vom Graphen) zu verstehen, und bestehende Assoziationen zwischen den Attributen und den Relationen festzustellen.

Die Arbeit präsentiert eine Reihe von Erweiterungen zu den Attributen, die beispielsweise, graphtheoretische Merkmale, aggregierte Werte und hierarchische Gruppierungen repräsentieren können. Diese Attribute werden gleich wie die intrinsischen Attribute behandelt, und können mit den selben Methoden analysiert werden.

Zusätzlich präsentiert die Arbeit neue Methoden für das Zeichnen von Graphen, die speziell für attributierte Graphen geeignet sind, insbesondere um die Verteilung und Assoziation von Attributen zu betonen. Diese Methoden können sowohl für die Navigation im Graphen als auch für die Darstellung von Abfrageergebnissen angewendet werden.

Des weiteren erfoscht die Arbeit mehrere Arten von Assoziationen in relationalen Daten, und bietet Lösungen für die visuelle Analyse von diesen Assoziationen an. Dafür werden Erweiterungen für die bekannte „Parallel Sets“-Technik präsentiert, um die Perzeption zu verbesssern, und zusäzliche Informationen von den Daten einzubeziehen. Darüber hinaus werden neue Methoden für anderen Arten von Assoziationen vorgeschlagen.

Die präsentierten Techniken sind generalisierbar und lassen sich auf beliebige Ereignisdaten anwenden. Das System wurde anhand zweier realer Datensätze auf seine Einsatzfähigkeit geprüft. Im ersten Datensatz werden aus einem Ereignis „Produkt gekauft“ die Entitäten „Kunde“ und „Produkt“ abgeleitet samt ihrer Attribute (wie Kundenwohnort u.Ä) und Relationen (die durch die Historie der Käufe entstehen). Das System erlaubt in der Folge eine Reihe von Analysen um beispielsweise festzustellen, welche Arten (Kategorien) von Produkten vermehrt von Damen gekauft wurden, oder welche Altersgruppe besonderes Interesse für Bücher hat. Im zweiten Datensatz werden Ereignisse aus einem sogenannten Issue-Tracking-System analysiert um festzustellen, wie Support-Fälle bearbeitet und auch zwischen Support-Büros delegiert werden.

Keywords

Graph Visualization, Relational Data Analysis, Multidimensional Data Analysis.

Erklärung zur Verfassung der Arbeit

„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 am 20.07.2009

Acknowledgement

I would like to thank all the people who helped me doing this work. First I would like to thank my supervisors, Prof. Eduard Gröller and Dipl.-Ing. Martin Suntinger, for the valuable discussions and guidance throughout the work. I would like also to thank Dr. Josef Schiefer for his reviews and inspiring remarks, Senactive Inc. for hosting the work, and the developers at Senactive: Florian, Heinz and Philip for helping in the implementation, and Markus, Michi, and Ulrich for providing rendering tips. Further thanks go to Mohamad Tayssir Alkowatly and Peter Kepplinger who helped in solving the equation for enhancing parallel sets, and to the visualization group at TU Vienna, in particular Peter Rautek and Stefan Bruckner, for their great feedback. I would like to express my gratitude to the Mondi Austria Privatstiftung for financing my master studies, and the ÖAD for organizing my stay in Austria. Finally, special thanks go to my mother and sister for their continuous encouragement and support.

Bilal Alsallakh

Table of Contents

Chapter 1

Introduction

"Setting goals is the first step in turning the invisible into the visible"
- Anthony Robbins

1.1. Motivation

Binary relations between entities, and their corresponding graphs, are one of the most fundamental modeling tools in many areas of business and science. With their high level of abstraction, they can represent diverse concepts such as social networks (club membership, friendship, co-authorship, and family relationships), biological concepts (gene regulatory, protein-protein interaction), and business concepts (organization structures, document flow) to mention a few.

The graphs of these relations can be enriched with many extensions, which contribute to their power of modeling. One such extension is multi-graphs, which allow multiple relations to exist between two entities. Attributed graphs are another such extension which attaches a set of attributes with the relations between the entities which themselves can also have their own attributes. The underlying relational data are hence called *attributed relational data*.

For example in a relation which links customers to the shops they have bought from, a link between a customer and a shop can bear information about a purchase this customer made at this shop (such as the purchase date and total price).

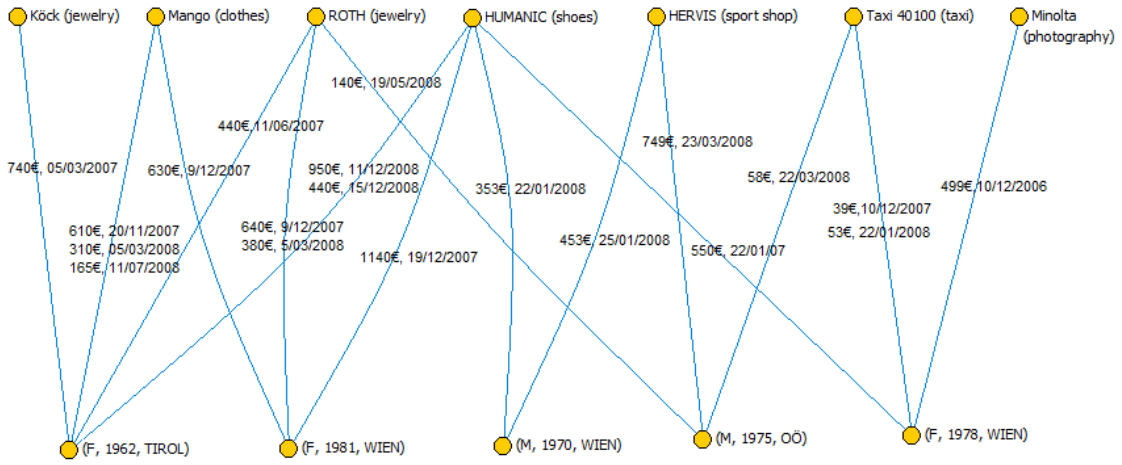

Fig. 1: An attributed graph

Also, the customer node may hold customer information (gender, birthday, living place), and the shop node can bear shop name and category. Fig. 1 shows an example of such a graph. Multiple links can exist between a customer and a shop to represent multiple purchases.

Attributed graphs are rich data models that contain information of potential interest about the problems at hand. Besides graph-theoretic features, information about the attributes and how they interrelate with the relations enables better understanding of the data.

This work aims at designing and implementing interactive visual analysis methods for attributed relational data and their graphs, that focus into visual exploration of the graph and analyzing the interrelation between the attributes and the structure of the graph to find meaningful associations between them.

The remainder of this chapter presents a formal definition of attributed graphs, the purpose of the visual analysis implemented in this work and the problems it addresses, the environment this work was targeted for, and two examples applications addressed throughout the work. In chapter 2, related work in the areas of graph drawing and visual analysis of multivariate data is discussed, and existing approaches that have been employed in this work are presented.

Chapter 3 explains how the attributes are modeled in this work, and how the attribute model is extended with additional attributes that enable performing more elaborate analysis tasks using the existing tools. It also presents attribute distribution analysis and how it can be used to map attributes to visual properties that used in other modules, and to visually define queries that select a subset of the entities. Chapter 4 elaborates on graph drawing for visual exploration in the graph, and presents novel visualization approaches for attributed graphs. It also presents how these views can be used to visualize results of queries in the graph.

Chapter 5 presents the three types of associations that can be visually analyzed in this work, namely attribute-attribute, attribute-relation and relation-relation associations. Chapter 6 provides details about the implementation and finally, chapter 7 presents evaluation results, discusses future extensions and enhancement, and concludes the work.

1.2. Attributed Relational Data / Attributed Graphs

Formally, attributed relational data are defined as follow:

$$R = \left(E_1, E_2, G \subseteq E_1 \times E_2, A_{E_1}, A_{E_2}, A_G\right)$$
$$A_{E_1} = \left\{a_{i,E_1}\right\}_i, a_{i,E_1}: E_1 \longrightarrow S_{i,E_1}$$
$$A_{E_2} = \left\{a_{i,E_2}\right\}_i, a_{i,E_2}: E_2 \longrightarrow S_{i,E_2}$$
$$A_G = \left\{a_{i,G}\right\}_i, a_{i,G}: G \longrightarrow S_{i,G}$$

Where:

- E_1 and E_2 are the domain and co-domain of the relation. We call these sets, *entity types**, and elements of these sets *entities* (or *nodes* in the context of graphs).
- G is a multiset of relations (to account for multiple edges).
- A_{E_1}, A_{E_2} and A_G are the sets of the attributes of the entity types E_1 and E_2, and the relation G respectively.
- a_{i,E_1} is the i[th] attribute of E_1, and S_{i,E_1} is the target set for this attribute.
- For each of the sets E_1, E_2 and G, an attribute $ID_X \in A_X, X \in \{E_1, E_2, G\}$ is defined by an injective mapping to denote the keys for the values in this set.

The corresponding attributed multi-graph is defined as $(E_1 \cup E_2, G)$. We can distinguish between two types of attributed relations depending on E_1 and E_2:

- $E_1 = E_2$: The relation is between elements of the same type. We call the corresponding graph, a *1-mode graph*, or *1-mode network.*
- $E_1 \cap E_2 = \emptyset$: The relation is between elements of different entities. We call the corresponding graph a *2-mode graph* or *2-mode network.* This graph is bipartite.

In this work the case of $E_1 \neq E_2 \wedge E_1 \cap E_2 \neq \emptyset$ is reduced to the first case by transforming the relation R to R' where:

$$R' = (E, E, G, A_E, A_E, A_G) : \; E = E_1 \cup E_2, \qquad A_E = A_{E_1} \cup A_{E_2}$$

The bipartiteness of the graph has an impact on the visualization and can be exploited to produce a clearer layout than in the case of general graphs. Chapter 2 mentions existing approaches for this purpose, and chapter 4 presents a novel approach developed in this work for the same purpose.

To simplify the visualization in the graph drawing module, the graph is treated as an undirected graph. This is straightforward to justify for the case of bipartite graphs, since it causes no loss of information as all the edges adjacent to a vertex have the same direction determined by the vertex class. In case of 1-mode graphs, this restriction is compensated by adding the direction as an attribute of the edge that can be mapped to its visual properties. All other modules (association analysis, attribute modeling and distribution analysis) take the edge direction into consideration.

* In this work we abbreviate the term *entity type* to *entity* when no confusion occurs.

1.3. The Purpose of the Visual Analysis of Relational Data

In real-world applications, the attributed graphs used for modeling usually contain large number of nodes (tens of thousands or even millions). The visual analysis aims at providing better understanding of these graphs both at the global and local levels, with main focus on how attributes are distributed and interrelated in the graph.

At the local level, the visual analysis aims at providing an interactive exploration utility which provides a detailed view of the selected part of the graph, showing all nodes in this part, how the nodes are linked, and how attributes are distributed in it. These views also provide interaction facilities such as exploring attributes of specific nodes, selection and filtering, and finding paths and commonalities, to mention a few. The exploration is aided with an animated navigation facility to explore other parts of the graph.

At the global level, the visual analysis focuses on the global distribution of the attributes, and on exploring associations between them based on the structure of the graph, i.e., how the relations are related to the attributes of the nodes. In particular the analysis helps to find nodes with some value of a specific attribute, which nodes they tend to be related with, and what values do these nodes have for a specific attribute. This helps in discovering existing patterns in the relation, and establishes which attribute values are positively or negatively correlated.

Additionally, the visual analysis aims at providing a method to perform visual queries in the graph using the utilities in both of the two levels of analysis. At the global level, the user can define the queries based on the values of the attributes of interest. The query results are shown at the local-level exploration utilities, with all the data in the query results shown as graphical elements, optionally along with part of the subgraph they induce. All the functionalities provided in the local-level analysis can be used to analyze the query results.

The two levels can be linked in the other direction too: The analysis utilities at the global level can be applied to the data at the local level and the results can be compared with those of the global data. Fig. 2 depicts an overview of the visual analysis tools and how they are related with each others.

The above mentioned generic functionalities provide a framework for the analysis of attributed graphs. It can be extended with further application-specific functionalities. In this work, we examine an extension to analyze the flow of support tickets in a customer support system, with the purpose of optimizing the ticket assignment.

Finally, visual analysis enables switching back from the relational data being analyzed to possibly underlying data upon which the relational data were defined. This makes sense when the visual analysis is integrated as a module in a business intelligence application which has its own data models. In this work, the relational data analysis is implemented in an event-based system. The next section explains how events are mapped to relational data in such a system.

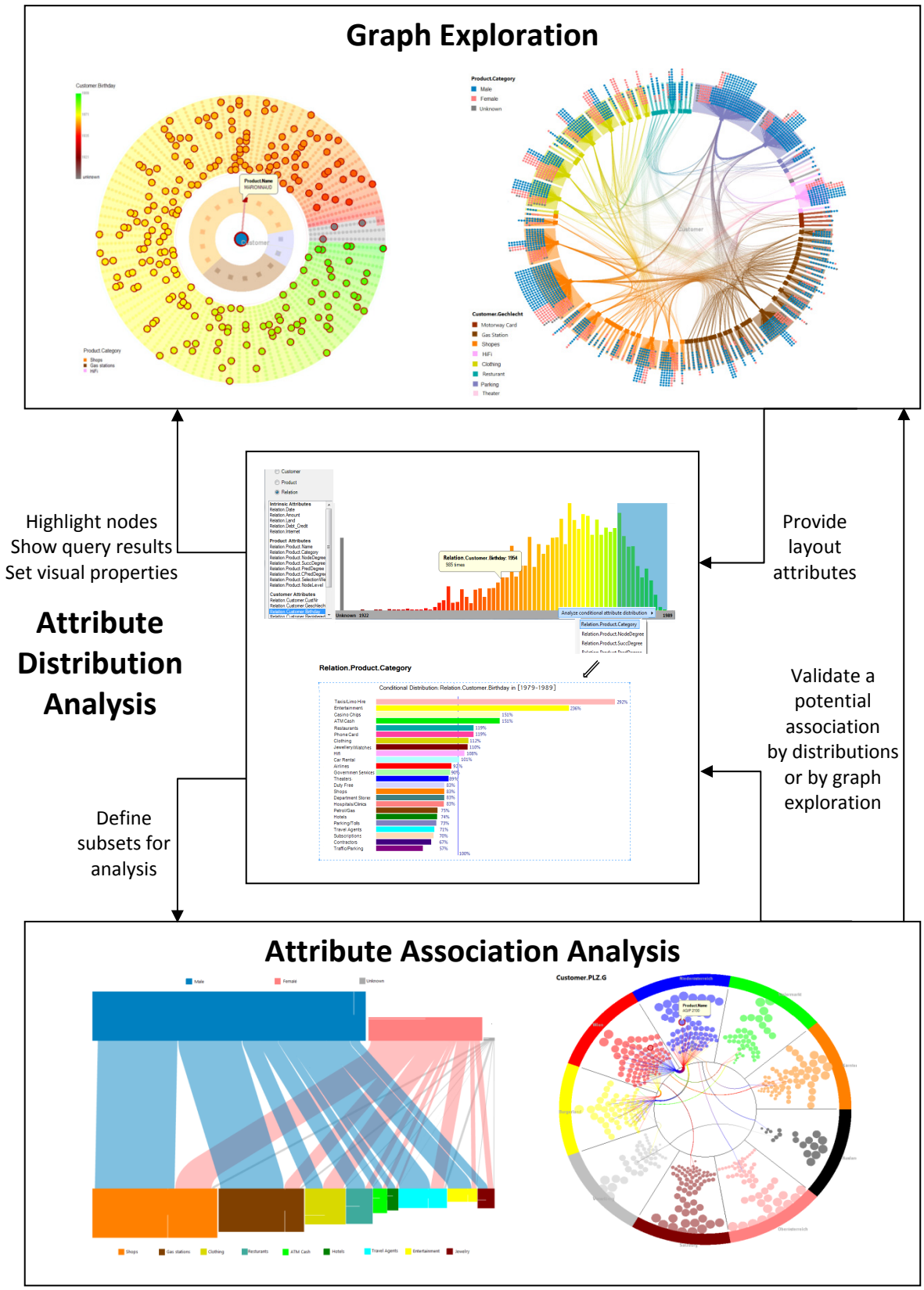

Fig. 2: Overview of the visual analysis

1.4. Relational Data in Event-based Systems

In an event-based system, the system receives events, processes them and responds to them accordingly. Events are used to represent many concepts in the real world; they can be triggered by actions taking place in the system (e.g. opening an account in a web application, sending a message between two users, etc), or they can be triggered by conditions that are met (e.g. a mobile phone user is detected in a new zone, a fraudulent behavior is recognized in a betting system, or an error is generated in a business process). An event-based system is designed to receive these events from the external applications via adapters, and respond accordingly. For example, the system might respond to a newly detected mobile phone user from a partner network with a welcome message and provide information about roaming tariffs, and respond to account creation by sending a welcome-email.

Due to their simplicity and ability to perform asynchronous processing, event-based systems have emerged as a highly flexible backend solution for integrating distributed applications, and offering real time processing.

SENACTIVE InTime™ is one such system, and was used as the design and execution environment for this work. Fig. 3 depicts the SARI (sense and respond infrastructure) architecture of SENACTIVE InTime™. The system continuously senses for relevant business events. Captured events are then interpreted and analyzed. Based on rules, decisions are made and finally fed back into the business environment in the respond step. This architecture enables automated near real time decision making at an operational level.

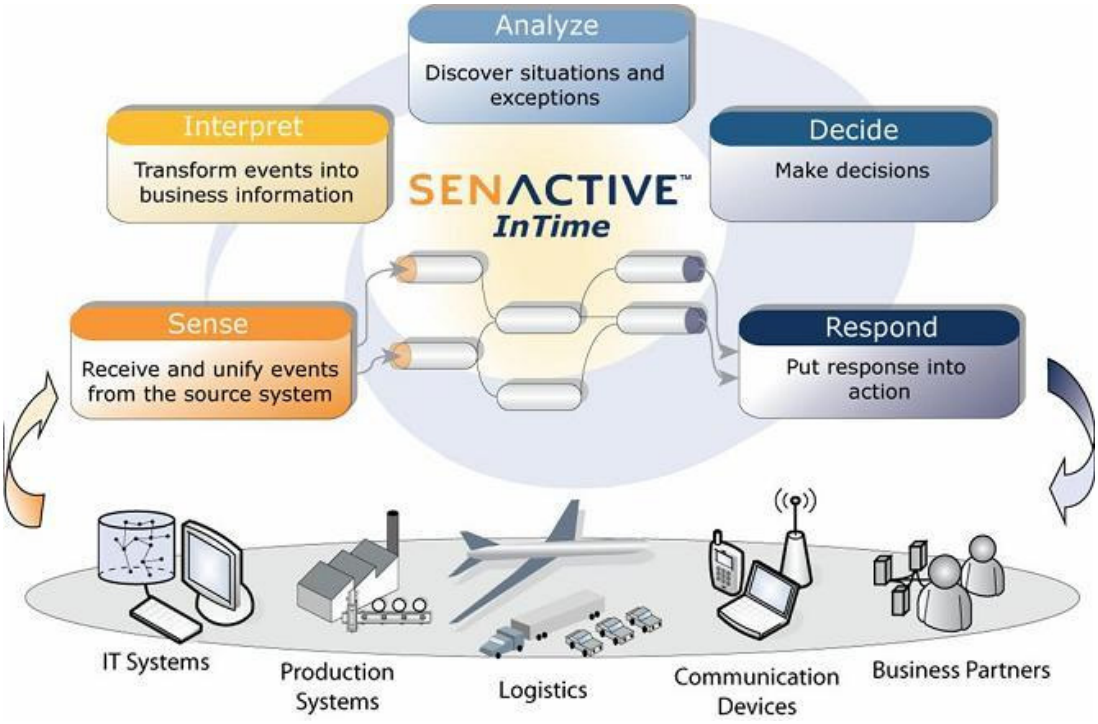

Fig. 3: The Senactive InTime™ event processing system (courtesy of Senactive Inc.)

Various models were introduced for representing and processing events. Rozsnyai et al. [RSS07] discussed the concept of event types, and how the events can be linked and correlated. In this model, each event belongs to one of a set of user-defined types and bears a set of attributes based on its type. The event type determines the structure and data content of the event, i.e., its attributes and their types. Events can be defined to be correlated based on their attributes. Fig. 4 shows example of typed events in Senactive InTime and how they are correlated.

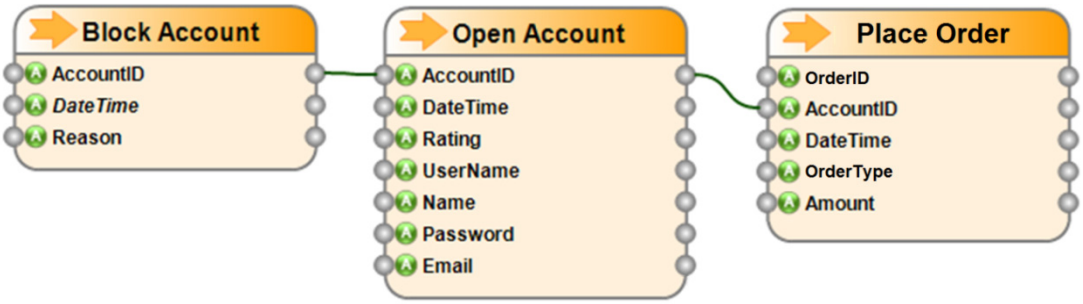

Fig. 4: Typed events and Event Correlation in Senactive InTime

Events are atomic incidents of purpose and are used to model various aspects in real-world applications. However, the relation addressed in the analysis might be between two entities that are not directly represented by two corresponding events, but rather, these entities represent more abstract concepts that are realized by means of several events. For example no event might represent the entity “customer”, but several events like “open account”, “block account” etc. represent partial aspects of it that participated in defining this entity. For this purpose, this work defines entities and relations as stand-alone analysis concept that can be integrated within event-based systems.

Each entity type has a name and a set of attributes that are defined upon the related events. Relations can be defined between entity types, and likewise, these relations can have attributes that are defined upon suitable events. Chapter 3 presents in details how the entity types and relations are defined upon events, and how the entities and the relations between them are actually created upon receiving new events.

For each pair of related entity types, the entities that belong to them and the relation instances between these entities constitute the relational data defined in section 1.1 (where the attribute sets are based on the attributes of the two entity types and the relation between them). The analysis tools presented in this work are intended to understand how these entities are related and what associations exist between their attributes and relations, which in turn gives insight into the events they are based on.

Entities and relationships are familiar concepts for business analysts, and can be defined intuitively upon events, which makes them potentially useful modeling tools for many problems at hand.

1.5. Example Applications

In this work, we used two example applications to demonstrate and validate the presented approaches. The evaluation of these examples is discussed in chapter 7.

1.5.1. Customer-product relationships

In this example, we analyze a real-world dataset from a credit card provider that includes nearly 80.000 transactions made by about 4000 customers in about 8000 shops/service providers in Austria. The transactions do not specify detailed purchases, rather total purchases. Therefore we consider the shops or the service providers as "products" subject to analysis. This makes sense in case of specialized shops or service providers which offer homogenous products/services (travel agents, jewelry shops, taxi services, etc.) which is the case in this dataset. This means, we consider all products from the same shop to be one entity in the analysis, which has the advantage of reducing the amount of data, and provides reasonable clustering of otherwise over-detailed data. Each product has its name (shop/service name) and a category specified by the domain expert.

The cards provide information about the customers. This includes the gender, date of birth, postal address, payment facilities, and registration date. Each transaction is mapped to a relation between the customer and the product, which has a set of attributes that specify the purchase date, location, total price, whether it is performed by internet or not, and whether it was debt or credit transaction.

Basic analysis of this dataset involves exploring distributions of the attributes to get an initial understanding of the dataset. It also comprises the exploration of adjacency relations over multiple levels for a specific customer / product, and finding all customers who bought all specified products, and vice versa, and finding whether these customers have common attributes.

Of special interest for marketing analysts is to check if there is an association between the customer attributes and his/her shopping patterns, and associations between the products. Examples of such associations are:

- Clothes and jewelries are bought more frequently by women than by men.
- Travel agents are used more frequently by older people than by younger ones.
- Customers from Styria* who used gas stations are mostly men.
- Customers who used gas stations are more likely to buy parking coupons.
- Young women who bought jewelries are more likely to buy luxury watches.
- Hotel reservations by customers from a specific province in Austria take places more often in specific months of the year.

* One of Austria's nine provinces.

Also, analysts are interested in validating a given segmentation of the market according to the customer attributes. For example, the analysts check which products are more popular in each customer group, and how these groups are related by their favorite products, i.e., if two groups share a large part of their favorite products.

1.5.2. Support Ticket Assignment

In this example, we analyze a data set of support tickets provided by the customer support department in an Austrian company. Support tickets are issued by the company customers either to ask technical questions, or to report problems. The company has about 150 support offices in Austria, Czech Republic, Germany, Hungary, Romania and Slovakia. These offices are hierarchically organized. Each customer ticket is assigned to a support office which has to resolve the issue. Until resolved, the ticket undergoes a process which changes its status among possible values (open, accepted, work_in_progress, pending, closed, reopen, reassigned, resolved).

Of particular interest is the ticket re-assignment, which changes the assignee office. We defined the relational data by choosing the two entity types to be ticket and assignee, where by assignee we mean a support office. A relation between a ticket and an assignee is created when the ticket is assigned to this assignee or when its status changes.

The basic analysis enables exploring for each ticket the assignee(s) it has been assigned to, the current status, and the process it has undergone, and for each assignee the tickets it has supported and what their current states are. A more elaborate analysis allows establishing any association between type of tickets being processed and the support office they are assigned to. This helps in understanding existing tendencies in choosing a support office for a given ticket in the assignment process.

Of main interest for the analysis in this example is to understand the ticket re-assignment process, i.e., the paths tickets are following when they are reassigned. The goal is to establish any existing patterns the tickets follow in order to optimize the ticket assignment process. It would be more efficient to assign tickets directly to the appropriate person and avoid as many re-assignment events as possible. This helps reducing the support time and cost.

Chapter 2

Related Work

"Copy from one, it's plagiarism; copy from two, it's research" - John Milton

This work combines ideas from three areas that have been extensively researched. This chapter presents the developments in these areas and discusses other approaches with which the problems addressed in this work have been tackled.

2.1. Graph Drawing

Graph drawing has been an active research topic since the 1960s. It has first drawn the interest of mathematicians and graph theorists who researched topics such as graph planarity and planar embedding. Tutte [T63] was the first to devise an algorithm for planar drawing of 3-connected planar graphs in 1963, based on a simple force-directed formulation which places each node in the center of gravity of its neighbors. In the 1980s, with the growing graphical capabilities and the wide utilization of graphs in computer science problems and software modeling, the interest in graph drawing grew further, and computer scientists participated in developing more advanced algorithms for this task. Eades [E84] extended the force-directed approach to layout general graphs, by simulating a physical system in analogy to the graph, where edges are replaced by springs, and nodes by balls of the same electric charge. In any state of this system, the springs try to shorten the edges, while the charges prevent the nodes from collapsing to a single point. The system is iteratively simulated until equilibrium.

Later on, Kamada and Kawai [KK89], and Fruchterman and Reingold [FR91] introduced heuristics and enhancements to this layout. Fig. 5 shows three graph drawings using the force-directed layout.

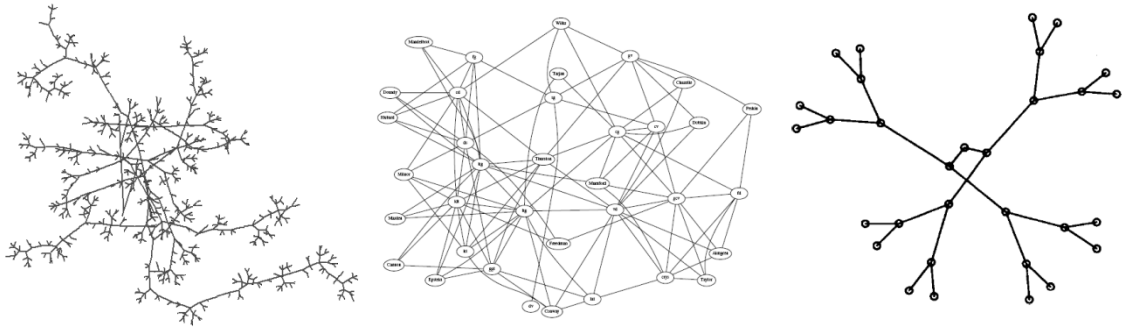

Fig. 5: Force-directed layouts ([QE01], [GN98] and [FR91])

Various aesthetic criteria for graph drawing have been investigated to maximize cognition such as [D07]:

- Edge crossing minimization (and no crossing in case of planar graphs).
- Bend minimization: an edge should make as few bends as possible.
- Area minimization: i.e., compact drawings with even distribution of the nodes.
- Length minimization: making edges as short as possible.
- Angle maximization: maximizing the angle between edges leaving a node.
- Symmetry: existing symmetries in the graph should be preserved.
- Clustering: existing clusters in the graph should be revealed in the visualization.

The force-directed methods produce drawings that tend to perform rather well according to these criteria (except for angle maximization) without implicitly checking for them, which, besides their simplicity and speed, has made them one of the most popular methods for generic graph drawing [GN98].

In addition to force-directed approaches, several other layouts have been devised for many types of graphs and applications. For example, Hasse diagrams were applied to layout directed acyclic graphs, and a variety of layouts has been introduced for process diagrams, flow charts and similar graphs in the area of software engineering. Also, various layouts dedicated to trees have been devised (such as the Radial Layout, TreeMaps, Balloon Layout, Rooted Tree Layout, etc). The radial layout places the root of the tree in the center, and places its children on a circle around it, and continues placing their children successively on larger circles [Fig. 6]. The sector of each node is divided between its children depending on their sub-tree widths, The Radial Layout has the advantage of intuitively communicating the tree-distance between each pair of nodes.

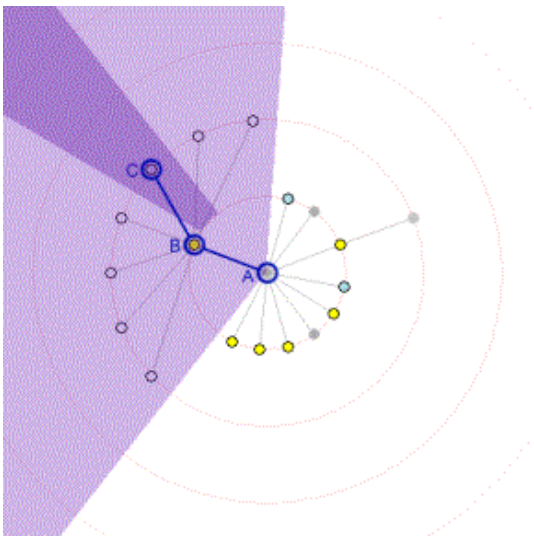

Fig. 6: The radial layout [YFDH01]

2.1.1. Large graph drawing

With many applications that require analysis of large graphs, the challenge of graph drawing increases not only because of the limited capabilities of graphical systems, but also because the readability suffers quickly with the growing size of the graph in typical layouts. Many approaches have been developed to reduce clutter and gain insight in large graphs, most of them based on clustering and level-of-detail exploration of the graph, edge bundling, and filtering.

Various schemes to perform clustering have been researched such as geometric clustering presented by Quigley and Eades [QE01], the multi-scale method of Harel and Koren [HK00] and the multi-dimensional approach of Gajer et al. [GGK00].

Geometric clustering (as opposed to graph-theoretic clustering) is performed by first mapping the graph nodes to their locations, and then applying a clustering algorithm such as k-means [M67] on the nodes according to their locations. By iteratively repeating this process, this approach creates a hierarchical clustering for the graph and enables level-of-detail exploration in it. The graph is layouted by a modified version of the force-directed layout, which restricts the direct force interaction between non-connected nodes to a local neighborhood, while distant nodes interact indirectly by means of their clusters, resulting in a significant reduction in computation time.

Edge bundling is a common method to reduce clutter and enhance readability in a variety of layouts. In fact, clutter in many layouts is largely caused by edges. Gansner and Koren [GK06] introduced a method to perform bundling in circular layouts, where the nodes are drawn on the circumference of a circle. Instead of leaving the edges drawn as straight lines (as in [Fig. 7-a]), non-crossing edges which link between nearby nodes are bundled to free-up drawing space [Fig. 7-b]. Additionally, the nodes are ordered to minimize edge crossing, which makes the bundling more effective. Optionally some links can be routed outside of the circle in the presented method [Fig. 7-c]. Another method for edge bundling is discussed in section 2.1.3.

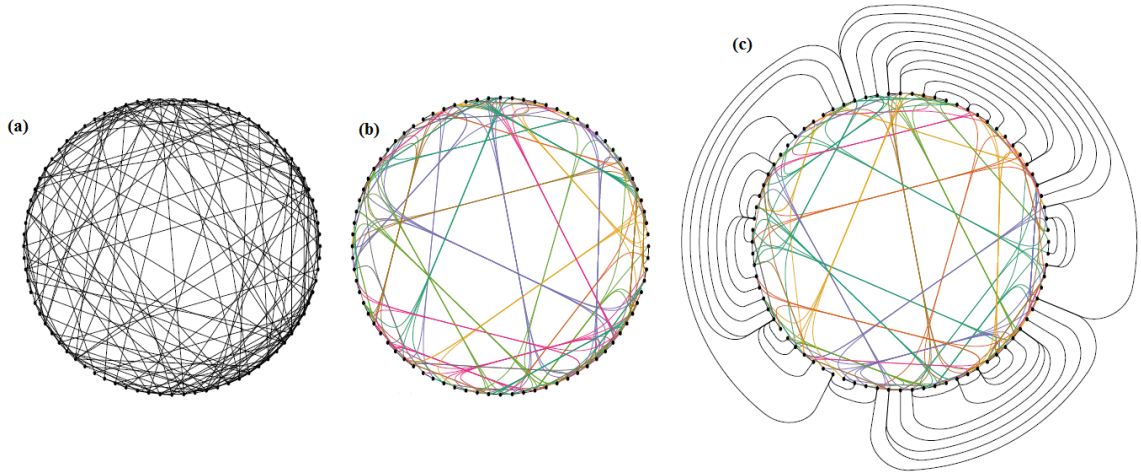

Fig. 7: Edge bundling in circular layouts [GK06]

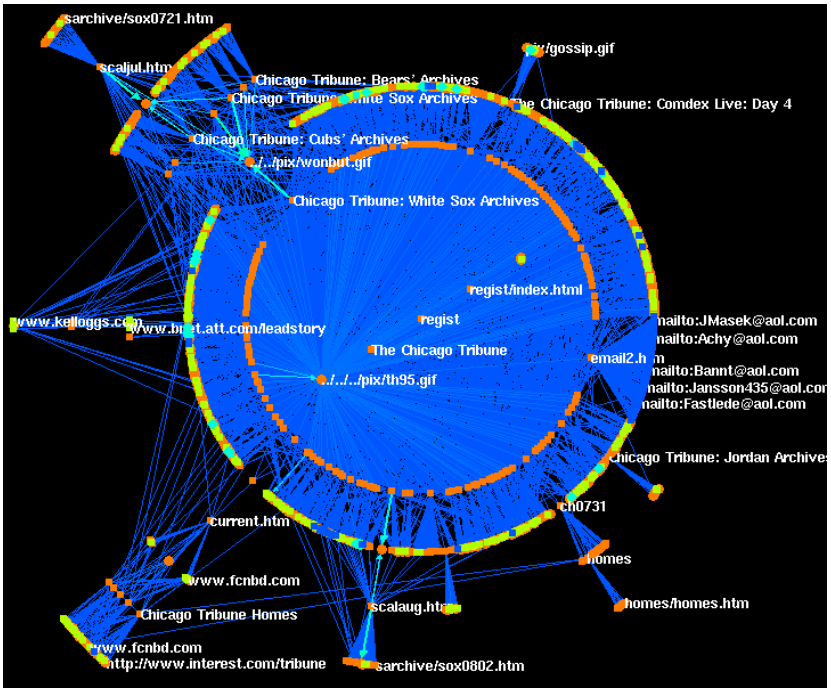

Fig. 8: Visualization of attributed graphs in NicheWorks [W97]

Another approach to reduce the clutter is to perform attribute-based filtering of nodes and edges. The readability improvement depends on the distribution and the semantics of the attributes used for filtering. Yet, this approach does not handle the problem in essence and is limited to attributed graphs. The NicheWorks system [W97] uses both attribute-based filtering and mapping of attributes to visual properties to visualize large graphs [Fig. 8].

Besides focusing on drawing large graphs as a whole, approaches were developed to enable navigation in these graphs, viewing only a portion of the graph at once but in details. In such a view, usually one node is the center of navigation and its neighborhood subgraph is visualized, and the graph can be explored by changing the center of navigation.

Yee et al. [YFDH01] developed one such approach, which treats the neighborhood subgraph of the central node as a tree rooted at this node in order to layout it using the radial layout [Fig. 9]. For this purpose, it chooses a spanning tree of this sub-graph according to a user-specific criterion. After performing the layout, the nodes, the tree-edges and optionally non-tree edges are drawn. To perform navigation, the user can choose a visible node to be the new central node, and the system computes the new subgraph to be drawn and layouts its nodes in an orientation which minimizes the angular distances between their old and new positions. To keep the continuity of perception, the presented method animates the node from their old positions to their new ones by linearly interpolating the old and new polar coordinates which is more compatible with the cognition scheme of the radial layout than interpolating the Cartesian coordinates.

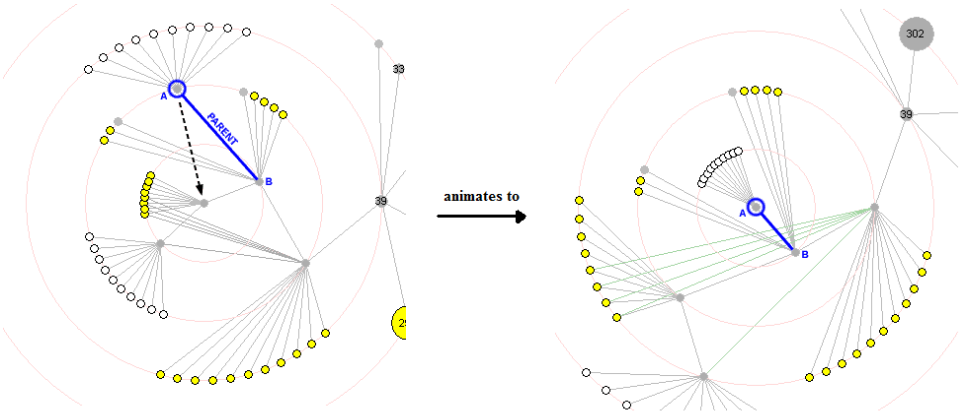

Fig. 9: Radial Layouts for animated exploration of graphs [YFDH01]

2.1.2. Attributed graphs

Apart from showing the relations between its elements, a visualization of the graph can also show attributes of these elements. Huang [H01] presented a formalization of attribute visualization by dividing the visualization process into geometric mapping, which is responsible for computing the placement of the nodes and edges, and graphical mapping which assigns the visual properties to each element (shape, size, color and opacity of nodes and edges). Fig. 10 shows an example graph of web pages and links between them. The node size represents the amount of direct traffic to the corresponding page, and the edge thickness represents the amount of traffic between the connected pages.

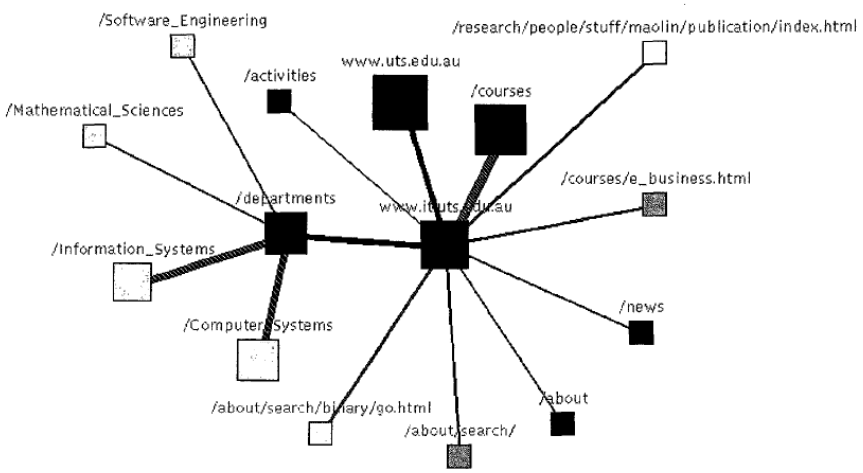

Fig. 10: Using size and color to visualize attributes in an attributed graph [H01]

Several systems have been developed to visualize attributed graphs such as UCINET [J87], Pajek [NMB04], GraphViz [EGKNW01], and VizSter [HB05]. They employ various graph drawing layouts, and allow the user to customize the visual mapping and perform exploration in the graph and several other analysis tasks.

2.1.3. Bipartite graph drawing

As mentioned in section 1.1., a bipartite graph can represent a relation between two different sets (entity-types). This fact can be exploited to produce a more efficient visualization which emphasizes the distinction between the two sets of nodes. Bipartite graphs play a major role in 2-mode social network analysis, where memberships of people in clubs / organizations are analyzed. Borgatti [B09] discussed visual analysis of these networks, and used a force-directed layout for this purpose. Different shape and color are used for each mode (entity-type) as depicted in Fig. 11.

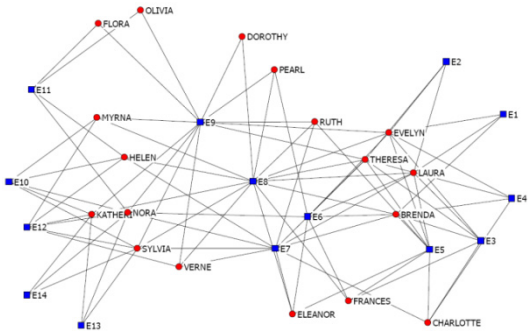

Fig. 11: a 2-mode network with force-directed layout [B09]

Misue [M08] argued that force-directed methods are inadequate for bipartite graphs and do not sufficiently exploit the bipartiteness. He presented the Anchor Maps approach to visualize bipartite graphs and applied it to the authors-papers graph.

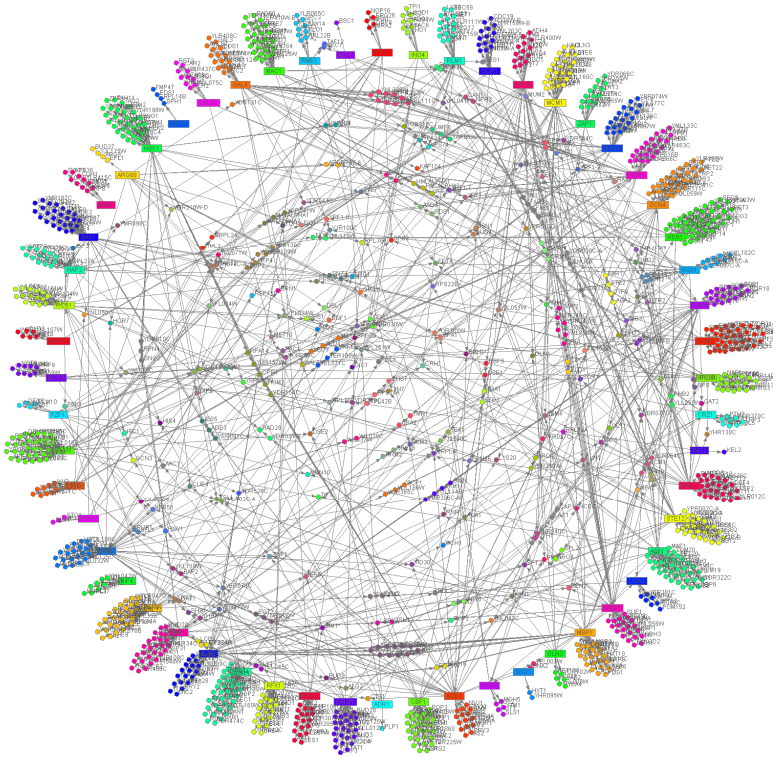

Fig. 12: Anchor Maps [M08]

In this approach, one of the two sets of nodes are placed evenly on a circle and used as “anchors” for the other set of nodes. Each node from the other set is placed in a suitable location depending on the anchors it is linked with. The published paper did not provide details about how this placement is performed. Anchor nodes are ordered so that the anchor nodes that share a lot of nodes from the other set are placed close to each other. This helps in revealing clusters of anchor nodes (local-communities), and shows for each non-anchor node, if it belongs to a local community or is otherwise linked to different ones. Fig. 12 shows an example of anchor maps. In this example, one can see the nodes representing the authors distributed along the circumference, the papers that are exclusive to each author in the neighborhood of the corresponding author’s node, and papers shared between more than one authors located inside the map.

Another technique used particularly in social network analysis is to convert a 2-mode network into a 1-mode network whose nodes are the nodes of either mode. In the 1-mode network a weighted link connects each pair of nodes, with the weight equals the number of nodes from the other mode that are linked to both of them in the 2-mode network. Vana [V07] applied this technique to show how products are interrelated depending on customer ownerships. He used a circular graph layout with thick links corresponding to high product co-ownership [Fig. 13].

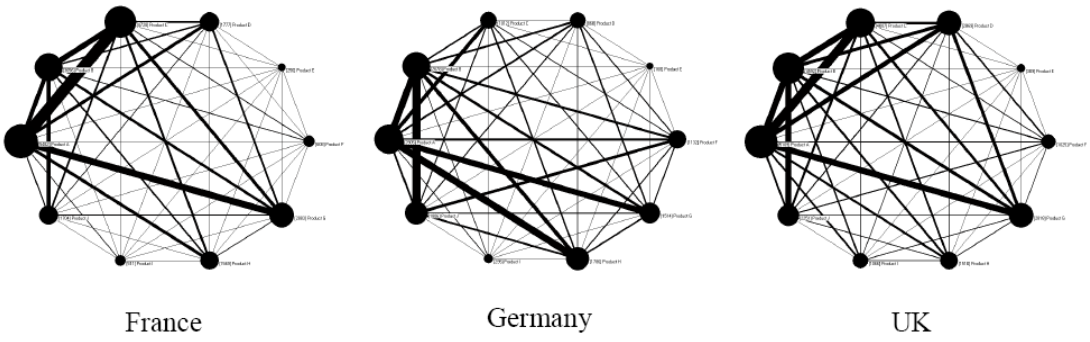

Fig. 13: Product-co-ownership in three countries [V07]

2.1.4. Hierarchical edge bundling

For a given graph $G = (V, E)$, where V is the set of nodes, and E is the set of edges, a hierarchy T can be defined over the nodes. For example, in the ticket assignment application (section 1.4), the nodes V can be chosen to be the support offices, and they additionally organized by a hierarchy T depending on their region and services. The support tickets are exchanged between these offices to form the edges E of the graph. To understand how the relations are distributed in the hierarchy, a visualization which combines both the graph and the hierarchy is needed.

One method to perform this visualization is to first draw the hierarchy using a suitable tree layout and then draw links between its leave nodes. Drawing links as straight lines often result in a cluttered visualization [Fig. 14-a]. Holten [H06] presented a technique to reduce clutter by means of bundling the edges according to the hierarchy [Fig. 14-b]. He also suggested the use of an inverted radial layout which inverts the hierarchy nodes outwards to prevent interference between the hierarchy and the relations, which produces a clearer view.

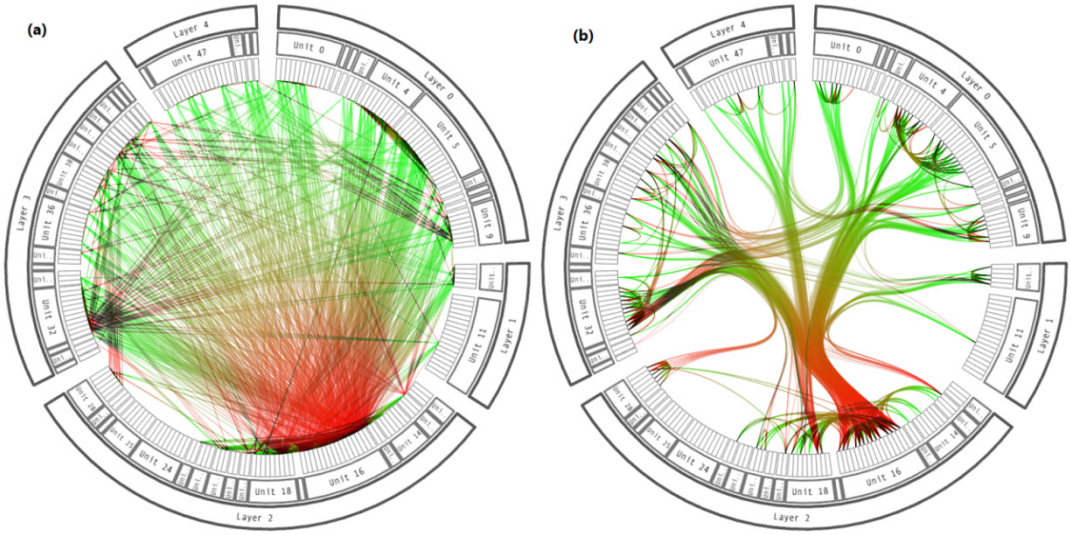

Fig. 14: Visualizing relations in hierarchical data [H06]. a) Drawing links as straight lines in a radial layout. b) Bundling links in an inverted radial layout.

2.2. Visualization of Multidimensional Data

Analysis of multidimensional data (multivariate data) has been one of the central topics in data mining and knowledge discovery in databases. Each data item is represented by a tuple of values in a multidimensional space, where the values are typically numerical or categorical. Typical analysis tasks are to understand distributions, find correlations between the dimensions, and perform clustering, to mention a few.

Histograms are one of the most fundamental tools in the visual analysis of multidimensional data. They enable understanding the frequency of items for each possible value (or range of values) of a given dimension (variable). A variety of extensions have been introduced to histograms to enrich their informative value, such as incorporating a conditional distribution of the histogram variable or incorporating another variable in the analysis. For example, stacked histograms show inside the bar for each possible value of the dimension at hand, the conditional distribution of the values of another dimension. Fig. 15 shows stacked histograms for imaginary sales of three products along three years. To the left, the absolute values are visualized to enable comparing absolute sales in each year, and to the right, the bars are normalized to the same height to compare how each product contributed to the sales of each year. To perform visual analysis of two or more dimensions, the approaches used depend on whether the variables are all numerical, all categorical, or mixed.

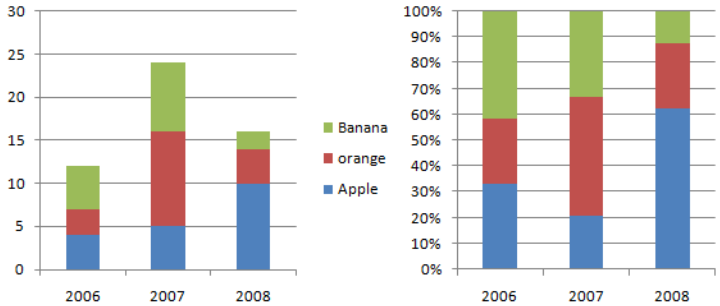

Fig. 15: Stacked histograms. Left: absolute values. Right: percentage values.

2.2.1. Visualization of numerical variables

Two-dimensional scatter plots are a common approach to perform visual analysis of two numerical variables at a time. They are suitable for showing correlation and results of regression analysis, as well as clusters [Fig. 16-a]. A scatter plot can use color to fit an additional variable in the analysis (or use 3D coordinates for this purpose, which however suffer from occlusion issues). Alternatively, linked scatter-plots can be used to analyze pairs of variables for higher-dimensional data.

Parallel coordinates have also been a popular approach for higher-dimensional numerical variables, with lots of extensions and applications. A data item is mapped to a poly-line which links its coordinates in the parallel axes. Fig. 16-b shows an example of three parallel coordinates, with colors used to highlight item similarity.

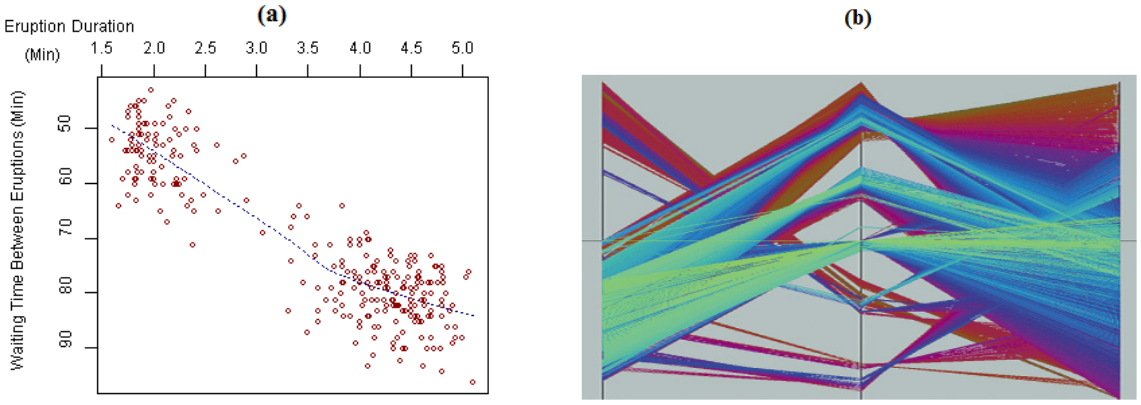

Fig. 16: a) Scatter plot [WP1]. b) Parallel coordinates [K01]

2.2.2. Visualization of categorical variables

Many approaches have been devised for the visual analysis of categorical variables, in particular *mosaic displays* and *parallel sets*.

A mosaic display, introduced by Hartigan and Kleiner [HK81], first divides the space according to the frequency of the values of the first variable, and then recursively divides the space for each of these values according to the frequency of the values of the second variable, and so forth. This bears similarity to the normalized stacked histograms, but mosaic displays do not compromise the distribution of the first variable. In Fig. 17-a, a mosaic display shows how 592 statistics students at the University of Delaware are distributed according to hair and eye colors (reported by Snee in 1974). The view highlights disproportionately high or low frequencies in blue (resp. red) grades. For example, the blue box at bottom left of the display indicates that among people of black hair, the proportion of those having brown eyes is higher than the average. In Fig. 18-a, the boxes were split further according to the gender. The colors here reveal hair-color/eye-color pairs with high disproportionality of genders.

Mosaic displays intuitively communicate the proportion of elements in the boxes. However, they impose an ordering on the variables to perform the recursive sub-division.

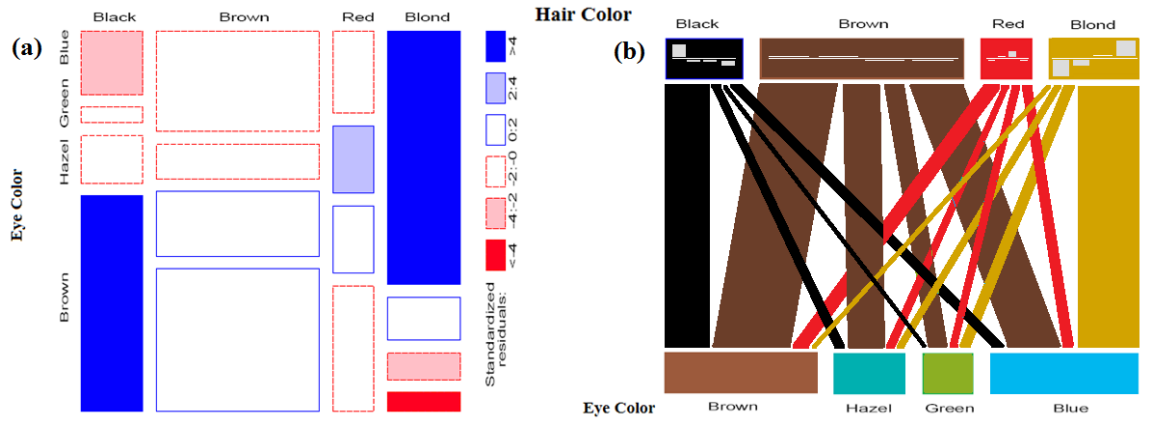

Fig. 17: Analysis of categorical variables using: a) Mosaic displays [F98]. b): parallel sets

Apart, from the first variable, the dataset items for a given category of a secondary variable are distributed into different divisions instead of being grouped in one visual element. This might be a drawback for some applications, especially if the ordering of the variables is not purposeful from the analysis point of view.

Parallel sets, introduced by Hauser et al. [HKB06], treat all variables equally by representing each category of each variable with one visual element whose size is proportional to the category frequency, and placing the categories of the variables along axes parallel to each other. In case of two variables, each pair of categories is connected with a stripe where horizontal width is proportional to the frequency of items which fall in both categories [Fig. 17-b]. Disproportionally high or low frequencies are communicated by means of histograms in the boxes of either variable. For example, in the box of the category “blond” one can see that blue eyes are more frequent than on average, but brown eyes are remarkably uncommon.

Another variable can be incorporated in the analysis using parallel sets, by placing its categories in a new axis parallel to the former ones. The stripes are extended depending on the selected dimension: they start splitting from the categories of the selected dimension into the categories of neighboring dimensions, and then continue splitting into the categories of further dimensions as depicted in Fig. 18-b. In this figure the selected dimension is the hair-color (at the top), and the disproportionality of genders in hair-color/eye-color pairs is shown by means of a histogram in the box of each gender.

Parallel sets have the advantage of mapping each category of each variable to exactly one visual element, and treating the dimensions equally in this sense. The successive splitting of the categories of the selected dimension into stripes preserves the visual continuity and helps understanding the distribution quickly. The area of a stripe is proportional to the frequency of the data elements it represents, however, due to the overlapping of the stripes and the arbitrariness of their directions, it is not as intuitive to comprehend this frequency as is the case in the mosaic display. In this work we present an improvement to parallel sets to address this issue.

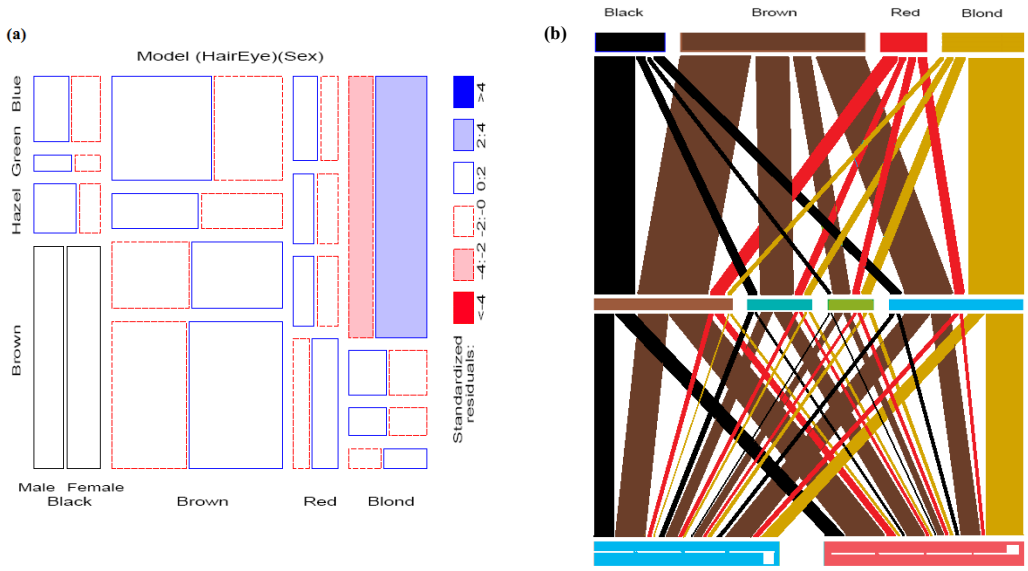

Fig. 18: Analysis of three categorical variables using a) a mosaic display [F98] b) Parallel sets.

2.2.3. Visualization of mixed variables

The different nature of numerical and categorical variables makes visualization of mixed variables challenging. Several approaches have been devised to cope with this problem. Havre et al. [HSPR06] presented a technique that extends parallel coordinates to handle categorical variables [Fig. 19-b]. Ziemkiewicz [Z09] proposed the Forked Parallel Sets, an extension to parallel sets to handle numerical variables [Fig. 19-a]. Often, these approaches try either to quantify categorical variables, or to map numerical variables into categories.

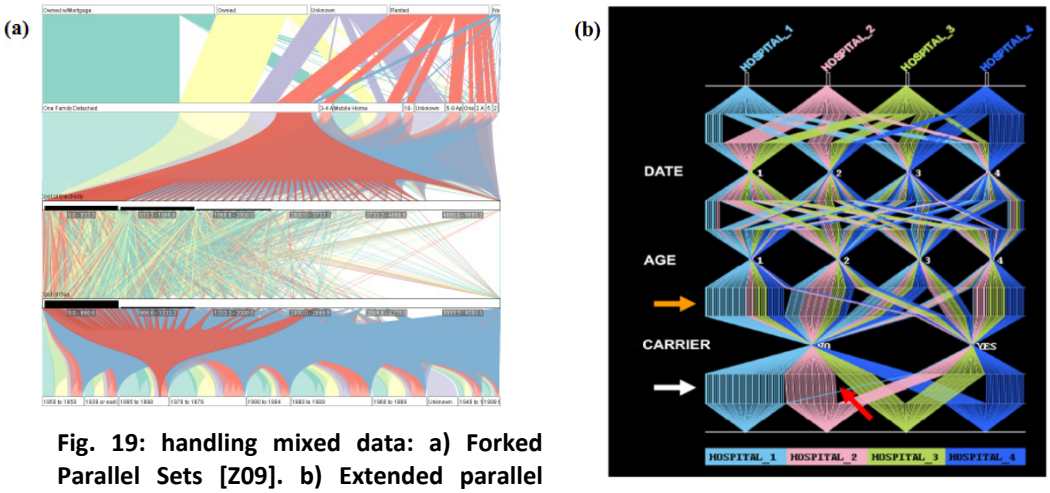

Fig. 19: handling mixed data: a) Forked Parallel Sets [Z09]. b) Extended parallel coordinates [HSPR06].

Quantification of categorical variables maps them into numerical variables. This can be performed either manually or automatically. Defining quantification by hand has the advantage of incorporating domain knowledge and enabling full control over the quantification process, but is not applicable in practice for a large number of categories.

Conversely, automatic quantification is performed typically using Correspondence Analysis [JJJ08] which computes the association between each two categories and determines which categories are close together and hence imposes an ordering and maps the categories as ordinary values. Though automatic methods are efficient, but they hinder both the control over the process and the incorporation of domain knowledge. To overcome this limitation, Johnson et al. [JJJ08] discussed a method to make automatic methods run interactively.

Quantification of categorical data makes it possible to take advantage of the well established methods to visualize numerical data, and to apply mathematical methods to analyze the data (clustering, correlation, etc.). However, to avoid making incorrect conclusions, one should always keep in mind that the ordering is artificial.

Alternatively, a numerical variable can be mapped to a categorical variable by defining range(s) of values for each desired category. The resulting categorical variable can then be used in mosaic displays, parallel sets, or other categorical data visualization techniques. The loss of ordinality can be alleviated by techniques which place the groups of one variable in a visually ordered fashion and enable the user to define this order. The boundary effect resulting from crisp ranges is still a drawback, however.

2.3. Classical and Relational Data Mining

Knowledge discovery in databases (KDD) has been defined as "*the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data*" [FPS96]. Data mining is the main activity in this process, which is concerned with applying computational techniques to actually find the patterns in the data [DL01].

Data mining techniques have been mostly applied to discover knowledge in a single database table of multidimensional data, where each row represents a record or data item, and the columns correspond to the dimensions and specify the attributes of each record. The most popular data mining tasks are [DL01]:

- Regression and classification: the value of one field of the database table is predicated from the values of the other fields of the table. In case this field is numerical, the task is called *regression*, while if it is categorical, the task is called *classification*.
- Clustering: the data items are grouped into classes of similar items.
- Finding association rules: the data table is searched for rules of the form $Attr_{i_1} = val_1 \wedge \ldots \wedge Attr_{i_k} = val_k \implies Attr_j = val_j$.

A variety of approaches and algorithms have been devised to perform these tasks, such as neural networks for regression and classification, the k-means algorithm for clustering, and the Apriori algorithm [M67] for finding association rules in the case of binary attributes.

2.3.1. Relational data mining

By the late 1990s, research in data mining grew beyond mining a single database table, and the above tasks were extended to the case of multiple tables. The generalized techniques are often described by the name *relational data mining*.

In relational data mining, classification or regression of one field in a database table is not only performed based on the values of the other field in this table, but also based on values of related records in other tables. Clustering is also extended to take the relations into account (and hence incorporating graph-theoretic clustering). Association rules are also extended to incorporate relational expressions in the antecedent and consequent of the rules, which generalizes the formula from propositional to first-order logic. In fact, relational association rules mining has been found closely related to Inductive Logic Programming (ILP) which then has been used to generalize the Apriori algorithm to The WARMR algorithm [DD97] to find relational association rules.

The generalization of the above concepts and algorithms to multiple tables is, however, not straightforward and there are many issues that need to be defined (such as distance measures, thresholds, etc.) [DL01]. An alternative approach to perform relational data mining is to employ the non-relational algorithms by transforming the relational data into one table. However, this approach restricts the expressiveness. Additionally, new problems arise in relational data mining such as learning from networks of examples and time-changing relational data [D03].

2.3.2. Data mining vs. interactive visual analysis

Data mining techniques have proved effective in many applications, and with the growing power of modern computers, these techniques have been efficiently applied to analyze large databases and extract hidden knowledge using advanced and complex computations. However, these techniques work usually as black-boxes which limit the role of human reasoning and insight [FGW01]. Even if these algorithms might provide some parameters that the analyst can adjust to “explore” the data, there is still little room for the analyst to participate actively in the mining process, and generally little cognition over the data.

On the other hand, information visualization techniques usually perform simple computations, leaving the task of knowledge extraction to the analyst by means of generating visualizations of high informative value, and allowing intuitive interaction with them. Fayyad et al. [FGW01] discussed the possibilities of combining techniques from both fields in order to exploit the power of each of them, and presented some approaches that have been developed for this purpose.

The authors suggest that information visualization can provide appropriate methods to explore and validate results from data mining techniques. In this sense, the classical data mining algorithms are used efficiently to find patterns, and visualization techniques are used to check the validity of these results and analyze why they happen to exist, and whether the results are intrinsic or due to outliers. The human perception has a high ability to spot abnormalities and special cases that are difficult to be incorporated apriori in blind algorithms. This work tries to combine ideas from the areas of graph drawing and multi-dimensional data analysis to develop a framework for visual analysis

of attributed relational data, and tries to visually answer some typical data mining questions with the ability to validate the results.

Chapter 3

Attribute Modeling and Analysis

"If you wish to converse with me, define your terms" – Voltaire

As mentioned in section 1.2, attributed relational data and their associated graph are defined by the two entity sets E_1 and E_2 and a multiset of relations $G \subseteq E_1 \times E_2$. Associated with each of these sets is a set of attributes and for each attribute a function which computes the value of this attribute for a given element in the corresponding set [Eq. 2.1].

$$R = \left(E_1, E_2, G \subseteq E_1 \times E_2, A_{E_1}, A_{E_2}, A_G\right)$$
$$A_{E_1} = \left\{a_{i,E_1}\right\}_i, a_{i,E_1}: E_1 \longrightarrow S_{i,E_1}$$
$$A_{E_2} = \left\{a_{i,E_2}\right\}_i, a_{i,E_2}: E_2 \longrightarrow S_{i,E_2}$$
$$A_G = \left\{a_{i,G}\right\}_i, a_{i,G}: G \longrightarrow S_{i,G}$$

Eq. 2.1: Definition of Attributed Relational Data

Attribute analysis lies in the heart of this work. Since the above definition of attributes is generic, we have introduced several extensions to the data model by means of adding attributes to the above attribute sets based on existing attributes. The additional attributes enrich the model and make it possible to formulate a broader range of analysis tasks and queries using unified notion and analysis tools.

The above extensions start with a set of *intrinsic attributes* in each of the attribute sets A_{E_1}, A_{E_2} and A_G. The next section presents, without loss of generality, how these intrinsic attributes are defined in this work, based on the specific notion of *events*. The following section discusses how the additional attributes are defined both for the entities and for the relation. Section 3.3 presents the language used to name the resulting set of attributes in a unified manner, and section 3.4 presents how the attribute histograms are created and used for a variety of tasks.

3.1. Intrinsic Attributes

An entity type is defined by means of specifying its name and a set of intrinsic attributes. Each attribute has a name and a type, and it gets its value in a given entity instance when this instance is created (or modified) from internal or external sources.

Likewise, a relation is defined by specifying its name, the two entities it relates, and a set of intrinsic typed attributes. Each attribute has a name, and gets its value when a relation instance is created between two instances of the entities in relation. In this work, entities and relations are defined upon events, as explained in the next section.

3.1.1. From events to entities and relations

As mentioned in section 1.4, the analysis data in this work are built upon typed events in the SENACTIVE InTime™ event-based system. An entity type has a name and a set of typed attributes, and is associated with a base event type from which the unique keys for the instances of this entity type are defined. The values of other attributes can be defined from the base event type or event types correlated with it (the event type correlation can be defined in the modeling utility of SENACTIVE InTime™).

Fig. 20 shows an example of how entities are defined based on events. In this example, two entity types (customer and product), and the relation between them are defined based on the corresponding events.

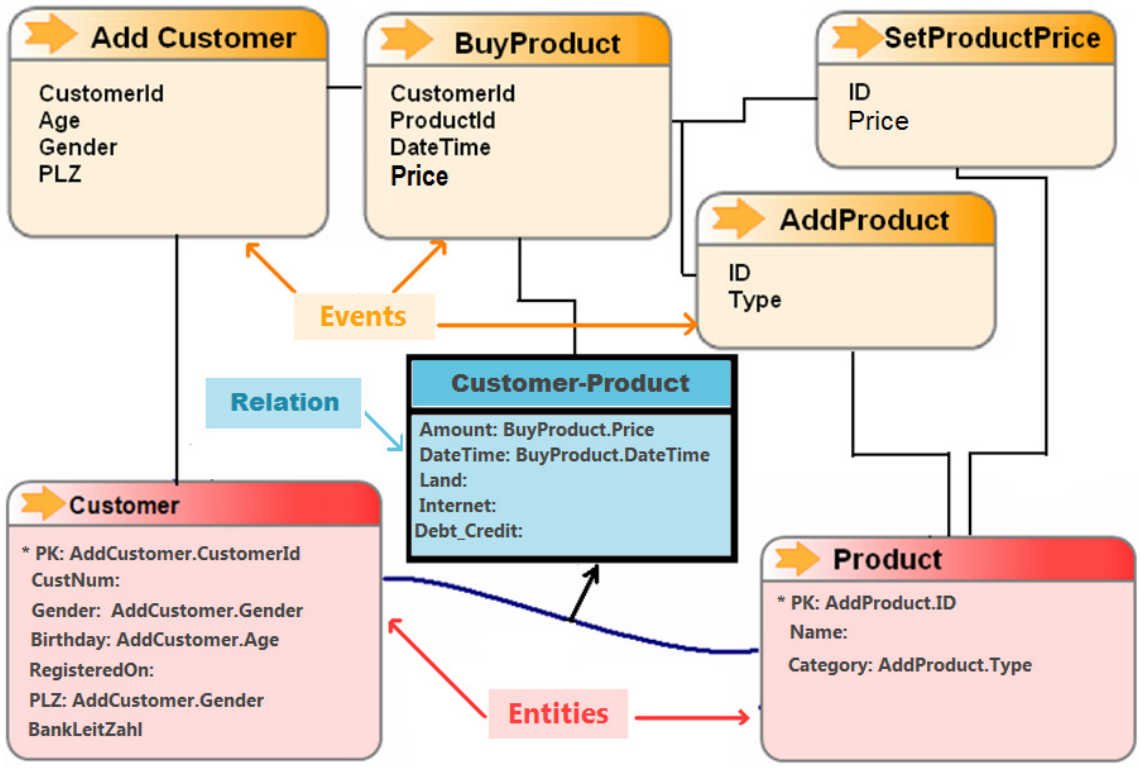

Fig. 20: Entities and relations derived from events

Upon receiving an event of a specific type A, the system creates new instances of entities which base event type is A, and updates entity instances where A is involved in the definition of one or more of their attributes.

Once entity types are defined, relations can be defined between them. A relation can be defined between entities of the same type (e.g., mobile phone users who phone each other, friends in a social network, etc), or between entities of different types (customers are related to the purchased products, support tickets to customer service employees, etc). As is the case with entity types, a relation has also a set of attributes and a base event type. From this event type, the key of both entity instances that need to be related and the values for the relation attributes can be defined. When an event of this type is received, the system will add a new relation instance between the corresponding entity instances. Multiple relation instances may exist between two entity instances. In Senactive InTime™, events are received and processed in near real time, and hence, the attributed relational data are updated in real time, which implicitly enables performing analysis based on the most recent data.

3.1.2. Attribute types

Each attribute in entities and relations is given a type when it is defined. These types fall in two categories:

- Ordinal types
    - Numerical types: these include types for integer and floating point numbers.
    - Date/Time type: in this work, the values from this type are discretized according to the year, month, day_of_month, or day_of_week component. The discretization result is represented by an integer number which is treated equivalently to numerical types. The year component is used by default in case no discretization is explicitly defined.
- Nominal Types:
    - String type: typically used as identifier or to represent textual information.
    - Categorical type: a set of finite, un-ordered values that represent the possible values for the variable. This type is implemented as a string type.

As discussed in section 2.2.3, the analysis of mixed data which has variables both of ordinal and nominal types is challenging. This work follows the approach of mapping variables of ordinal types to nominal types. The mapping is performed by defining a nominal value for a range of ordinal values as discussed in section 3.2.3.

3.2. Extended Attributes

In this work, four types of extensions to the entity attributes are introduced. These extensions are network-based attributes, layout-based attributes, grouping attributes, and aggregated attributes. These extensions are specific to a given relation $R = (E_1, E_2, G)$.

The sets of all attributes for both entities are then defined as follows:

$$A_{E_1} = A_{E_1\_Intrinsic} \cup A_{E_1\_Network} \cup A_{E_1\_Layout} \cup A_{E_1\_Grouping} \cup A_{E_1\_Aggregated}$$

$$A_{E_2} = A_{E_2\_Intrinsic} \cup A_{E_2\_Network} \cup A_{E_2\_Layout} \cup A_{E_2\_Grouping} \cup A_{E_2\_Aggregated}$$

The next sections present each of the above extensions in details.

3.2.1. Network-based attributes

Each entity in the relational data corresponds to a node in the corresponding attributed graph. There are various graph-theoretic metrics associated with each node in the graph such as the node degree and other centrality measures (betweenness, closeness, etc). These metrics are computed for a given node based on the network structure.

This work implements "node degree" as an example for a network-based attribute. In case $E_1 \neq E_2$ this attribute is defined as follows:

$$nodedegree_{R,E_1}: E_1 \longrightarrow \mathbb{Z}: e_1 \longrightarrow |\{e_2: (e_1, e_2) \in G\}|$$

$$nodedegree_{R,E_2}: E_2 \longrightarrow \mathbb{Z}: e_2 \longrightarrow |\{e_1: (e_1, e_2) \in G\}|$$

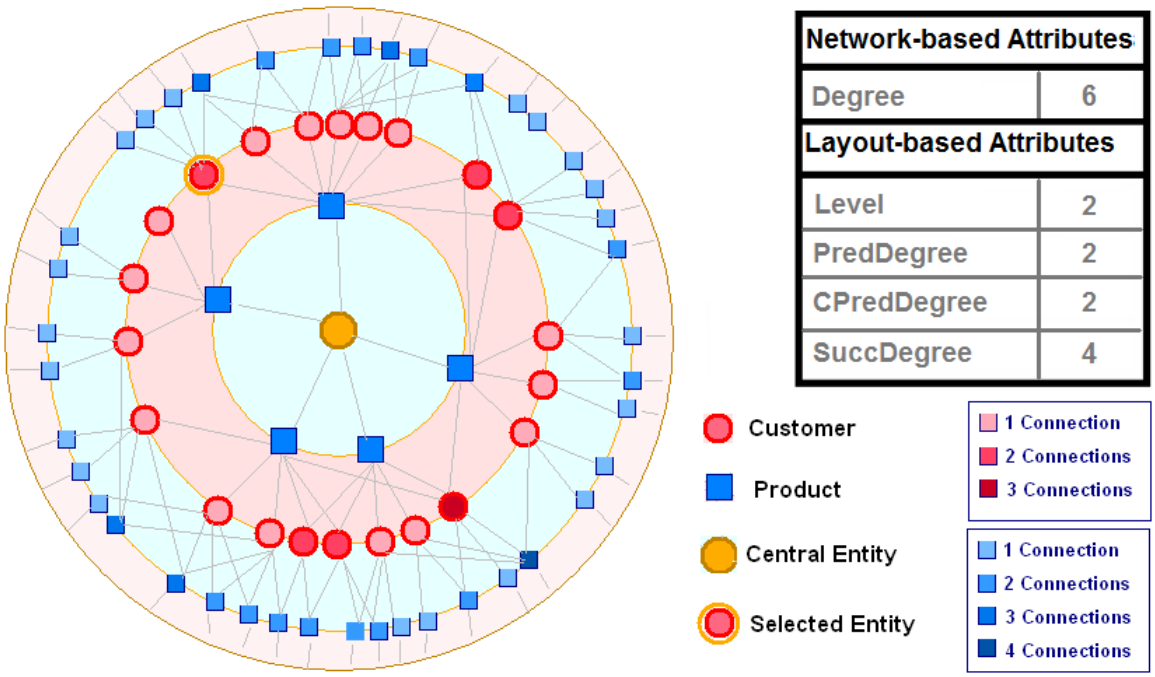

Fig. 21: Network and Layout-based Attributes.

3.2.2. Layout-based attributes

Each graph drawing layout performs its own node placement and associates specific metrics with the nodes which might be useful for the analysis.

For example, in the radial layout [Fig. 21], the network distance* between the central node and a given node decides on which circle this node will be placed. We call this distance the level of the node in the current view.

Formally, if $e_c \in E_i$: i $\in \{1,2\}$ is the central node, the level of a given node is defined as follows:

$$Level_{R,e_c}: E_1 \cup E_2 \longrightarrow \mathbb{Z}: e \longrightarrow \begin{cases} \infty & if\ no\ path\ exists\ bewteen\ e_c\ and\ e \\ \min\left\{\text{k: } \exists(e_1,..,e_k): \begin{matrix} e_1 = e_c \wedge\ e = e_k \wedge \forall i \in \{1..k-1\}: \\ (e_i \in E_1 \Rightarrow (e_i, e_{i+1}) \in G) \wedge (e_i \in E_2 \Rightarrow (e_{i+1}, e_i) \in G) \end{matrix}\right\} & otherwise \end{cases}$$

In this work, four other radial-layout-based attributes are defined:

- PredDegree: for a given node, this attribute computes the number of nodes in the previous level (ring) to which this node is connected.

  Formally it is defined as follows:

  $$PredDegree_{R,E_1,e_c}: E_1 \longrightarrow \mathbb{Z}: e \longrightarrow \left|\{e_2: (e,e_2) \in G\ \wedge Level_{R,e_c}(e_2) < Level_{R,e_c}(e)\}\right|$$
  $$PredDegree_{R,E_2,e_c}: E_2 \longrightarrow \mathbb{Z}: e \longrightarrow \left|\{e_1: (e_1,e) \in G\ \wedge Level_{R,e_c}(e_1) < Level_{R,e_c}(e)\}\right|$$

- SuccDegree: for a given node, this attribute computes the number of nodes this in the next level (ring) to which this node is connected.

  Formally it is defined as follows:

  $$SuccDegree_{R,E_1,e_c}: E_1 \longrightarrow \mathbb{Z}: e \longrightarrow \left|\{e_2: (e,e_2) \in G\ \wedge Level_{R,e_c}(e_2) > Level_{R,e_c}(e)\}\right|$$
  $$SuccDegree_{R,E_2,e_c}: E_2 \longrightarrow \mathbb{Z}: e \longrightarrow \left|\{e_1: (e_1,e) \in G\ \wedge Level_{R,e_c}(e_1) > Level_{R,e_c}(e_2)\}\right|$$

- CPredDegree: for a given node, this attribute computes the number of shortest paths that connect it to the central node. For nodes in the first two levels, it is equal to the PredDegree attribute, but for a node in farther-away levels, it is computed recursively as the sum of the CPredDegree values of nodes related to it in the previous level.

  Formally it is defined as follows:

  $$CPredDegree_{R,E_1,e_c}: E_1 \longrightarrow \mathbb{Z}: e \longrightarrow \begin{cases} Level_{R,e_c}(e) & if\ Level_{R,e_c}(e) \le 1 \\ \sum\{CPredDegree_{R,E_2,e_c}(e_2): (e,e_2) \in G\ \wedge Level_{R,e_c}(e) > Level_{R,e_c}(e_2)\} & otherwise \end{cases}$$
  $$CPredDegree_{R,E_2,e_c}: E_2 \longrightarrow \mathbb{Z}: e \longrightarrow \begin{cases} Level_{R,e_c}(e) & if\ Level_{R,e_c}(e) \le 1 \\ \sum\{CPredDegree_{R,E_1,e_c}(e_1): (e_1,e) \in G\ \wedge Level_{R,e_c}(e) > Level_{R,e_c}(e_1)\} & otherwise \end{cases}$$

- SelectionWeight: this attribute is defined based on node selection in the view and will be explained in details in section 4.2.2.

Fig. 21 shows an example of network and layout-based attributes for a selected node. The values of layout-based attributes depend on the view parameters.

* The network distance between two nodes is the number of edges in the shortest unweighted path between them.

3.2.3. Hierarchical grouping attributes

For each of the attributes in $A_{E_i} \setminus A_{E_i\_Aggregated}$, a grouping can be defined which maps each value of this attribute to one of a set of groups. Formally, we define the mapping of the values of the attribute a_{j,E_i} to their groups $Gr_{a_{j,E_i}}$ as follows:

$$gr_{a_{j,E_i}}: S_{j,E_i} \rightarrow Gr_{a_{j,E_i}}: x \rightarrow gr_{a_{j,E_i}}(x)$$

The set of groups, $Gr_{a_{j,E_i}}$, and the corresponding mapping, $gr_{a_{j,E_i}}$, can be defined by the user for each attribute in $A_{E_i} \setminus A_{E_i\_Aggregated}$. In fact this grouping can be formulated as an additional attribute $g_{a_{j,E_i}}$ for the entity type E_i as follows:

$$g_{a_{j,E_i}}: E_i \rightarrow Gr_{a_{j,E_i}}: e \rightarrow g_{a_{j,E_i}}(e) = gr_{a_{j,E_i}}(a_{j,E_i}(e))$$

This attribute is of categorical type and its categories are the defined groups. Moreover, since $g_{a_{j,E_i}} \in A_{E_i} \setminus A_{E_i\_Aggregated}$, a grouping for this attribute can be defined, which means that the above groups can be further grouped, which enables defining a hierarchical grouping for each intrinsic, layout-based or network-based attribute.

Attached with each attribute of an entity type, a default grouping exists, which consists of one group which name is the entity type name and which contains all the possible values. After defining the groups for an attribute, the mapping function can be defined by assigning a group to each category for categorical attributes, and to each value range for ordinal attributes.

Fig. 22 shows an example, in which for the intrinsic attribute "PLZ" (postal code) in the entity type "Customer", a grouping was defined by means of the grouping attribute $g_{Plz,Customer}$, and a further grouping was defined by means of the attribute $g_{g_{Plz,Customer}}$. The default grouping attribute defines the last level of grouping in the hierarchy.

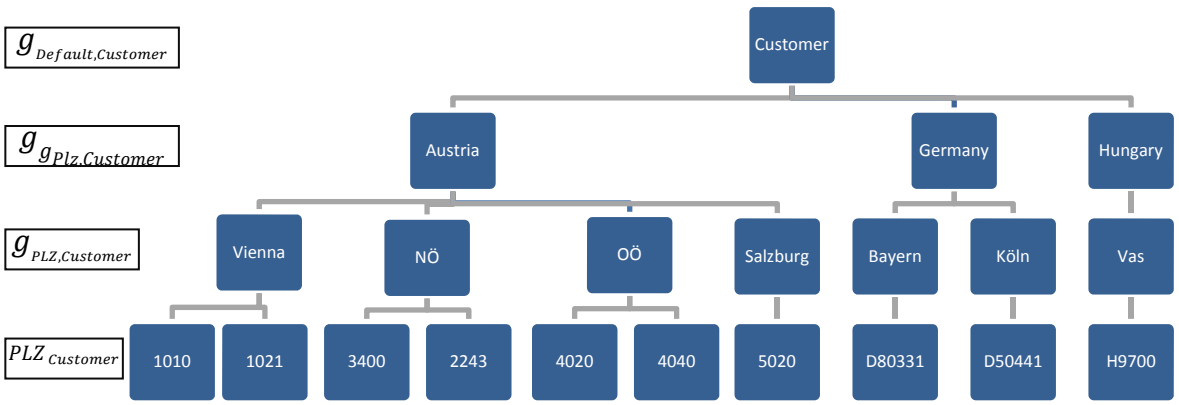

Fig. 22: Example hierarchy of grouping attributes.

3.2.4. Aggregated attributes

An aggregated attribute in the entity type E_1 in the relation R is an additional attribute defined based on a non-aggregated attribute (called atomic attribute) in the other entity type E_2. For a given entity instance $e \in E_1$, the value for this aggregated attribute is computed by aggregating the values of the atomic attribute in the entity instances related to e.

Formally, an aggregated attribute in E_1 based on the atomic attribute a_{i,E_2} in E_2 is defined as follows:

$$a_{a_{i,E_2},E_1}: E_1 \rightarrow AGG\_S_{i,E_2}: e \rightarrow Agg\{a_{i,E_2}(e`): (e, e`) \in G\}$$

Agg is one of a set of aggregation functions available for the type of the atomic attribute. For example, in the "product" entity type defined in Fig. 20, we can define the attribute "Customer.Birthday.Avg" which is computed for each product by aggregating the ages of the customers who bought it using the Average function to compute the average age of these customers.

There are several methods to perform aggregation of a list of values L, and they depend on the type of the atomic attribute whose values are to be aggregated:

- Numerical / DateTime attribute
  - The minimum, maximum, average, or standard deviation of the values in L .
- Categorical attribute
  - The absolute mode value: the value which has the highest frequency in L. It is formally defined as follow ($\text{freq}_L(x)$ denotes the frequency of x in L):

$$Mode(L) = \arg\max_x \{\text{freq}_L(x) : x \in L\}$$

  - The normalized mode value: this attribute uses normalized frequencies, $\text{Nfreq}_L(x)$, to compute the mode value. The frequency of an element x in L is normalized by dividing it with the frequency of x in E_2 (denoted by $\text{freq}_{E_2}(x)$):

$$NMode(L) = \arg\max_x \left\{ \text{Nfreq}_L(x) = \frac{\text{freq}_L(x)}{\text{freq}_{E_2}(x)} : x \in L \right\}$$

  - The distinctivity factor of the (normalized) mode category: this attribute gives an idea on how much the mode value is dominant in the list of values. It is computed as $Distinctivity(L) = Distinctivity_L(Mode(L))$ for the absolute mode value, and as $NDistinctivity(L) = NDistinctivity_L(NMode(L))$ for the normalized mode value, where:

$$Distinctivity_L(x) = \frac{\text{freq}_L(x) - \frac{|L|}{|S_{i,E_2}|}}{|L| - \frac{|L|}{|S_{i,E_2}|}}, \qquad NDistinctivity_L(x) = \frac{N\text{freq}_L(x) - \frac{ws}{|S_{i,E_2}|}}{ws - \frac{ws}{|S_{i,E_2}|}}$$

    In these equations, $|S_{i,E_2}|$ denotes the number of possible categories for the atomic attribute, and the term ws is computed as $ws = \sum_{x \in L'} \text{Nfreq}_L(x)$ (where $L' = \{x : x \in L\}$ is the set of elements of the multiset L).

3.2.5. Extensions to relation attributes

Besides the intrinsic attributes of a relation, this work extends the set of the relation attributes to include the attributes of the both entities in relation. Formally, this set is defined as follows:

$$A_G = A_{G\_E_1} \cup A_{G\_Intrinsic} \cup A_{G\_E_2}$$

$$A_{G\_E_1} = \{a_{i,G\_E_1}\}, a_{i,G\_E_1}: G \longrightarrow S_{i,E_1}: g = (e_1, e_2) \longrightarrow a_{i,E_1}(e_1)$$

$$A_{G\_E_2} = \{a_{i,G\_E_2}\}, a_{i,G\_E_2}: G \longrightarrow S_{i,E_2}: g = (e_1, e_2) \longrightarrow a_{i,E_2}(e_2)$$

$$A_{G\_Intrinsic} = \{a_{i,G\_R}\}, a_{i,G\_R}: G \longrightarrow S_{i,R}: g = (e_1, e_2) \longrightarrow a_{i,G\_R}(g)$$

Fig. 23 shows an example relation and the subsets of attributes that form the relation attributes.

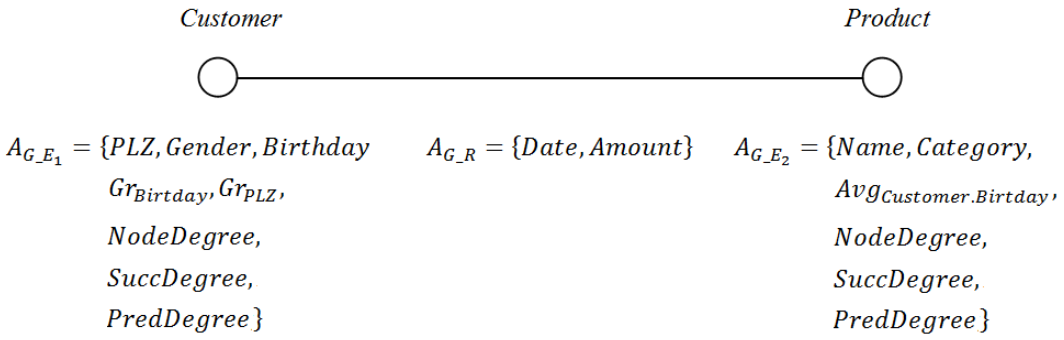

Fig. 23: Extended relation attributes

Incorporating the attributes of the related entities in the relation attributes enables analysis of their distributions in the set of relations G, as opposed to distributions in the entitiy sets E_1 and E_2. Also, it enables performing conditional distributions as well association analysis of attributes from the two different entities. This analysis is performed using the same mechanism as the analysis of intrinsic attributes, and no additional tools is needed since all attributes can be treated in the same way.

The distribution of an extended relation Attribute a_{i,G_E_1} in the set of relations G can be compared with the distribution of entity attribute it is based on a_{i,E_1} in the set of entities E_1. This helps finding how the relations are distributed among the elements based on the values of the attribute a_{i,E_1} in them.

3.3. Attribute Naming

Various attributes for entities and relations have been introduced and mathematically defined in this chapter. To enable a unified mechanism to work with attributes, a naming system for the attributes is needed which assigns for each attribute a unique name. This name is chosen to enable easy extraction of the semantics of this attribute. In what follows, the naming rules are listed for the different types of attributes. A combination of these rules can be used when applicable.

- **Intrinsic attributes**

  The intrinsic attribute “Attr” in entity type “EntityName” is named:

  “EntityName. Attr”

  The intrinsic attribute “Attr” in the relation “RelationName” is named:

  “RelationName.Attr”

- **Network-/layout-based entity attributes**

  The network/layout-based attribute “Attr” in the entity type “EntityName” is named:

  “EntityName.Attr”

- **Hierarchical grouping attributes**

  The grouping attribute based on attribute “Attr” in the entity type “EntityName” is named:

  “EntityName.Attr.G”

  Using the same rule, the higher levels in the hierarchy can be named (e.g. *“EntityName.Attr.G.G”, “EntityName.Attr.G.G.G”*)

- **Aggregated attributes**

  The aggregated attribute in entity type “EntityName” based on the atomic attribute “Attr” in the related entity type “Entity2Name” using the aggregation function “Agg” is named:

  “EntityName.Entity2Name.Attr.Agg”

  The aggregation function “Agg” can be one of the following functions

  - For numerical attributes: *Max, Min, Avg, StdDev*
  - For categorical attributes: *Mode, Distinctivity, NMode, NDistinctivity*

- **Additional relation attributes**

  The additional attribute in the relation “RelationName” based on attribute “Attr” in the entity type “EntityName” of the relation is named:

  “RelationName.EntityName.Attr”

- **Date attributes**

  For the date time attribute “Attr” in the entity type “EntityName”, the components extracted from it are given different names and treated as different attributes:

  “EntityName.Attr.Year”

  “EntityName.Attr.Month”

  “EntityName.Attr.DayOfMonth”

  “EntityName.Attr.DayOfWeek”

Examples

Here are some attributes named using the above rules applied to the customer-product relation depicted Fig. 22:

- *Customer.Age.G*: denotes the grouping of the customer age (18-25, 26-32, 33-39, 40-50, 51-60, 60-70, 70+)
- *Product.Customer.NodeDegree.Avg*: denotes the average number of products purchased by customers who purchased a given product.
- *Relation.DateTime.DayOfWeek*: denotes the day of week on which the given relation instance occurred.
- *Relation.Customer.PLZ*: denotes the postal address of the customer who was involved in a given relation instance.
- *Customer.PLZ.G.G*: denotes the second level grouping of the postal address of the customer. This corresponds to the country of the customer in the example depicted in Fig. 22.

3.4. Attribute Distribution Analysis

One of the most fundamental tools in the visual analysis of attributed graphs is the visual analysis of attribute distributions by means of histograms. In this work, the attribute analysis module is used extensively for various tasks, and serves as the central module that enables interaction with other visualization modules. These tasks include:

- Showing a global or conditional distribution of a given attribute: this helps understanding how the attribute values are distributed in the whole or a subset of one of the sets E_1, E_2 *or* G.
- Mapping attributes to visual properties: each attribute can be mapped to color and opacity which can be used for coloring the graphical elements (nodes, edges, groups, etc.) associated with it. Additionally, the node size can be mapped from the specific layout attribute “SelectionWeight”, and the color saturation can be mapped from the distinctivity factor of a categorical aggregated attribute “EntityName.Entity2Name.Attr.Distinctivity”.
- Defining a grouping for a given attribute: as mentioned in section 3.2.3, this grouping maps the values of the attribute to the corresponding groups defined by the user for this attribute.
- Highlighting / filtering nodes and defining visual queries: this will be discussed in section 4.4.2.

3.4.1. Histograms and transfer functions

As mentioned in section 2.2, a histogram of an attribute is the visualization of the frequencies of data items in each category in case a categorical attribute, and in each interval in case of a numerical attribute. The histogram is created and visualized in a similar manner for both cases, with some aspects specific for each case.

3.4.1.1. Ordinal attributes

To create a histogram of an ordered attribute, the smallest range, which contains all the values of this attribute in the graph, is computed, and data binning is performed to map the data to intervals (bins). The bins are then visualized as bars [Fig. 24]. The bin width determines the resolution of the histogram, with large bin width used to get fewer details, and vice versa. In case of discrete attribute types, the smallest bin width is 1.

In Fig. 24, the list of attributes of the entity type “Customer” is shown, and the histogram of the attribute “Customer.Birthday” is created and viewed. The figure uses discretization of the attribute values according to the year component. Items that have no values assigned for the attribute are counted in a separate bar named “Unknown”.

The resolution of the view can also be adjusted by specifying the number of bins. Also, the boundaries can be adjusted by the user. Zooming and panning enable closer examination of a specific part in the histogram.

Multiple value ranges can be defined with the mouse in the above view. Defining ranges is vital to create conditional distributions, define attribute grouping, and brush data in other visualization modules. For each attribute, a set of groups can be defined. The mapping of attribute values to their groups is done by selecting a range of values and choosing one of the groups so that all values in this range are assigned to this group.

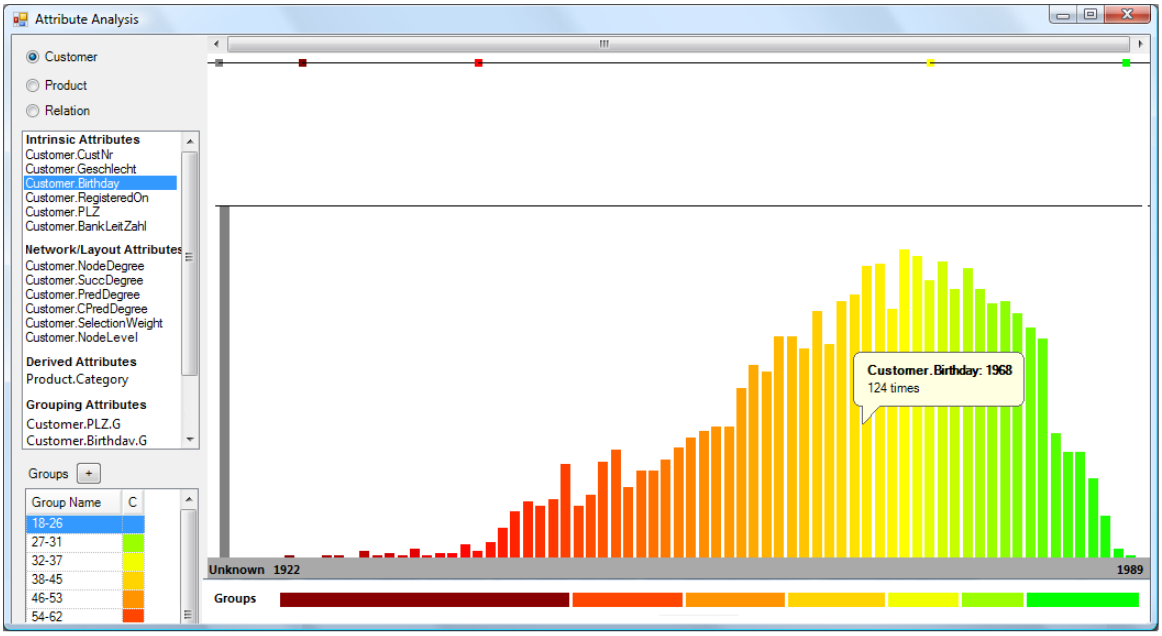

Fig. 24: Histogram and transfer function of an ordinal attribute

The bars of the histogram are assigned color by applying a transfer function to their middle values. The transfer function defines the mapping of the attribute values to color and opacity values. It is specified by means of control points. Each control point has an x-position which determines the value of the point in the histogram, a y-position which determines the opacity of the node, and an RGB color value [Fig. 24]. Two control points are fixed at the two boundaries of the histogram, and can only be moved vertically. Further points can be added at any location in the visible range.

For a given value inside the boundaries, the color and opacity are computed by interpolating the colors (resp. the opacities) of the two control points that surround it. For values outside the range, the color and opacity of the closest endpoint to the value is assigned. The colors of the bars of the histogram are updated continuously when the transfer function points are moved, in order to aid the user to define the desired mapping.

3.4.1.2. Categorical attributes

The histogram for a categorical attribute is created by finding all the possible values (categories) of this attribute in the graph along with their frequencies. The values are visualized as horizontal bars whose lengths correspond to the frequencies, and are also placed textually besides the name of the category [Fig. 25]. The categories can be sorted either alphabetically or according to their frequencies.

To define a transfer function for a categorical attribute, a color and opacity can be assigned to each category on its own.

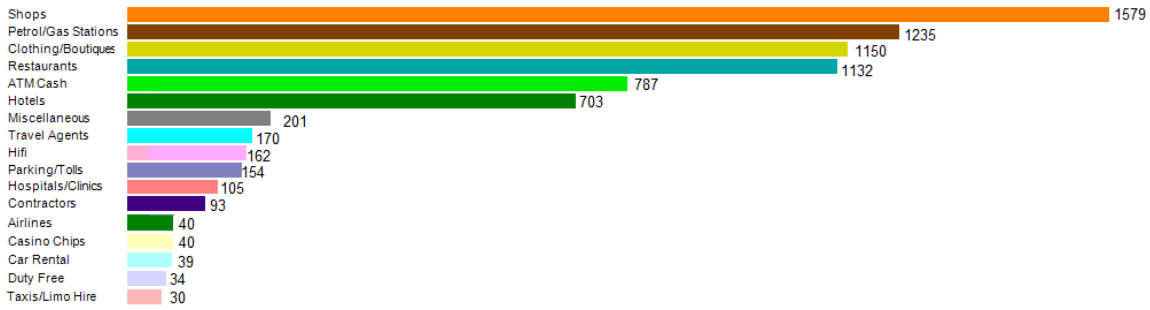

Fig. 25: Histogram and transfer function of a categorical attribute.

The same as in the case of an ordinal attribute, multiple values can be selected in the view. Also, the selected values are used to define groupings, create conditional distributions and brush data in other visualization modules.

3.4.2. Conditional histograms - defining visual queries

A conditional histogram of an attribute is a histogram created from a subset of the data items defined by a set of membership conditions on the graph attributes. The subset of entity/relation instances is defined as follow:

$$I \supset S(I) = \{x \in I : \forall j, attr_{I,j}(x) \in range_{1_j} \cup \ldots \cup range_{k_j}\}, \qquad I \in \{E_1, E_2, G\}$$

The conditions can be visually defined by first creating a global histogram of the first attribute in the conditions and selecting the desired ranges / categories in it, and then creating a conditional histogram for the second attribute (applied to the subset of the data items which satisfy the first condition) and repeating the process until all the conditions are defined, where at the last condition, the conditional histogram of the desired attribute is created. Fig. 26 shows how the conditional distribution of the attribute "Customer.NodeDegree" is created with the conditions $1979 \leq Customer.BirthDay \leq 1989 \wedge Customer.Gender \in \{F\}$

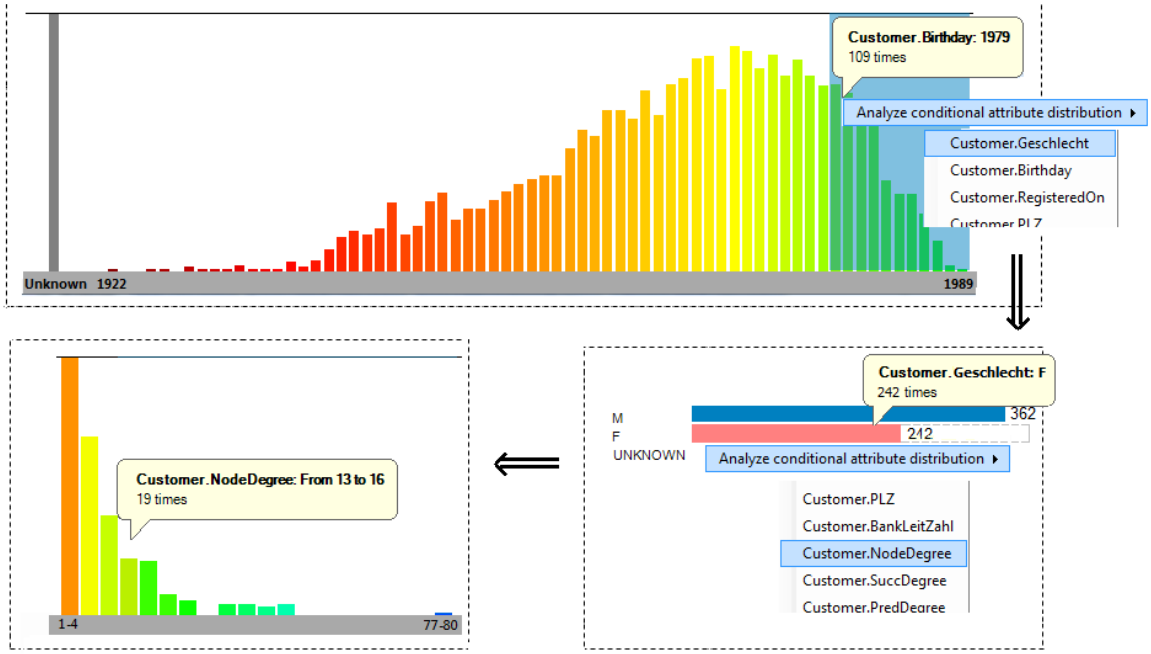

Fig. 26: Creating a conditional histogram.

The ranges used in creating a conditional histogram can be interactively changed and the resulting histogram is updated accordingly. This helps doing manual exploration of associations between the attributes. In the current implementation, the update is propagated only to the next conditional histogram.

Besides viewing the absolute frequencies in the conditional histogram, the user can choose to view these frequencies as a sub-histogram in the global histogram of the same attribute. Additionally, a further mode for the histogram enables comparing the conditional and the global distributions by showing the ratio growth for each bin/category *v* computed as follows:

$$ratiogrowth(v) = \frac{freq_{S(I)}(v)}{freq_I(v)} \cdot \frac{|I|}{|S(I)|}$$

The ratio growth is mapped to the bar height, and its support, defined as $freq_I(v)$, is mapped to the bar opacity [Fig. 27-c].

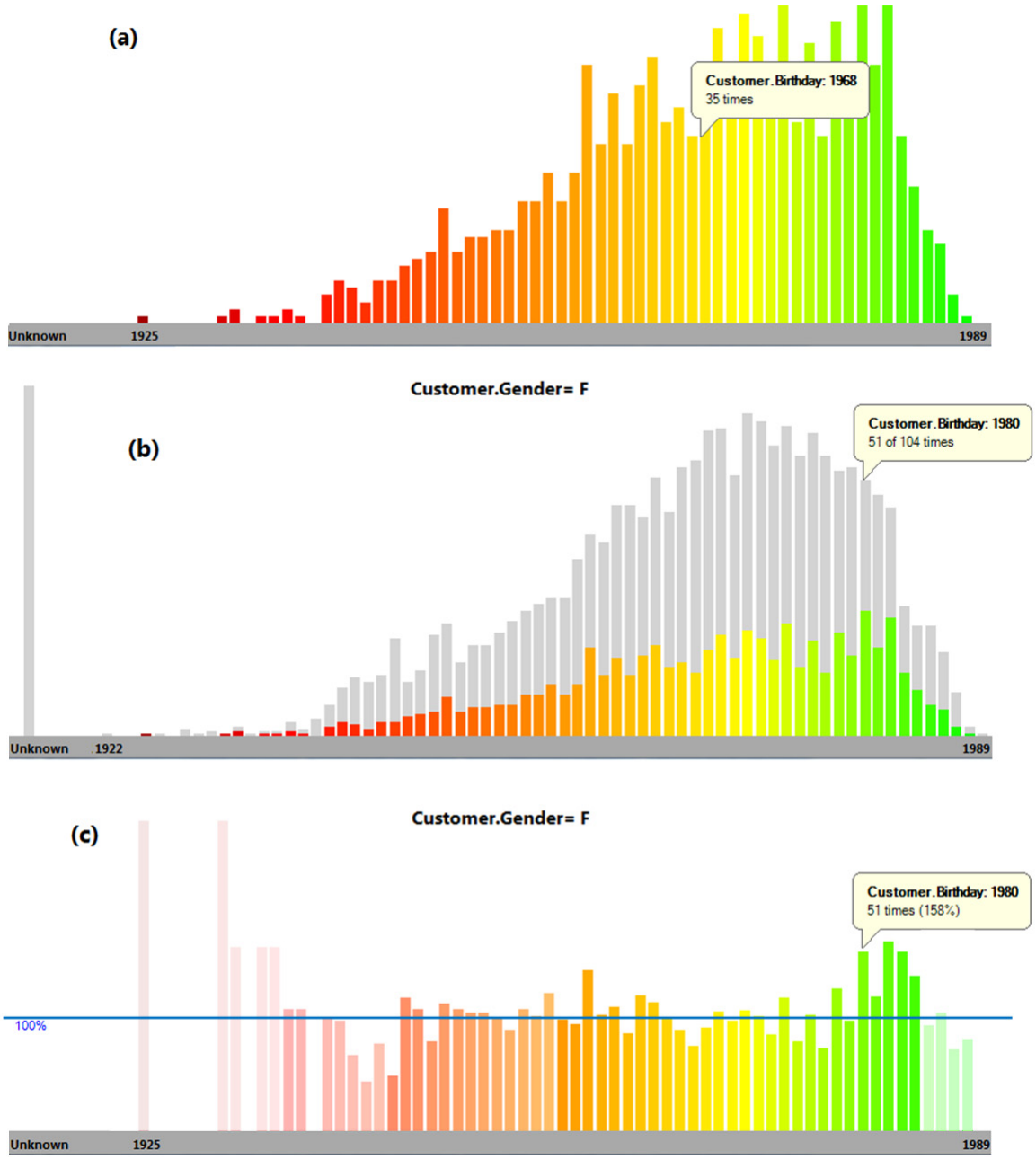

Fig. 27: Modes of conditional histograms. (a) Absolute values, (b) sub histogram, (c) ratio growth.

In this mode one can easily find which bins/categories have increased their relative frequencies in the dataset, and the opacity helps distinguishing if the growth is statistically backed with sufficient evidence. Fig. 27 shows the conditional distribution of customer birthday in the set of female customers using the above defined three

different modes. In Fig. 27-c we can notice that birthdays between 1980 and 1984 are overrepresented in the case of female customers.

In case of an extended relation attribute, i.e., attributes defined from one of the related entities), it is interesting to compare the histogram of this attribute in the relation instances and its histogram in the entity instances. This is achieved by introducing an additional mode in the histogram of such attribute which computes for each bin/category *v*, the average number of relations per item in it computed as:

$$RelationPerItem(v) = \frac{Freq_R(v)}{Freq_E(v)}$$

This number is mapped to the bar height, and its support, defined as $Freq_E(v)$, is mapped to the bar opacity [Fig. 28-b]. In Fig. 28, one can recognize that younger people tend to make a slightly higher number of purchases compared to older ones.

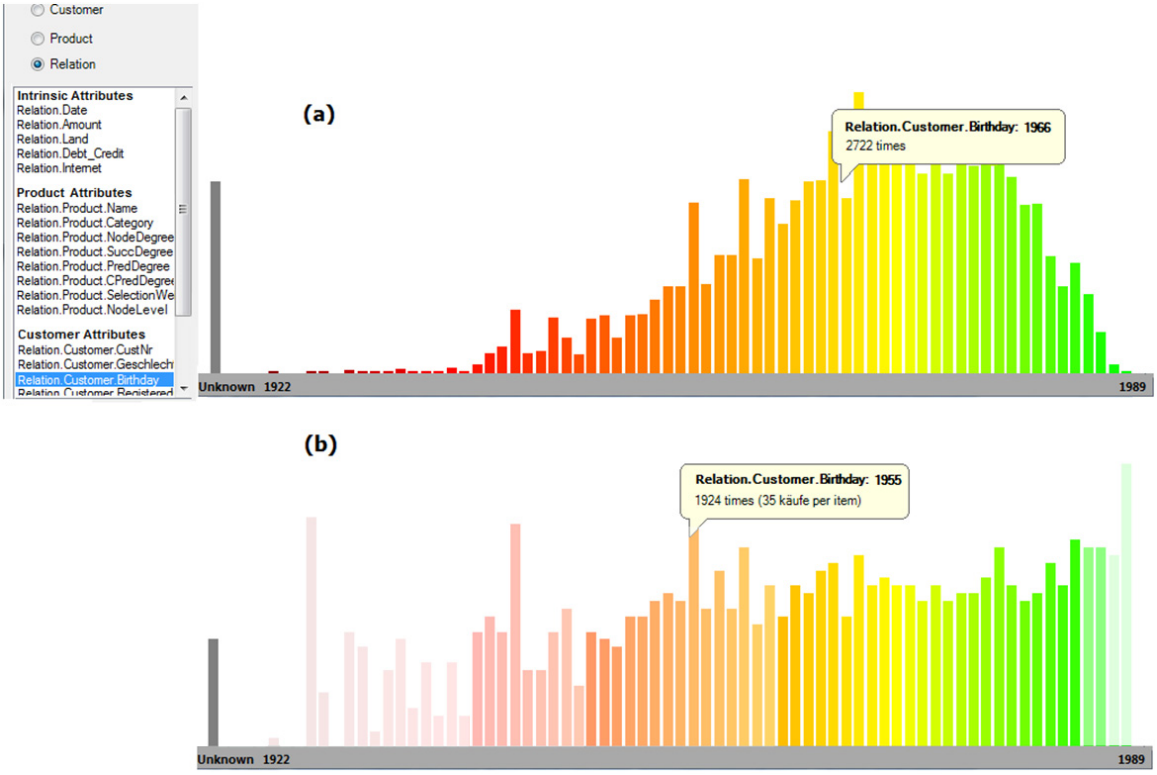

Fig. 28: Entity vs. relation attribute distribution

It is worth mentioning that since the relation incorporates attributes from both entities it relates, a conditional histogram of a relation attribute from one entity can be created with a condition defined on the relation attribute from the other entity. Fig. 29 shows an example, where the conditional histogram of the attribute "Relation.Product.Category" is analyzed in the relation instances that satisfy $1979 \leq Relation.Customer.BirthDay \leq 1989$. The ratio growth mode for the conditional histogram is chosen to show which products are overrepresented in the purchases of young people.

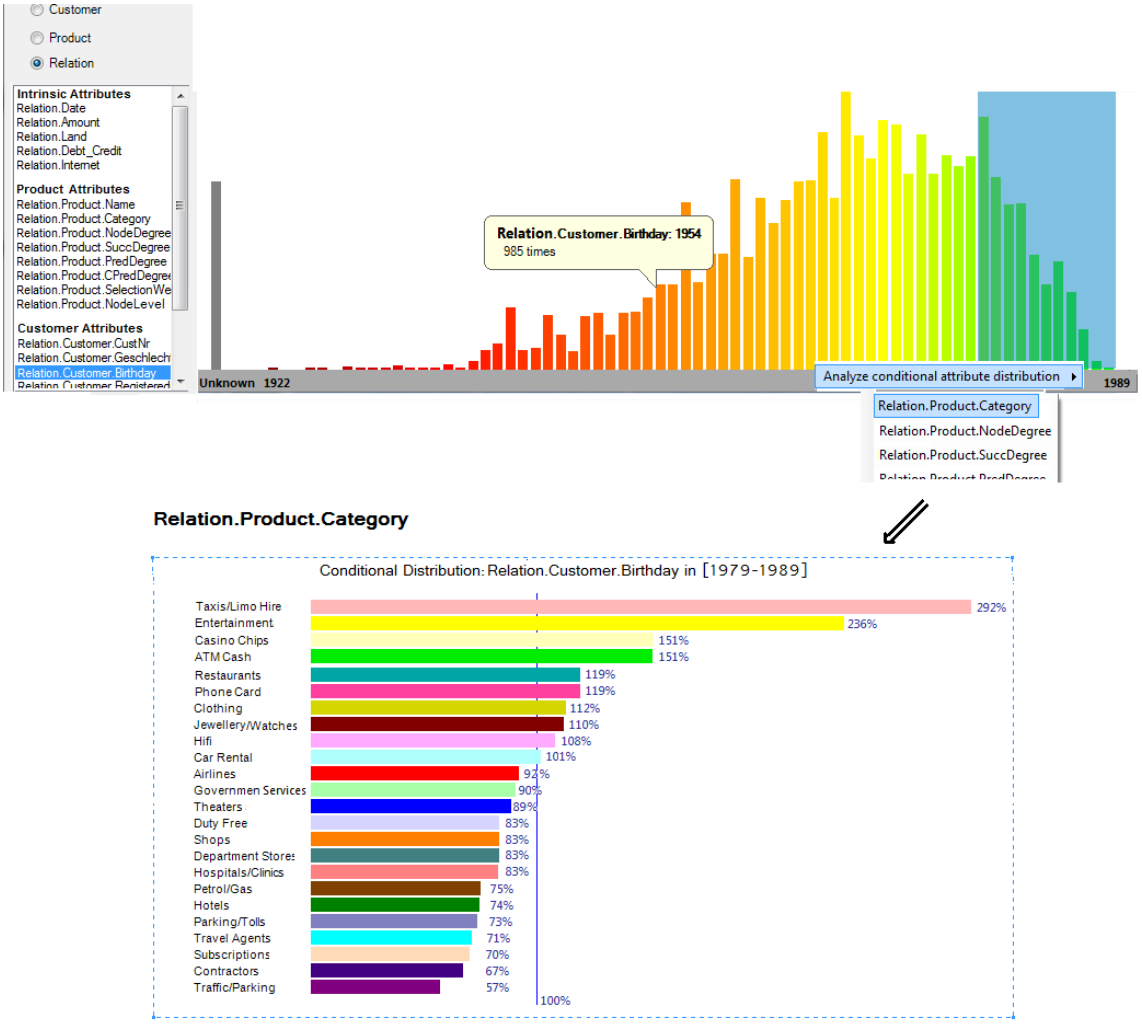

Fig. 29: Conditional histogram of relation attributes

Chapter 4

Graph Exploration

"Part of the fantasy technique is to visualize something as perfect. Then with the experiments you work back from the fantasy to reality, hacking away at the components", Edwin H. Land

As stated in section 1.3, one task of visual analysis in this work is to enable exploration in the attributed graph. This includes providing a detailed view of a selected part of the graph where all the nodes in this part are well visible in order to examine their attributes and enable interaction with them. Hence, this view gives emphasis on drawing visible nodes as large as possible.

Same as in the animated exploration method of Yee et al. [YFDH01] (section 2.1.1), the graph exploration in this work is based on the radial layout which places one node in the center of the view, and the nodes of its neighboring subgraph in successive rings surrounding it.

After this subgraph is visualized, each of the visible nodes can be chosen to be the next central node and hence perform navigation in the view. As in the graph exploration method of Yee et al., an animated navigation is implemented which animates the nodes from their former positions to their newer positions by interpolating between the old and new radial coordinates (with the origin in the view center).

The same as proposed by Huang [H01], the visualization of the subgraph is performed by first performing the geometric mapping (which computes the node positions according to the chosen layout and its parameters). Then the graphical mapping is performed, which assigns the visual properties (foreground color, background color and opacity) for the nodes based on their entity attributes using the transfer functions defined in the previous chapter. In the current implementation, the node size can be mapped only from the "SelectionWeight" attribute.

The next three sections will discuss the geometric mapping and present two solutions implemented in this work: drawing the subgraph with links and without links. Section 4.4 discusses how the nodes can be brushed in the implemented views, and how these views are used to perform visual queries.

4.1. Usage of the Radial Layout

The main advantage of using the radial layout [Fig. 30] is that it enables a node-centric and elaborate analysis of the neighboring subgraph of the selected node, and intuitively communicates the network distance between the subgraph nodes.

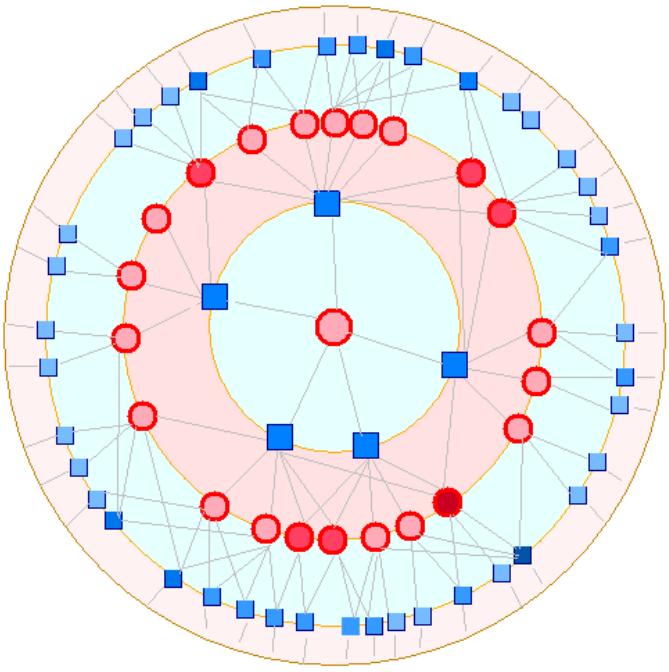

Fig. 30: Radial layout for graph exploration

However, the main challenge of using this layout to draw a graph is the clutter that might result from edges between the visible nodes, since a node might be connected to multiple nodes in the previous circle (we call these nodes, the parent nodes). The clutter increases rapidly with increasing branching factor of the graph. One way to overcome the resulting clutter is to connect the node only to one of its parent nodes (chosen according to a user-defined criterion) as suggested in [YFDH01].

Another challenge when using the radial layout is space efficiency. In its standard version, the layout places all the nodes in a certain level on a circle. This results in considerably small node sizes when the number of nodes in the same level is large. For example, the largest circle in an available screen area of 1000x1000 pixels can accommodate 230 nodes only, given that each node has a perimeter of 10 pixels, and the nodes are spaced by 1 pixel. This corresponds to a maximal average branching factor of 6 in case the outer circle is accommodating the 3rd level, which limits the applicability of the view to graphs that have a small branching factor. Reducing the node size does not overcome the problem since this will hinder the ability to map visual attributes to it (which is of particular importance in this work).

In this work, the above issues are addressed and improvements are introduced to the layout, based on two assumptions:

- Nodes closer to the central nodes are more important and relevant for the exploration task; in many applications, the first three levels contain most of the information needed for the analysis.
  For example, when analyzing the products bought by a customer, it is interesting to see who else bought one or more of these products, and what they in turn have bought, while further levels are less relevant. This assumption is exploited by allowing different scales for the nodes in each level, which simulates a fish-eye effect.
- Similarly, the edges between nodes in the first levels are more relevant for the node-centric analysis, while edges in further levels are less relevant.
  In the customers-products example, it is of interest to see how the products bought by a customer are related to the other customers who bought some of them, and find if there are customers who bought many of these products, or to see if two of these products are bought very often together.
  This assumption is exploited by restricting edge drawing only to the first two levels, which enables stacking of nodes in the second level. This in turn, enables drawing larger nodes. The drawn edges are bundled to reduce clutter and to understand associations between nodes.

Using the above ideas, two views for network exploration are implemented in this work:

- **Nodes-only exploration**: the links are ignored [Fig. 31-a], and the view treats the nodes of each level as one group of nodes. In addition, several methods to sort and manipulate the nodes are offered. The main focus in this view is to analyze the attributes of the nodes in each level.
- **Nodes with level-1 edges exploration**: the edges between the nodes in the first and the second level are drawn and bundled [Fig. 31-b].

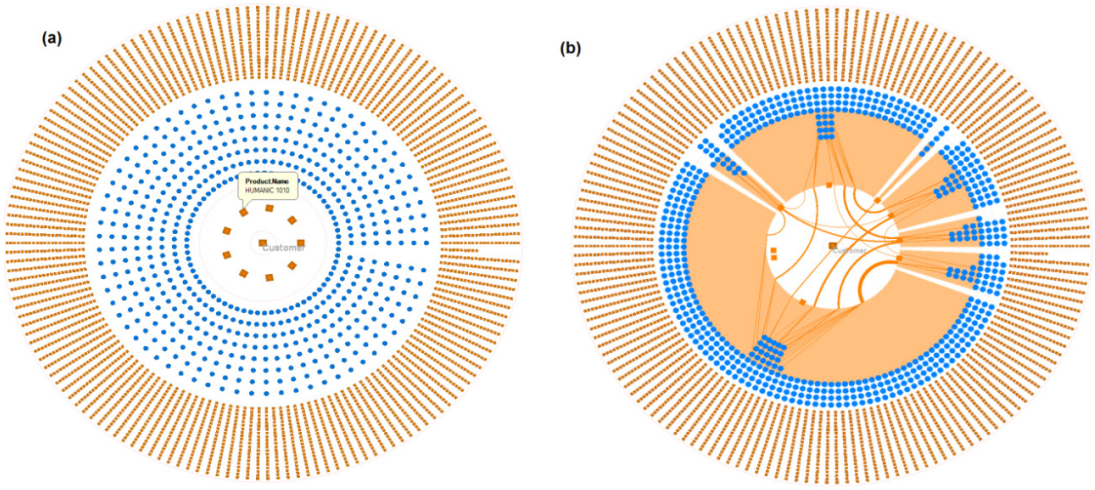

Fig. 31: Graph Exploration Modes

It is worth mentioning that in case of a 2-mode (bipartite) graph, the nodes in the same level are of the same mode and of different mode compared to nodes in the previous and the next rings. This means that the edges in this case can exist only between nodes of consecutive levels.

The number of levels to view, and the size of each ring can be customized by the user, which helps zooming into some rings zooming out of other rings out to achieve a basic focus-and-context navigation.

4.2. Nodes-Only Exploration

This view places the nodes of the neighboring sub graph of the central node in successive rings around it, without visualizing the edges between them. To increase the space efficiency, each ring might contain a number of concentric circles and the nodes in this ring are distributed among these circles [Fig. 32]. The number of circles in each ring is determined in order to fit the number of nodes in the ring while keeping the desired node size. The nodes are distributed in the circles with equal spacing (in terms of polar coordinates).

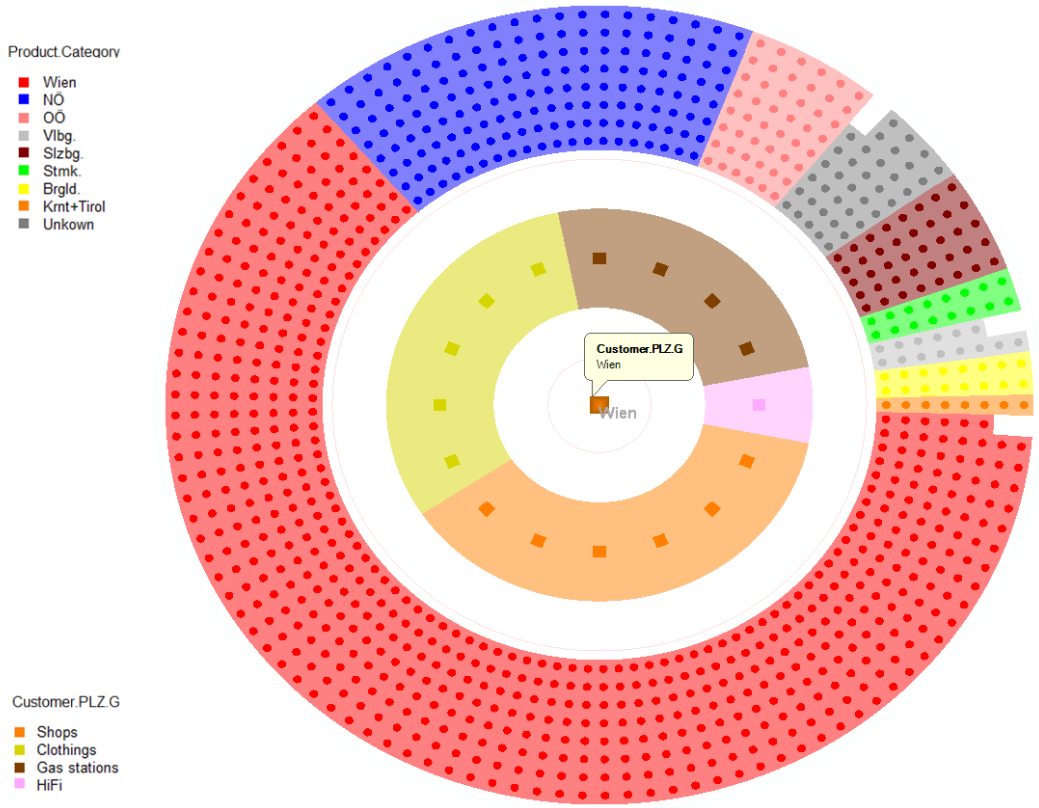

Fig. 32: Nodes-only exploration

The view provides several manipulation operations on the set of nodes including node sorting and selection, as well as viewing multiple relations. It can also receive highlighting, filtering and query commands from the attribute analysis view.

4.2.1. Node sorting

To ease understanding the distribution of an attribute among the nodes in each level, these nodes need to be sorted according to this attribute and placed appropriately. The sorting process depends on the type of the attribute (ordinal / categorical) and it should takes into consideration that the ring contains multiple circles.

4.2.1.1. Sorting by numerical attributes

In case of a numerical attribute, the nodes are sorted by this attribute and placed in a counterclockwise order in the ring (beginning from 0˚). The nodes at the same angular position (due to multiple circles) are ordered from the inner to the outer circle [Fig. 33]. To reveal the order, the color of graphical nodes is mapped from the attribute used for sorting. It can also be assigned to fill the background of the nodes (with lower opacity).

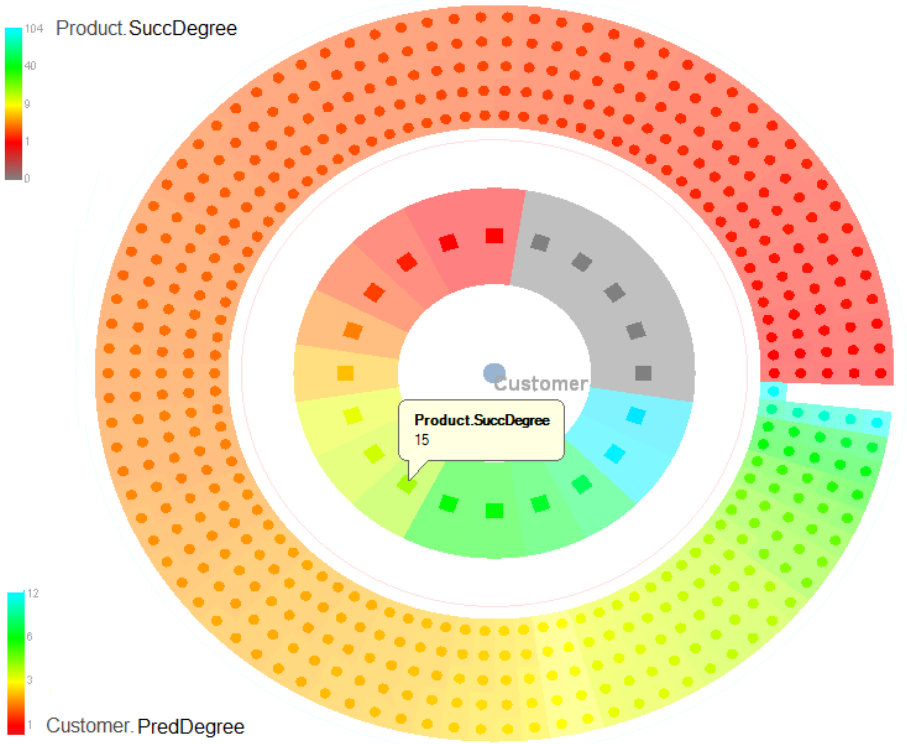

Fig. 33: Sorting nodes by an ordinal attribute

In Fig. 33, the nodes in the first level were assigned color based on their "SuccDegree" (the number of nodes in the 2nd ring they are linked with), and the nodes in the 2nd level were assigned color based on their "PredDegree". This helps giving a general idea about the degree of inter-connectivity between the two levels without visualizing the links. Nodes from one level that are highly (or lowly) connected to the nodes in the other level are easy to spot and to estimate in quantity. The nodes in the first level that are not connected to any node in the second level are assigned the gray color. These nodes are only connected to the central node.

4.2.1.2. Sorting by a categorical attribute

In case of a categorical attribute, the ring is divided into sectors, whereby each category present in the ring nodes is assigned a sector which size is proportional to the number of nodes that belong to this category [Fig. 34]. The sectors can be sorted by size or by the default order of their categories, i.e., the alphabetical order.

Additionally, the nodes in each sector can be further sorted by another attribute. In case of a numerical attribute, the nodes of higher values for this attribute are placed in inner circles or vice-versa, and along each circle the node are sorted counterclockwise [Fig. 34]. The value of this attribute is mapped to the node color, while the sector color is mapped from the sector category.

In case the secondary attribute is categorical, the problem of placing these nodes resembles a radial space-filling hierarchy visualization which has been studied thoroughly by Stasko and Zhang [SZ00]. However this work uses a simpler approach which groups the sector nodes by category and places them in a similar manner as in the case of a numerical secondary attribute (with the more frequent sub-categories placed in the outer circles).

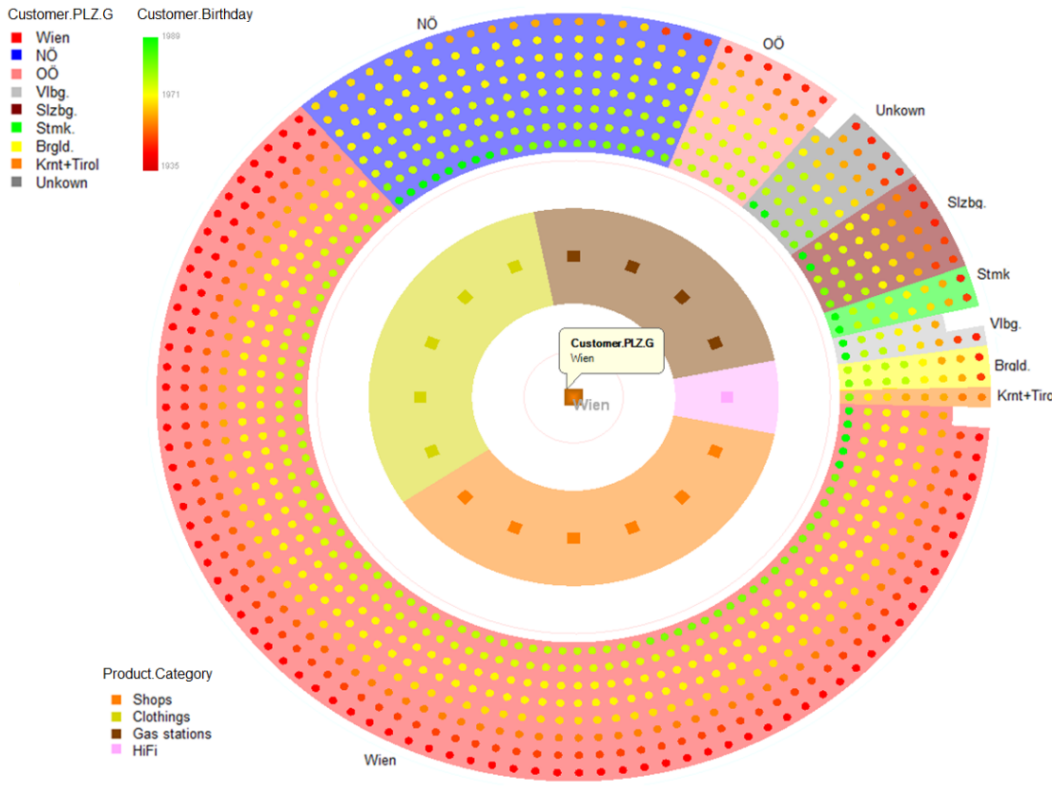

Fig. 34: Sorting nodes by a categorical attribute and by an ordinal attribute.

If the categorical attribute is an aggregated attribute (computed from the related nodes as the mode value), the distinctivity of the mode value is mapped to the saturation of the node, and the nodes in each sector are sorted according to this numerical attribute (with higher values outwards). This places nodes with low distinctivity on the inner side of the ring, to distinguish them from those having higher distinctivity.

Fig. 35 shows an example where the mode gender for each product in level-1 nodes, along with the distinctivity are mapped to the node color. Nodes with gray values (de-saturated) are placed in the inner circle. They represent products where neither male nor female customers were remarkably dominant in the purchases of these products. The saturation of level-1 nodes increase when either gender becomes more dominant. In Fig. 35, a node with distinctivity = 0.5 for the mode gender "Female" is selected and nodes related to it are in level-2 (and level-0) are highlighted.

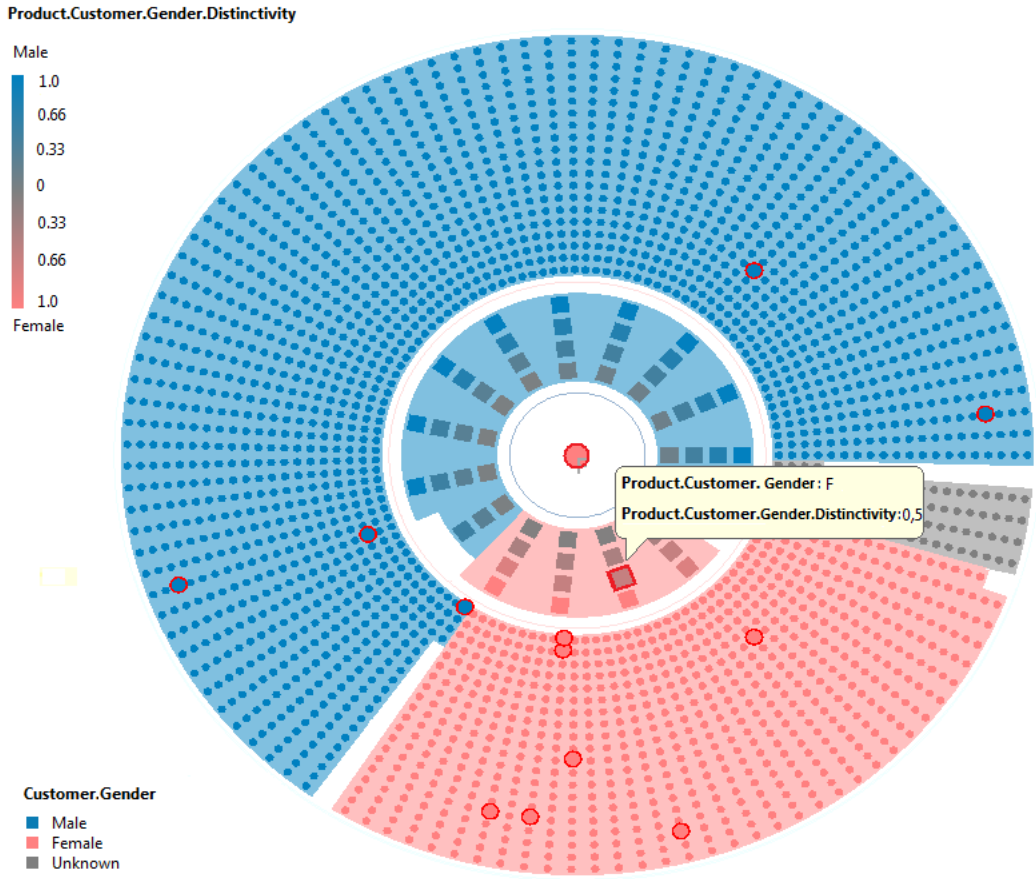

Fig. 35: Sorting by an aggregated attribute.

4.2.2. Node selection

Using the mouse, the user can select a node to see nodes related to it. The selected node and its related nodes will be highlighted, while the rest of the view can be chosen to be blended. Fig. 36-a shows an example of node selection, where the nodes of the customers who bought the selected product are highlighted. In this figure, the nodes in the second ring are sorted by the customer birthday, which helps finding how the age of highlighted customers is distributed among customers who bought one of the products in the first ring.

It is also possible to select multiple nodes within the same ring can. Each node in the two neighboring rings is assigned weight which indicates the number of selected nodes it has relations with. A node in a farther away level receives a weight which is the sum of the weights of nodes it is related with. This weight is an additional layout attribute named "EntityName.SelectionWeight", and hence can be used to sort or filter the nodes in any ring.

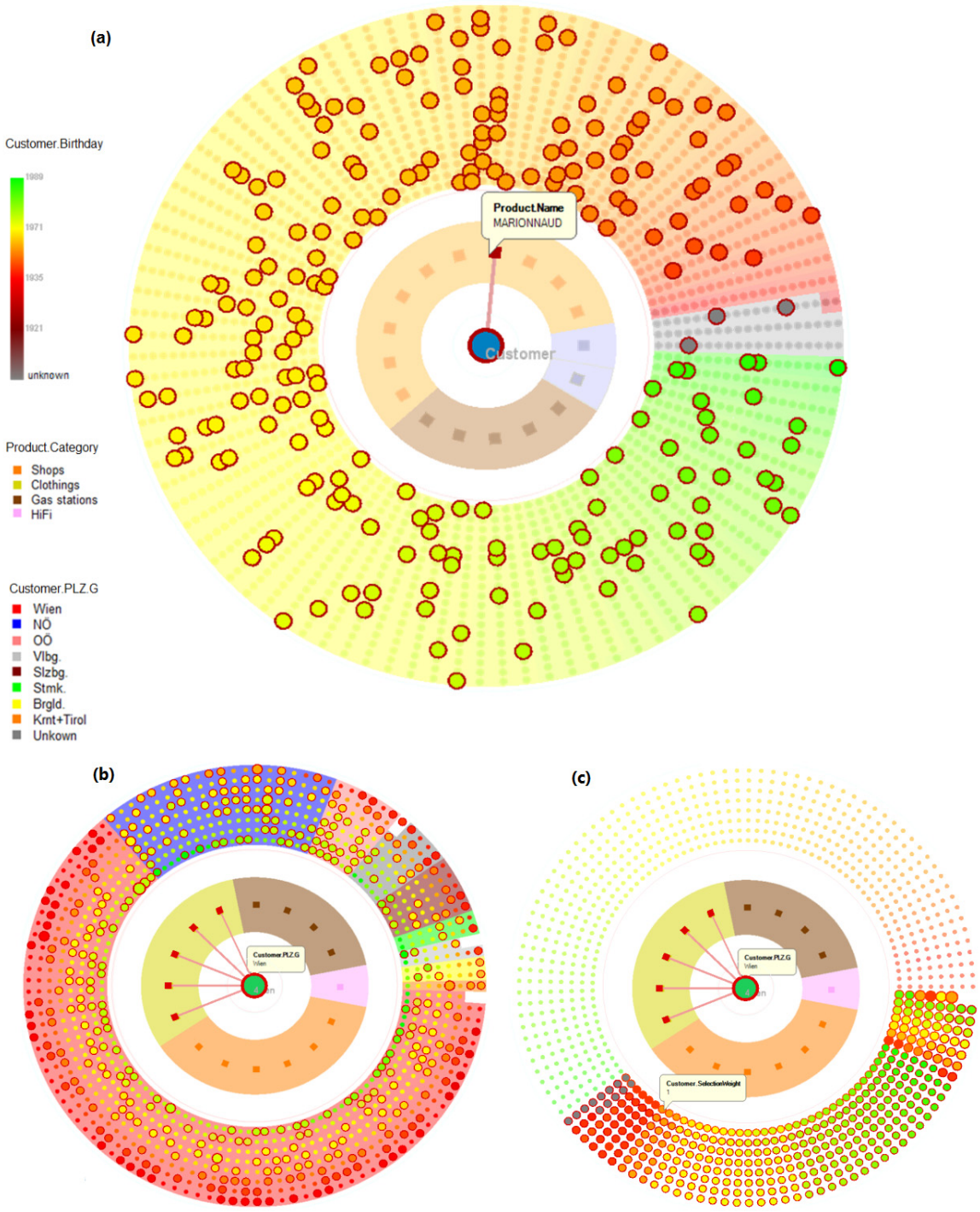

Fig. 36: (a) Single node selection (b) Multiple node selection: related nodes highlighted. Union is used to combine the related nodes (c) Multiple node selection: highlighted nodes in (b) are sorted by weight (number of selected node they are linked with).

Two modes can be employed to highlight the nodes in case of multiple node selection:

- **Union**: nodes with non-zero weight are highlighted and scaled according to their weight [Fig. 36-b]. They can be optionally sorted according to their weights [Fig. 36-c].
- **Intersection**: a node in the two neighboring rings is highlighted if it is linked with *all* of the selected nodes. Nodes in further levels are highlighted if they are linked to a highlighted node. This helps finding for example customers who bought all the selected products, and what in turns they have bought and how common.

4.2.3. Visualizing multiple relations

As mentioned in section 1.2, multiple relation instances can exist between two nodes in the graph, and a relation attribute can take different values in each instance.

In this view, the instances of relations between the central node and each of the nodes in the first level can be visualized. They are drawn as graphical objects on the straight line connecting the central node and the corresponding level-1 nodes [Fig. 37].

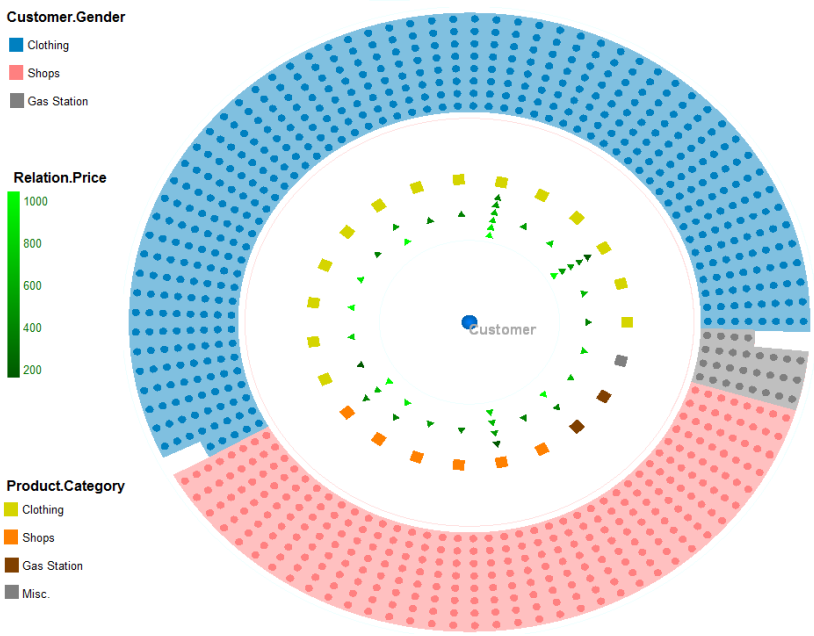

Fig. 37: Visualizing multiple relations.

The visual properties of these graphical objects can be mapped from the relation attributes in a similar manner as the visual mapping of the graph nodes. Also, using the mouse the user can select a relation instance to see its attributes.

4.3. Nodes with Level-1 Edges Exploration

The nodes-only view gives a general idea about the connectivity between the nodes in the first and the second rings (by means of layout attributes), and the nodes connected with a particular level-1 node can be highlighted by means of selection. But it is still advantageous to see all the edges between the two levels in details.

We assume that the graph is 2-mode (bi-partite) and then discuss how the view can be used for 1-mode graphs.

We can differentiate between two types of level-2 nodes in 2-mode graphs:

- **Nodes with PredDegree = 1**: such a node is connected to one level-1 node only, and hence needs to be placed in the neighborhood of its parent. These nodes are called *exclusive nodes* (exclusive for one parent node).
- **Nodes with PredDegree > 1**: such a node poses the challenge of finding an appropriate placement which intuitively implies its relationship with all of its parent nodes. These nodes are called *shared nodes.*

The same as in anchor maps [M08], we first place the level-1 nodes on the circumference of a circle, and place the level-2 nodes with PredDegree = 1 in neighborhoods specific to their parents. We consider this neighborhood to be defined by the angular sector assigned to the parent node.

Unlike anchor maps, we place the rest of the level-2 nodes in the above sectors using two different strategies:

- **Placement at a random parent**: each level-2 node is placed in the sector of one of its parents and edges are drawn from this node to all other parents. However, the readability in this approach was low as will be discussed in section 4.3.1.
- **Node repetition and linking**: an instance of each level-2 node is placed in the sectors of each level-1 node it is related with.

4.3.1. Placement at a random parent

For each level-2 node, one of its parent nodes is chosen randomly (with a probability proportional to the parent node-degree) and the node is placed in the sector of the chosen parent node. The nodes of a sector are divided further into sub-groups depending on their other parent nodes.

Exclusive nodes in the sector are placed in the outmost part of the sector and distributed into a user-defined number of circles. Shared nodes that have only one other parent, are grouped into sub-groups in the sector according to this parent. Only one edge is drawn from this sub-group to that parent. Shared nodes that have two or more other parents are placed in the sub-group of one of these parents in a manner similar to how the shared nodes are assigned to sectors. For such a node, a link is drawn from it to each of its parent nodes other than the sector node and the sub-group it lies in. Adjacent sectors are allowed to overlap and in this case, the nodes they have in common are drawn in the overlapping area [Fig. 38]. The subgroups of a sector can also overlap in the same manner.

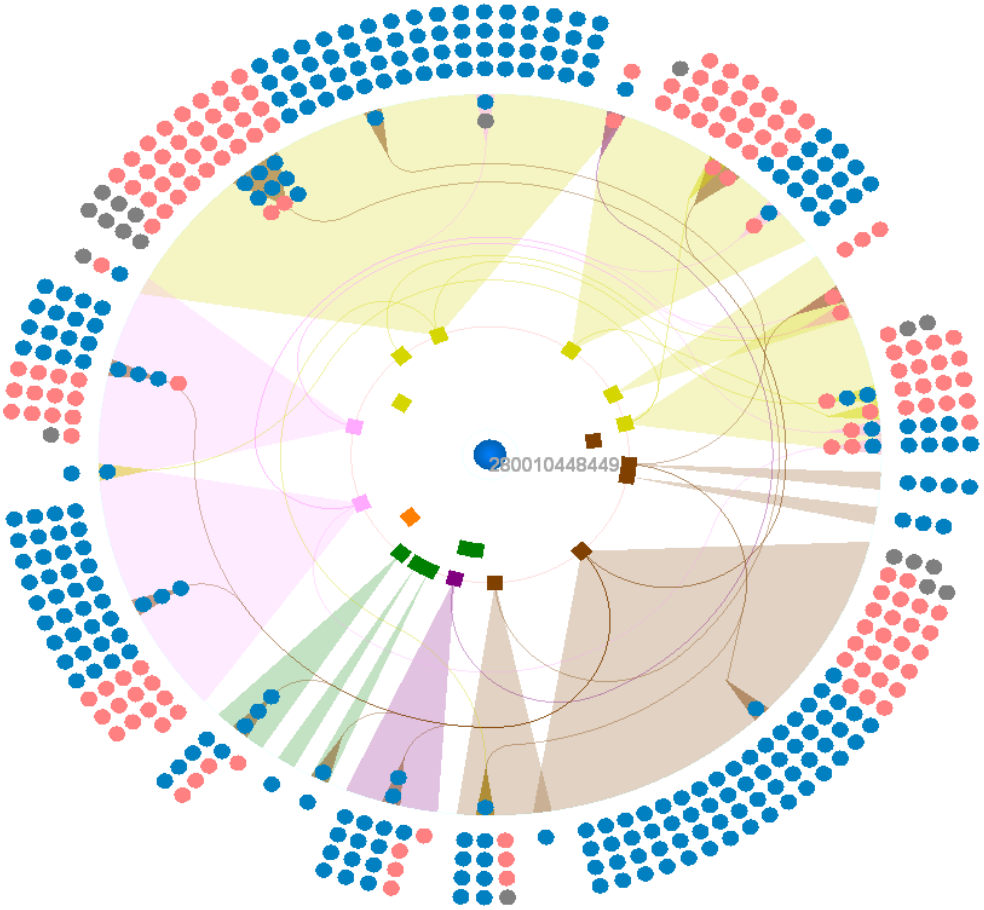

Fig. 38: Drawing links using placement at a random parent.

The sectors are assigned angles proportional to the number of related nodes they have. The links from a level-1 node to its subgroups in the sectors of other level-1 nodes are drawn as a branching set of C^2 piecewise circular curves, which start together as an arc tangential to the radius through the source node, and continue as an arc concentric with the rings. Then, these links branch into each sub-group by means of an arc tangential to the radius through the center of this subgroup, and also branch to related nodes in other subgroups in that sector by mean of straight lines.

The concentric arcs of different curves from different level-1 nodes are assigned different radii and are assigned the colors of the level-1 nodes they originate from, to help differentiate between them.

No links are drawn from a level-1 node to nodes in its sector. Instead, a sector with the node color and low opacity is drawn which resembles the grouping technique mentioned by Herman and Marshall [HM00]. Level-1 nodes which have no related nodes in level-2 are stepped back in the ring (assigned smaller radial coordinate).

Optionally, the sectors can be ordered to maximize overlapping using a greedy algorithm. The algorithm finds a suboptimal Hamiltonian tour in the weighted complete graph whose nodes are the level-1 nodes, and each edge is weighted by the number of level-2 nodes shared between its end-nodes.

The algorithm works by passing on these edges in the descending order of their weights and add the current edge as long as it does not close the tour before all the nodes are covered. The resulting cycle defines an ordering of the level-1 node in their ring, so that the overlapping is maximized (since edges with larger weights where chosen first, and such an edge corresponds to placing two level-1 nodes that share a large number of child nodes into level-2).

The user can specify the number of circles in the second ring which helps producing larger nodes, and can control the range in which the edges are bundled. Additionally, the nodes in level-1 and the nodes in each sector can be sorted by an attribute of their entity types (which ignores the maximization of sectors overlapping). This sorting works in the same manner as in section 4.2.1 and node selection works also the same as in the case of the nodes-only view.

This view helps finding the nodes exclusive to each level-1 node, and getting an idea of the degree of interconnection between the two levels. However two main drawbacks limit the usefulness of this approach:

- Not all of the level-2 nodes related to a level-1 node are placed in its sector: they can be randomly distributed into subgroups in other sectors without a meaningful purpose from the user point of view.
- Drawing edges as piecewise circular curves can result in long edges with a high degree of bending, which is disapproved by the aesthetic criteria for graph drawing mentioned in section 2.1. In fact, with a growing number of edges, this approach fails to reduce clutter or abstract the edges with an informative visual representation.

4.3.2. Node repetition and linking

To overcome the drawbacks of the previous edge drawing approach, we investigated another approach based on node repetition and linking. In this approach, an instance of each level-2 node is placed at the sector of each of its parent nodes. A link is drawn between each pair of instances of the same node. In each sector, the links which connect instances in it with instances in another sector are bundled together. The sector is assigned an angle proportional to the number of node instances in it.

In this work, two bundling techniques are implemented to show the links:

- Bundling by arcs: an arc connects each pair of sectors, and the arc thickness and opacity represent the number of shared nodes between them.
- Hierarchical edge bundling: a B-spline curve connects each pair of sectors, and the curves are also bundled according to a hierarchical grouping of an attribute of the level-1 entity type.

4.3.2.1. Bundling by arcs

A link between a pair of instances in different sectors is drawn as a straight line segment which connect each instance with the appropriate exit point in its sector, and an arc which connects the two exit points and is tangential to the radii that pass through them [Fig. 39].

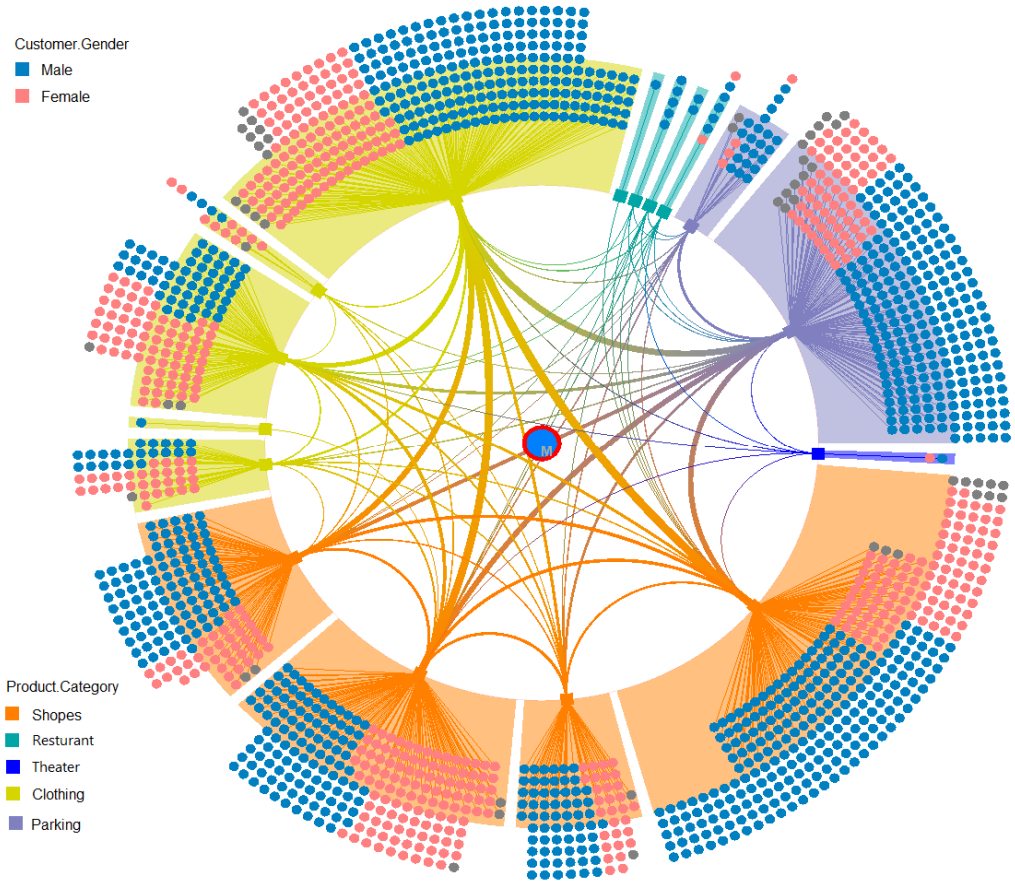

Fig. 39: Draw links using node-repetition-and-linking with bundling by arcs.

The arcs originating from a sector can be chosen either to exit from its center, or to be distributed (stretched) to exit from different points along the sector, with a user-defined stretching degree (Fig. 40-a and Fig. 40–b). In case of different exit points, an arc is assigned exit point depending on the other sector it is connected to: the closer the other sector is to one of the two sides of the first sector, the closer the exit point is to this side, while arcs coming from further sectors pass though exit points closer to the middle of the current sector. This helps reducing the edge crossing.

The arc thickness is proportional to the number of links bundled in it, which is equal to the number of nodes shared between the two sectors it connects. This number can be mapped optionally to the arc opacity too. The arc thickness and opacity help finding which two sectors share a large (or a small) number of nodes (Fig. 40-c and Fig. 40-d). This number denotes the number of paths of length 2 in the graph between each pair of level-1 nodes.

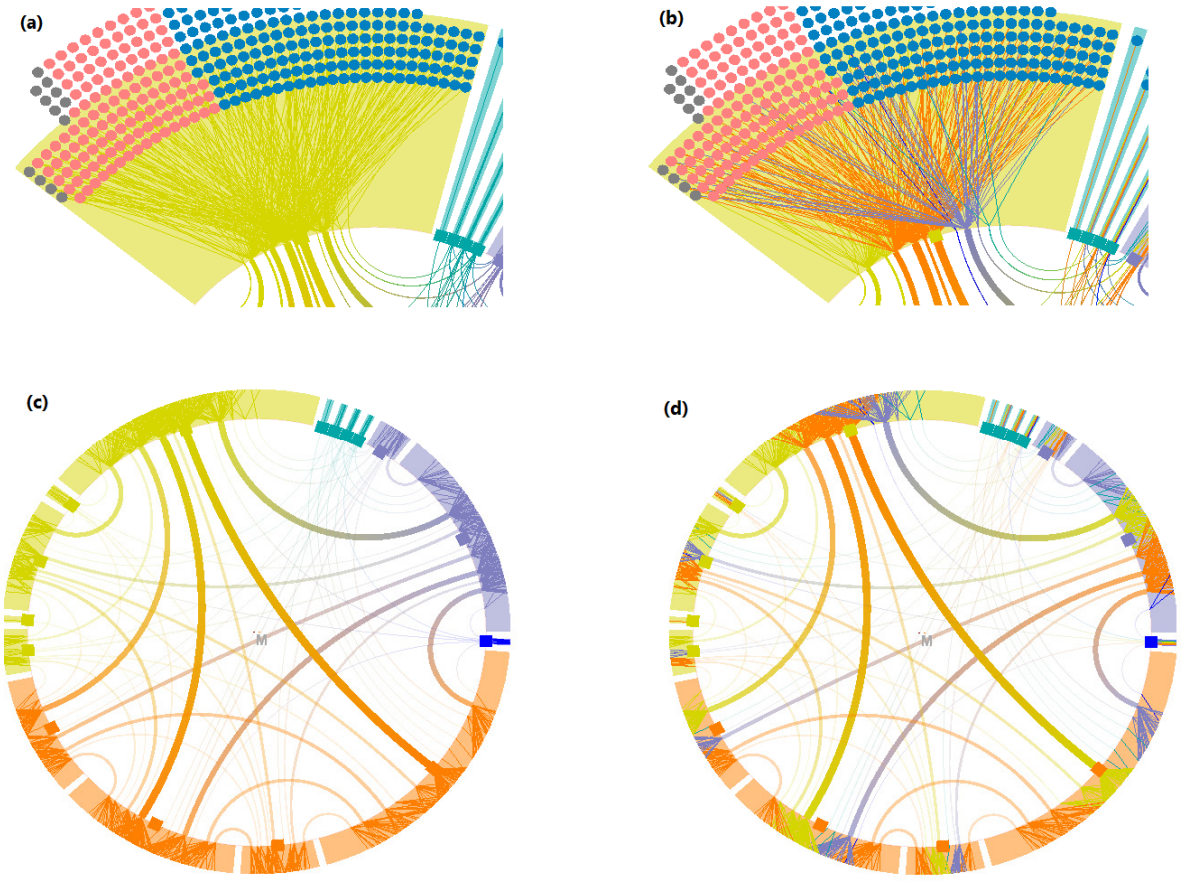

Fig. 40: Layout and visual parameters: a) Stretching edges along the arc and using forward coloring b) Higher stretching degree with backward coloring c) Forward coloring with edge blending to find inter-sector relations. d) Backward coloring with edge blending to find attribute values arriving at each sectors.

The arc color depends on the color of the sectors it connects. The color can be interpolated along the arc in two different ways:

- **Forward Coloring:** each end of the arc takes the color of its sector. This helps seeing how many links are going out from each sector, and how they are further distributed to other sectors [Fig. 40-a, Fig. 40-c].
- **Backward Coloring:** each end of the arc takes the color of the sector of the other end. This helps seeing which attribute values the level-1 nodes of the connected sectors mostly have [Fig. 40-b, Fig. 40-d].

In each sector, the level-2 nodes that are exclusive to it are placed in the outer part, and the shared nodes are placed in the inner part. These nodes are distributed into a user-defined number of layers to produce larger nodes [Fig. 40-a and Fig. 40-b]. Both exclusive and shared nodes can be sorted according to an attribute of their entity type.

The bundling by arc gives a detailed view of the inter-sector links, and connects the sectors independently from their attributes. The next view tries to exploit existing coherences in level-1 node attribute values to produce an attribute-based bundling.

4.3.2.2. Hierarchical edge bundling

With a growing number of level-1 nodes, bundling by arcs produces a cluttered visualization since each pair of sectors is connected with a separate arc which is drawn independently from other arcs or from the sectors attribute values. A more elaborate, attribute-based bundling can reduce the clutter, by placing nodes that belong to the same group or category next to each other, and bundling their links together.

The attribute-based grouping / categorization defines a hierarchy over the nodes in level-1, and the links between these nodes are adjacency relations between the leaf nodes of this hierarchy. In this work, the hierarchical edge bundling technique mentioned in section 2.1.4 is applied to produce a less cluttered visualization of the edges [Fig. 41].

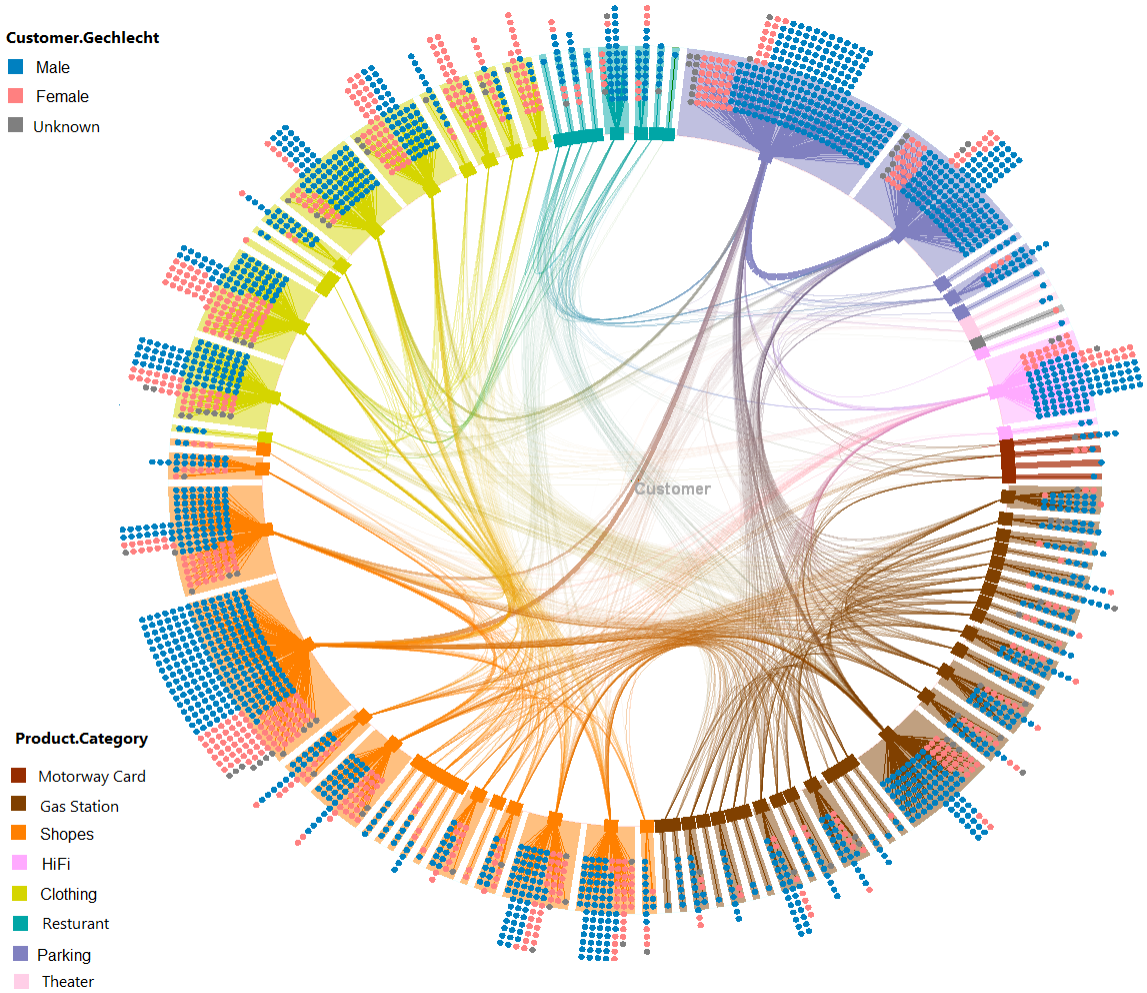

Fig. 41: Attribute-based hierarchical edge bundling.

In order to generate the presented view, the user first selects the attribute to be used for grouping. The grouping hierarchy of the values of this attribute in level-1 nodes is computed, and these nodes are sorted according to their hierarchy (so that nodes that belong to the same group lie next to each other). The nodes are then placed along the level-1 ring in the same manner as in the bundling-by-arc approach.

The standard tree radial layout is used to compute positions of the hierarchy nodes inside the ring. The link between each pair of sectors is then bundled to this hierarchy (with a user-defined bundling factor as explained in section 2.1.4).

The same as in the bundling-by-arc approach, the links are colored by interpolating the color of two sectors they connect either forwards or backwards, and the link opacity denotes the number of shared nodes between the two sectors it connects. Also, the links can originate from the center of the sector or be stretched along the sector arc.

The hierarchical edge bundling, along with attribute based link coloring, helps finding associations between the groups / categories in level-1 nodes. In Fig. 41, one can recognize that there is higher association between gas stations, parking coupons and motorway cards. Also, gas stations were more often associated with a number of shopping centers, than with clothing shops, possibly since commuting with one's own car is probably more often necessary when buying in shopping centers.

4.3.2.3. Discussion

The presented nodes-repetition-and-linking method solves the drawbacks of the random-parent method since it places all the nodes related to a level-1 node in its sector. This property is of particular importance for attributed graphs, since it enables straightforward understanding of the distribution of an attribute in these nodes, which would be difficult in case the nodes are scattered into different sectors or locations (as is the case for example in anchor maps).

Also, this properly eliminates the need to draw edge between level-1 and level-2 nodes. Rather, links are used to connect instances of level-2 nodes, and thanks to the above bundling, these links make it easy to recognize which level-1 nodes share a large (or small) number of level-2 nodes. This is done by blending edges with low multiplicity [Fig. 40 and Fig. 41].

Also, the links are more intuitive to follow than in the random-parent method since they are shorter and have a lower degree of bending (particularly in the case of long links). Also, existing bending is not arbitrary but intuitively related to the sectors that the bent link connects.

Section 5.3.2 discusses how the implemented view can be efficiently used to find relation-relation associations between a set of nodes in attribute graphs.

4.3.3. Applicability to 1-mode graphs

In case of 1-mode graphs, there are four types of edges that can be selected by the user to be incorporated in the visualization:

1. Edges between level-1 nodes: these edges can be individually drawn using the same methods mentioned in section 4.3.2, since they represent inter-sector links. The link width represents the number of relations between the connected nodes. Such a link denotes the multiplicity of the length-1 path in the graph between each pair of level-1 nodes.
2. Edges between multiple instances of the same level-2 node in different sectors: these edges can be also individually drawn using the same methods that were used in 2-mode graphs, with link thickness denoting the number of shared instances between the connected sectors. The resulting inter-sector links represent the number of length-2 paths in the graph between each pair of level-1 nodes.
3. Edges between two different level-2 nodes in different sectors: these edges can also be individually drawn using the same methods for multiple instances of the same node. The resulting inter-sector links represent the number of length-3 paths in the graph between each pair of level-1 nodes (since such link d. Visualizing these links enables finding which two level-1 nodes have sets of related nodes that are in turns highly inter-related.
4. Edges between two different level-2 nodes in the same sector: these edges indicate how much the nodes in each sector are inter-connected. In this work, these links are not directly visualized, but rather, their density can be mapped to the sector color and/or opacity. A detailed visualization of them can be obtained by choosing the level-1 node to which their sector belongs to be the central node (and hence these links become edges between level-1 nodes).

For example if the graph represents friendship relations between people, the first type of edges helps showing existing friendship relations between the friends of the person in the center. The second type of edges shows if two people in level-1 have a lot of friends in common. The 3rd type of edges shows if two persons in level-1 have sets of friends so that many friendship relations exist between these two sets. The 4th type of edges shows if a person in level-1 has a set of friends that have many friendship relations between them.

To analyse more than one type of edges at the same time, mutilple views can be created (one for each type). Alternatively, the links can be combined in the same visualization using the bundling-by-arcs technique. Since it layouts the links separately, it is possible to overload an arc with multiple types of links. The links on the arc are differentiated by means of color [Fig. 42].

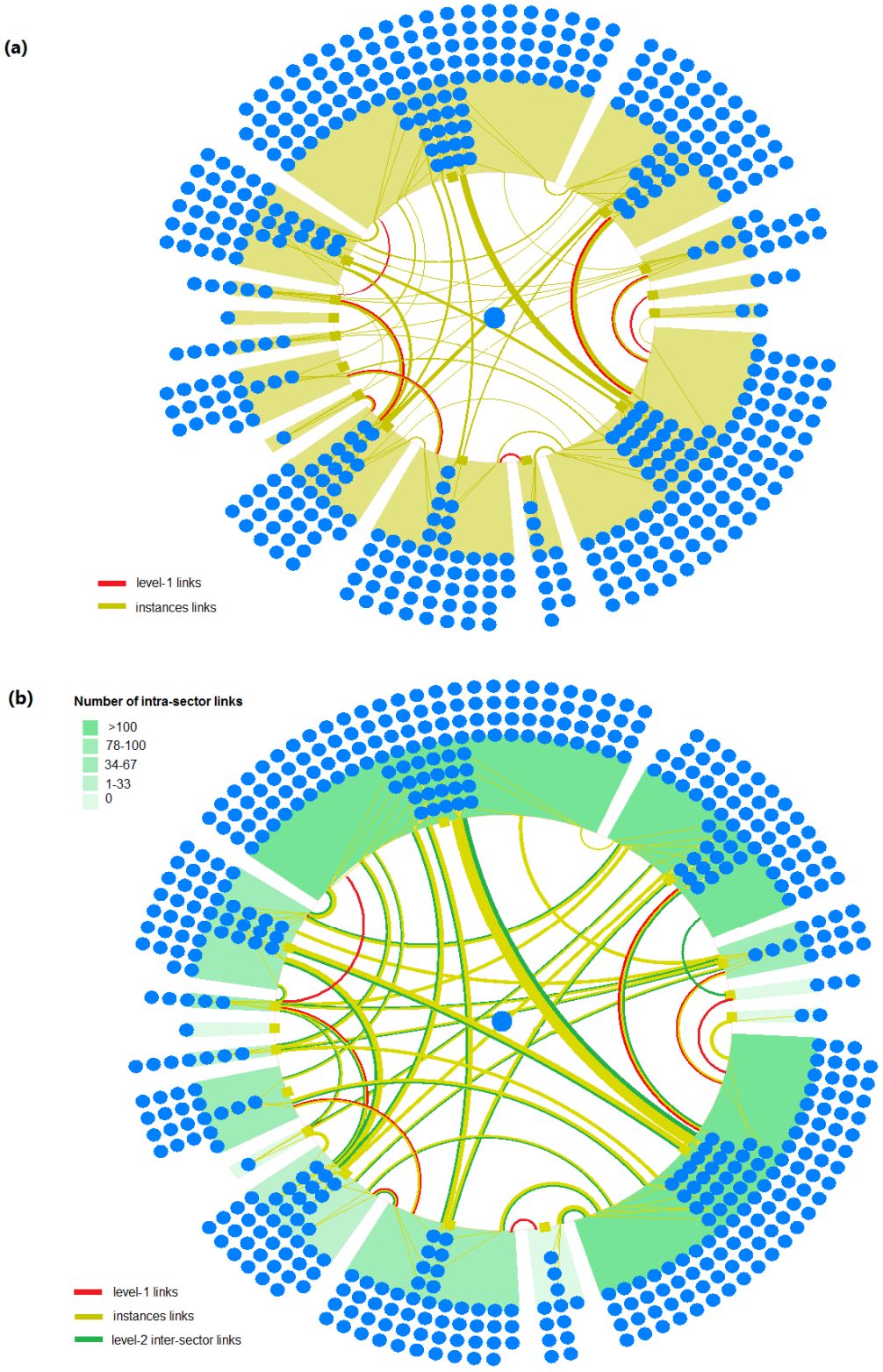

Fig. 42: Combining multiple types of edges: (a) links between instances and links between level-1 nodes. (b) links between level-2 nodes instances, links between level-1 nodes and links between different level-2 nodes.

4.4. Linking and Brushing

As mentioned by Keim [K02], "*the idea of linking and brushing is to combine different visualization methods to overcome the shortcomings of single techniques. Interactive changes made in one visualization are automatically reflected in the other visualizations. Note that connecting multiple visualizations through interactive linking and brushing provides more information than considering the component visualizations independently* ".

In this work, the attribute analysis view (which enables creating histograms of attributes) is linked with the graph exploration view in both ways:

- Using the attribute analysis view, queries are visually defined either to highlight nodes in the graph exploration view, or to show their results in it. Also, the former view defines the mapping of the attributes to visual properties.
- The layout attributes which the layout computes for each entity instance can be analyzed in the attribute analysis view to see their distribution and create conditional histograms of other attributes based on them. Also the later view enables using these attributes in other modules.

4.4.1. Node highlighting

Nodes in the view can be highlighted based on their attribute values. The visual query definition method explained in section 3.4.2 is used to specify the desired conditions on the attributes for the nodes to be highlighted. Highlighted nodes are drawn larger than other nodes and are emphasized by assigning a highlighting color to their circumference. This color can be adjusted by the user.

Fig. 43 shows an example of node highlighting. In this example, the sample query defined in section 3.4.2 is extended so that it restricts the selected customers to those who have has bought more than 10 different products. The query condition is then expressed as:

$$1979 \leq Customer.BirthDay \leq 1989 \ \wedge \ Customer.Gender \in \{F\} \ \wedge \\ 10 < Customer.NodeDegree$$

A nodes-only view of customers who bought from "ASFINAG", and the products they bought in turn are shown. The viewed customers are grouped and colored according to their provinces in Austria. The viewed customers that additionally satisfy the query condition are highlighted. The highlighting reveals in which province the young female customers who bought from at least 11 different shops including "ASFINAG" live.

More nodes can be highlighted using another query. In the example of Fig. 43, the shops in which the customers in level-1 bought are sorted according to their categories. A histogram of the layout attribute "Product.PredDegree" is created. A range of the PredDegree values that are larger than 10 is selected in the histogram. The product nodes in the view that belong to this range are highlighted. These nodes represent the products that are bought from more than 10 customers in the level 1. Thanks to sorting by category, one can recognize that shops are the most shared "products" among customers who bought from "ASFINAG".

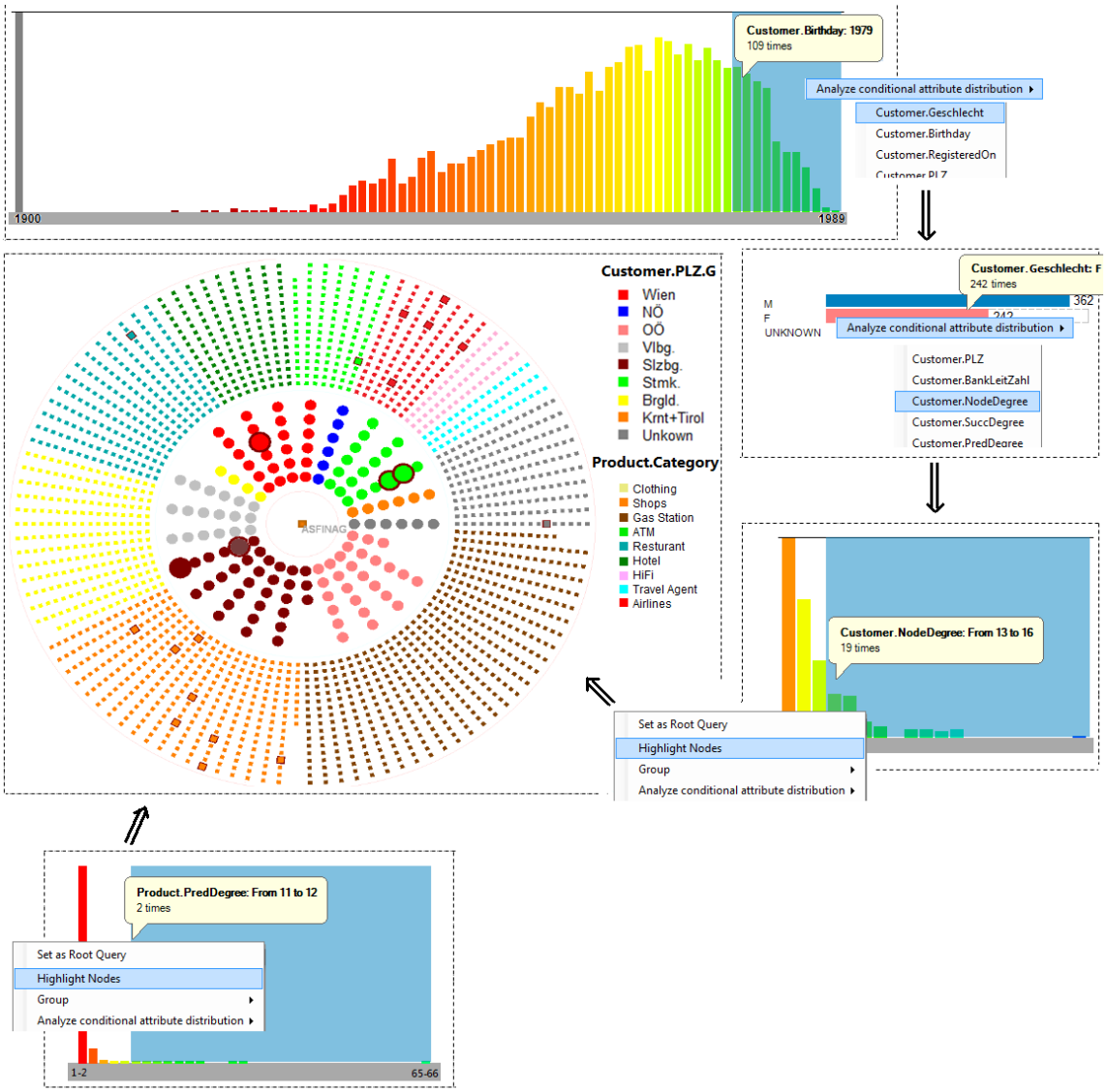

Fig. 43: Node highlighting.

4.4.2. Visualizing queries results

The views presented in this chapter enables detailed node-centric exploration of a part of the graph in the neighborhoods of the central node, with the ability to see the attributes of the nodes in this subgraph, and how they are related.

Besides exploring the neighboring subgraph of a specific node, the same views can be used to explore a set of nodes, not necessarily related to a central node, to see their attributes and what nodes they are related to. This set of nodes is the result of a query which selects them based on specific criteria on their attributes. To preserve the consistency with the definition of the layout [4.1], a dummy node is created and connected to all the nodes in the query result, and it is used as the central node in the layout. This node represents the query and has one intrinsic attribute only, which describes this query. Not only the results of the queries can be visually explored, but they can be also visually defined, as has been discussed in section 3.4.2.

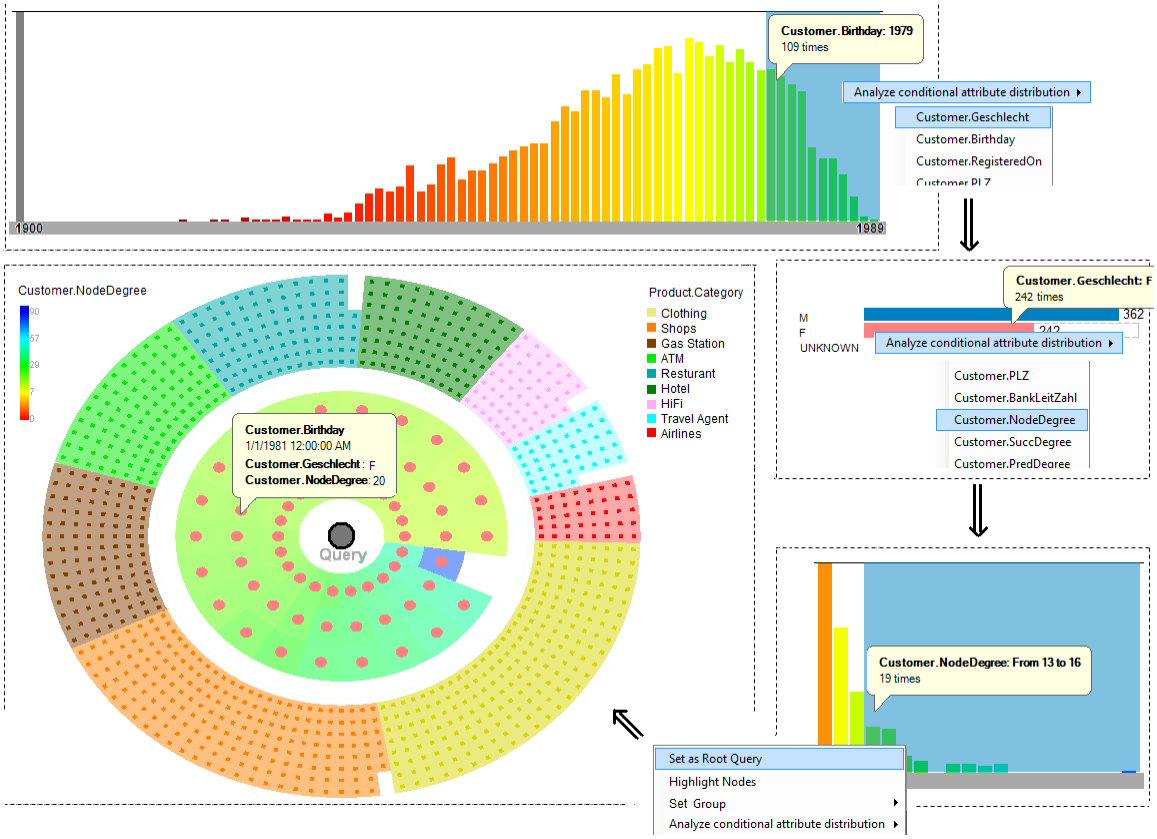

Fig. 44: Defining visual queries and exploring their results.

Fig. 44 shows an example of how queries are visually defined and how their results are visualized. In this example, a similar query to the one defined in the previous section is used to find female customers whose birthdays are between 1979 and 1989 and who bought from 13 or more different shops.

The query is visually specified by first creating a histogram of the "Customer.Birthday" then selecting the desired range and creating a conditional histogram of the attribute "Customer.Gender". The process is continued by selecting the "F" category of this attribute, then creating a conditional histogram of the "Customer.NodeDegree", and finally selecting the desired range in it and choosing as "root query". The system then starts executing the query by creating a query node, and connecting it to all the nodes that correspond to the selected entities. The query node is then visualized as the central node in a graph exploration view, and the set of nodes whose entities fulfill the query are placed in the first level.

All the features presented in this section can be applied to the resulting view (except for viewing multiple relations between the central node and the level-1 nodes). In the above example, the nodes in the 2nd level which correspond to the shops related to the selected customers are viewed and sorted by shop category. As can be seen, the largest category is "clothing". The gender attribute is mapped to the level-1 nodes color, and the NodeDegree attribute is mapped to the sector background to verify that the query was correctly executed.

Chapter 5

Association Analysis

"A man only learns in two ways, one by reading, and the other by association with smarter people", Will Rogers

In this work, association analysis focuses of finding non-trivial disproportionalities in the attribute relational data by using interactive visualization.

Several types of associations are investigated in this work

- **Attribute-attribute association**

  In an attributed dataset, we want to know if the distribution of the values of one attribute in a subset of entities / relations which have specific values for other attributes, deviates considerably from the global distribution of this attribute.

  For example, in the customer entities one might find an association between customer gender and age (female customers are disproportionally more frequent among young customers). Thanks to the extended relation attributes, the same analysis can be applied to the data set of purchases to figure out if there is an association between the customer gender and the product category.

  In this type of associations, the unit of analysis is the category or value range, in the sense that all elements in the data set that fall in the same category / value range are treated as one item in the analysis.

- **Attribute-relation association**

  In an attributed relational dataset, we want to know if the entities related to a specific entity have common values for their attributes. For example, in the customer-product network, we want to know if some products have been bought disproportionally more frequently by young customers, or if there are customers whose purchases contain a specific product category more frequently than other categories. In contrast to attribute-attribute association, the unit of analysis in this type of association is the single entity for one entity type, and the category / value range for the other entity type.

- **Relation-relation association**

  In an attributed relational dataset, we want to know if a considerable number of the entities related to a specific entity, are also related to another entity. For example, we want to know if many of the customers who fueled at a specific gas station, have bought parking coupons from a specific location, or if many of the products bought by a specific customer have also been bought by another customer. Additionally, we want to know if entities which are related to entities of a specific category for some attribute, are also related to entities of another category of this attribute.

To perform association analysis of one of these types, two kinds of visual analysis tools are used:

- **Comprehensive analysis tools**

  These tools intend to visualize all the existing associations in a given set of entities or relations with the purpose of discovering unknown associations.

- **Local analysis tools**

  These tools intend to provide a detailed visualization of the associations related to a specific entity or a specific value of the desired attribute, with the purpose of validating an assumed association.

In addition to the above types of associations, other types can be defined for specific applications. This chapter presents a technique to analyze the flow of support tickets in the support branches of the ticket assignment application presented in section 1.4.2.

Finding non-trivial associations means gaining new knowledge about the relational data, which makes association analysis a fundamental tool in data mining and knowledge discovery in databases. The next sections will discuss the above types of associations in detail, and present the proposed visual analysis tools implemented in this work to find them.

5.1. Attribute-Attribute Association

In one of the sets of the attributed relational data, $X \in \{E_1, E_2, G\}$, we want to see if the distribution $dist_S(A_1)$ of the attribute A_1 in the subset S of X, defined as follows,

$$X \supset S = \{x : A_2(x) = v_2 \wedge \ldots \wedge A_k(x) = v_k\}$$

deviates considerably from its global distribution $dist_X(A_1)$.

In this work, the local analysis enables the user to interactively specify $A_2 .. A_k$ and $v_2 .. v_k$ and observe the conditional distribution of A_1 to find potential associations. The comprehensive analysis requires the user to specify $A_2 .. A_k$ (with the restriction of being categorical attributes), and aims to visually highlight potentially interesting associations between the values. The parallel-sets technique is used for this purpose.

5.1.1. Conditional histograms

The conditional histograms introduced in Chapter 3 enable finding how one attribute of a network component (entity type or relation) is distributed under conditions on the other attributes, with the possibility to compare this distribution with the global one. Changes on these conditions will be reflected on the conditional histogram, which enables a manual search for associations between the attributes. For example we might create a conditional histogram of the product categories in purchases made by female customers, and another conditional histogram in purchases made be male customers and compare the results.

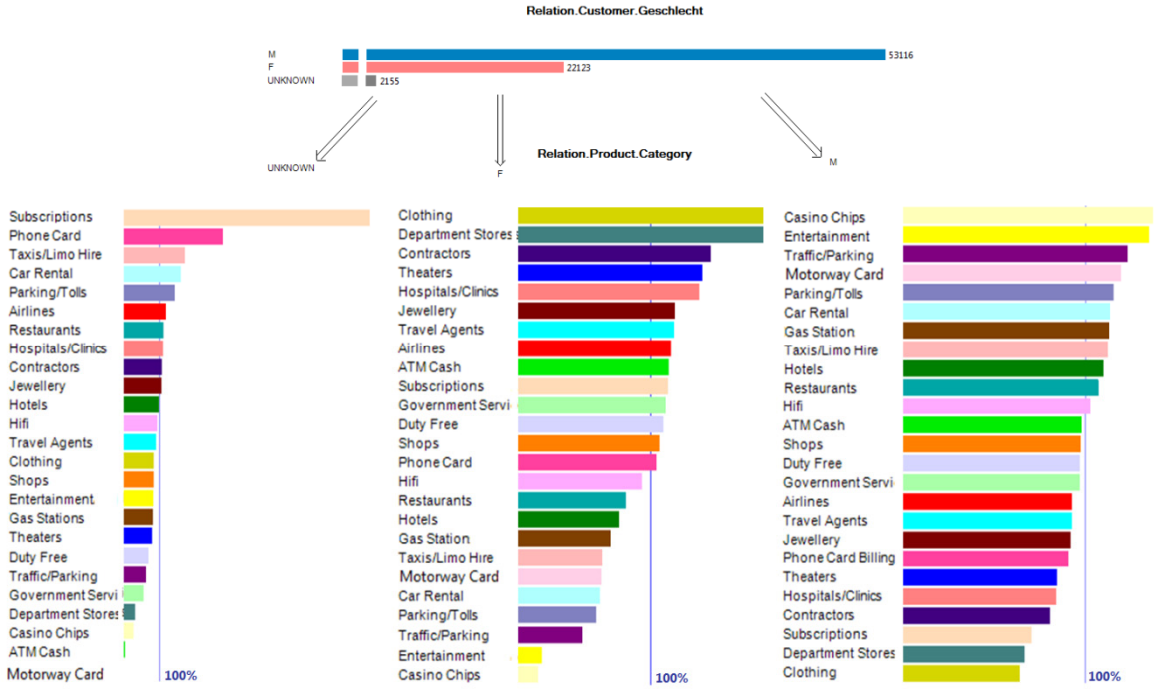

Fig. 45: Association analysis using histograms.

In Fig. 45, a conditional histogram of the attribute “Relation.Product.Category” (the product categories in the purchases) is created for each possible category of the

attribute “Relation.Customer.Gender”. The conditional histograms show the ratio growth for each product category (defined in section 3.4.2) in each case.

One may notice that the purchases of female customers in clothing shops and department stores are remarkably high, while their purchases in gas stations and driving-related products (parking coupons, motorway cards, etc.) are low. Purchases of male customers show roughly the inverse behavior. In the purchases with unknown customer gender, frequencies of gender-neutral products are remarkably high, and there are high frequencies for products who were disproportionally high in both genders. This suggests that the customers with unknown gender actually of both genders and follow a similar distribution of gender as in the purchases with known gender.

Conditional histograms enable detailed attribute-attribute association analysis for a given value of the first attribute, and offer many viewing modes that facilitate the comparison. Also, they can fit a reasonably large number of categories and enable quick browsing and manipulation. However, finding an unknown association requires creating a conditional histogram for each category / value range. Moreover, the created conditional histograms are not linked and cannot be compared intuitively.

The next approach presents a visualization technique for comprehensive association analysis, which is able to highlight some of the existing associations in the data. The user can then create a detailed view of the associations of interest in this visualization, by means of conditional histograms.

5.1.2. Parallel sets

As mentioned in section 2.2.2, parallel sets is a technique to visualize multi-dimensional categorical data which shows how the categories of the variables are connected and helps finding existing associations between them. In this work, the parallel-sets technique is used to visualize attribute-attribute association between categorical attributes in one of the sets $X \in \{E_1, E_2, G\}$.

In case of two variables, the disproportionality of two categories v_1 and v_2 from the variables A_1 and A_2 of the set X denotes the positive or negative association between these categories, and can be computed in terms of the frequency ratio fr as follow:

$$fr_{A_1, A_2, X}(v_1, v_2) = \frac{\frac{freq_{A_1,A_2,X}(v_1, v_2)}{freq_{A_2,X}(v_2)}}{\frac{freq(v_1)}{|X|}} = \frac{freq_{A_1,A_2,X}(v_1, v_2) \cdot |X|}{freq_{A_2,X}(v_2) \cdot freq_{A_1,X}(v_1)}$$

Where $freq_{A,X}(v) = |\{x \in X : A(x) = v\}|$

and $freq_{A_1,A_2,X}(v_1, v_2) = freq_{A_2,A_1,X}(v_2, v_1) = |\{x \in X : A_1(x) = v_1 \wedge A_2(x) = v_2\}|$

$$disp_{A_1, A_2, X}(v_1, v_2) = \begin{cases} fr_{A_1, A_2, X}(v_1, v_2) - 1, & fr_{A_1, A_2, X}(v_1, v_2) < 1 \\ \dfrac{fr_{A_1, A_2, X}(v_1, v_2) - 1}{\dfrac{|X|}{max(freq_{A_1, X}(v_1), freq_{A_2, X}(v_2))}} - 1, & fr_{A_1, A_2, X}(v_1, v_2) \geq 1 \end{cases}$$

The disproportionality $disp$ takes values in the range [-1, 1] (thanks to the normalization of the frequency ratio), with -1 corresponding to the absence of instances in both categories and 1 corresponding to these categories being linked only to each others. It is worth mentioning that $disp_{A_1, A_2, X}(v_1, v_2) = disp_{A_2, A_1, X}(v_2, v_1)$

In this work, to create the parallel-sets view, the user selects the categories of the variables between which she wants to perform association analysis. The set items that fall in these categories are then extracted. A box for each category v in each variable A is created which width $w_{A,X}(v)$ being proportional to $freq_{A,X}(v)$.

In case of two variables, the box for each category v_1 in A_1 is connected with the box for each category v_2 in A_2 by a parallelogram with base length $bl_{A_1,A_2,X}(v_1, v_2)$ being proportional to $freq_{A_1,A_2,X}(v_1, v_2)$. The base lengths of the stripes in each box sum up to the box width (in addition to a spacing factor):

$$w_{A_1,X}(v_1) = spacing_{A_1}(v_1) + \sum_{v_2 \in \, categories \, of \, A_2} bl_{A_1,A_2,X}(v_1, v_2)$$

Fig. 46 shows an example of parallel sets implemented in this work. Ten categories from the "Relation.Product.Category" and all three categories from "Relation.Customer.Gender" have been selected for association analysis and the purchases that fall in these categories have been visualized as stripes and colored either according to the customer gender [Fig. 46-a] or according to the product category [Fig. 46-b].

The disproportionalities between each pair of categories from both variables are computed and visualized by means of vertical line segments originating from the inner side of the respective boxes. The segment length is proportional to the absolute value of the respective disproportionality $|disp_{A_1, A_2, X}(v_1, v_2)|$, and its direction shows if the disproportionality is positive (pointing inwards to the other category) or negative (pointing outwards).

In Fig. 46, one can recognize that female customers are disproportionally highly associated with clothing and jewelry shops, while negatively associated with gas-stations entertainment providers and hotels. Male customers show roughly the inverse pattern. These findings have been also observed in the conditional histograms in the previous section. The advantage of parallel sets is that they show all the associations at once which enables easier comparison between them. Also the view provides additional information such as the global distribution for the selected categories of each variable which is not immediately available in conditional histograms.

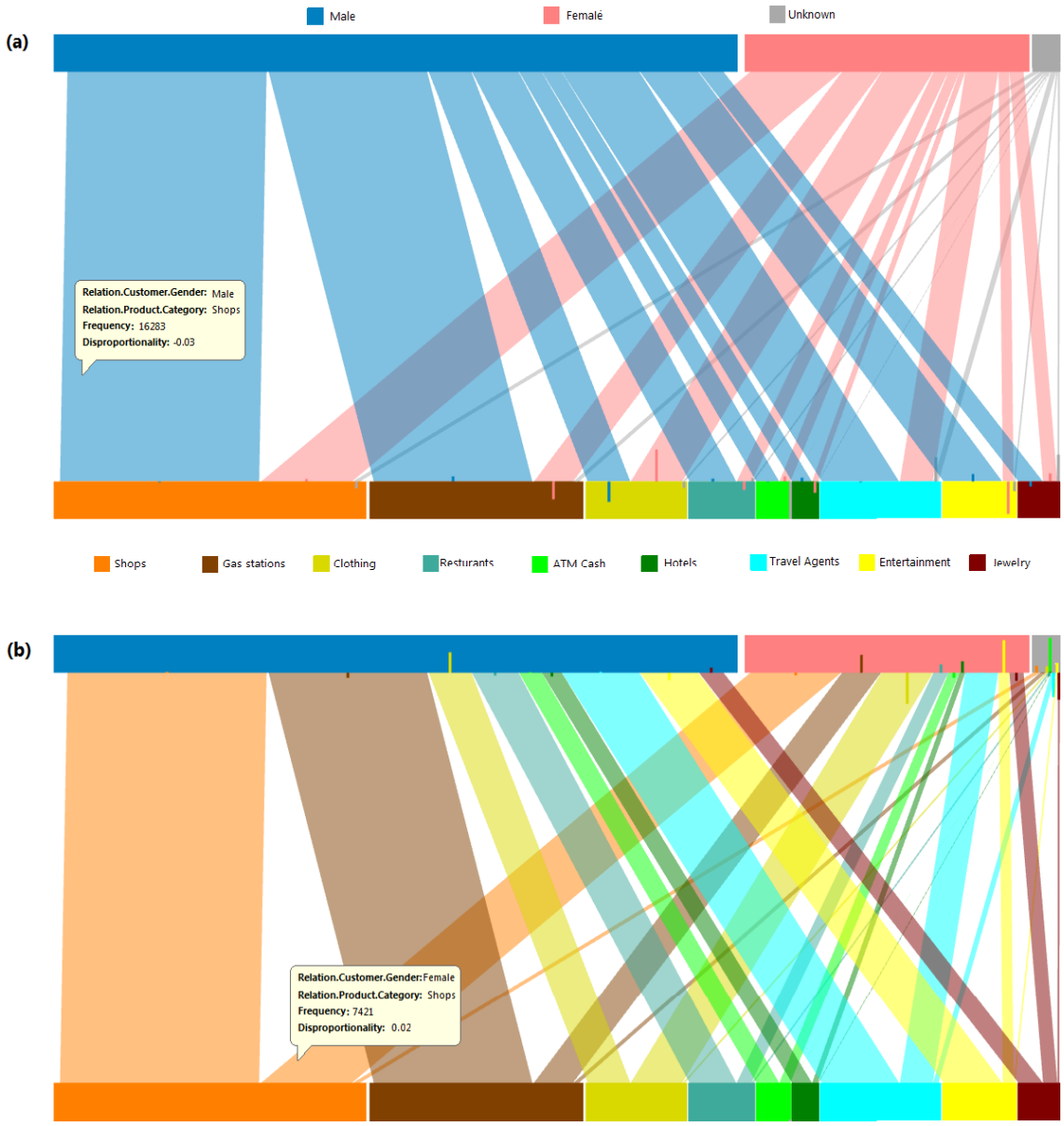

Fig. 46: Parallel sets.

In the next subsections we will examine a perceptual enhancement to parallel sets, and an extension which incorporates more information in case of visualizing relational datasets.

5.1.2.1. Enhancing frequency perception

In the work of Hauser et al. [HKB06], the stripe which connects the categories v_1 and v_2 from the variables A_1 and A_2 is drawn as a parallelogram which length is proportional to $freq_{A_1,A_2,X}(v_1, v_2)$ as has been explained in the previous section. The areas of all these stripes sum-up to the area of the rectangular space between the boxes of the two variables (assuming no spacing between the stripes) denoted by $totalarea$ (assuming no spacing between the stripes, i.e. $spacing_A(v) = 0$).

This property makes the area of a stripe a visual representative for the proportion of the items that fall in it:

$$\frac{area_{A_1,A_2,X}(v_1, v_2)}{totalarea} = \frac{freq_{A_1,A_2,X}(v_1, v_2)}{|X|}$$

However, to compare the frequencies of two stripes, visually comparing their areas is not particularly intuitive since the stripes can have arbitrary slant. Moreover, visually comparing the stripe thicknesses might be misleading for the same reason. This is due to the fact that the thickness of the stripe is perceived perpendicular to its direction. The correct way to compare frequencies would be to compare the base lengths of the two stripes, which is, however, not particularly intuitive either.

For example, in Fig. 46-a, the female customers did 3194 purchases in gas stations and only 2485 purchases at travel agents. Nevertheless, the stripes which connect them have almost the same perceived thickness (in fact the perceived thickness of the travel agent stripe is a little bit larger). Also, the purchases of female customers in the shop categories are about 46% of those of male customers; nevertheless the stripe for the first purchases has about 28% of the thickness of the other stripe.

To compensate for this effect, we adapt the visual representation for a stripe to achieve a constant thickness perception along it. The thickness is defined as the distance between the stripe sides and is made to be equal to the base length of the original parallelogram. This is achieved by drawing the stripe as the inner area between two offsets of a C^2-continuous curve which connects the boxes of the two categories the stripe connects. This curve meets the parallel axes orthogonally.

We implemented the curve by two arcs connected by a line segment. An offset of this curve is created on each side which meets the range end at this side (also orthogonally). The area between the two offsets and the stripe ranges in both axes defines the visual representation for the stripe. A more precise mathematical formulation of how the stripe is visualized is presented in Appendix I.

Fig. 47-b shows the adjusted representation of stripes of Fig. 47-a. With the introduced enhancement, the perceived stripe width (thickness) remains constant along each stripe, and consistent among different stripes.

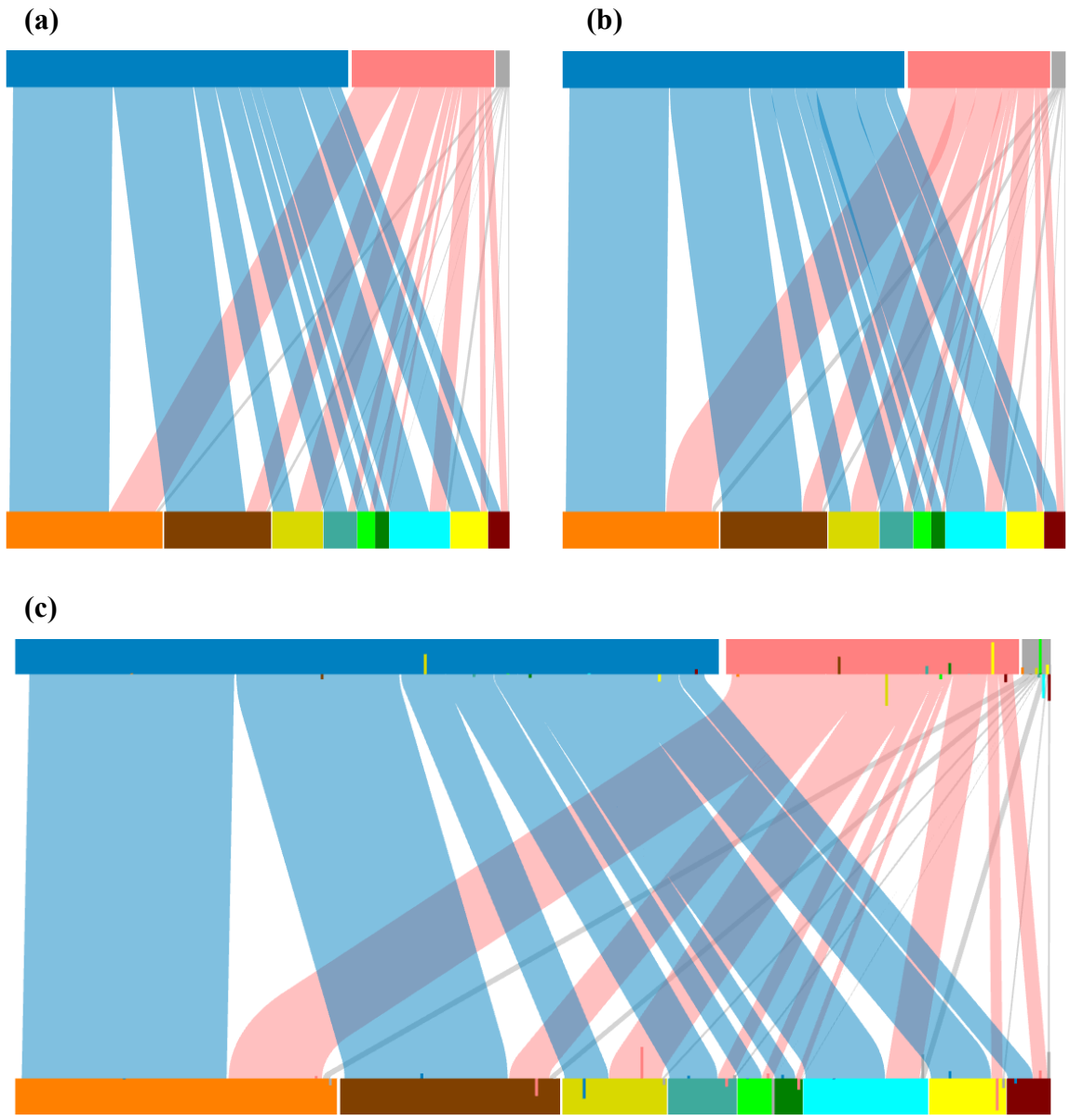

Fig. 47: Enhancing frequency perception in parallel sets. (a) The original technique which uses parallelograms (b) using curvilinear stripes (c) removing overlapping artifacts and stretching.

In fact, the above adjustment increases the horizontal width non-linearly along the stripes. This increase might result in adjacent stripes originating from the same box to overlap each other. This overlapping can be reduced either by increasing the spacing between the stripes. Alternatively, the overlapping can be retained, but the resulting visual artifacts are removed by changing the color composition to ignore drawing a color in pixels having the same color already [Fig. 47-c]. In this figure, the view has also been stretched from Fig. 47-b to the same scale as Fig. 46-a. The thickness adjustment was not updated to account for the stretching (and hence the thickness is not kept constant along stripes). Nevertheless, the resulting stripes remain C^2-continuous, with circular arcs being replaced by elliptical arcs.

5.1.2.2. Parallel sets for relational data

Apart from providing variables for association analysis, the set of relations GG in the relational data $R = (E_1, E_2, G \subset E_1 \times E_2)$ contains more information that can be incorporated in the visualization of parallel sets, and enrich the understanding of the associations. The additional information is derived from the related entity sets E_1, E_2 and is intended to show how the computed associations between relation variables are related to the entities in both sets. Without loss of generality, we restrict the analysis to the case of two variables.

The first extension we implemented maps the frequency of each view category in the corresponding entity set to the height of its box, while the frequency of this category in the relation is mapped to the box width:

$$w_{A,G}(v) \sim freq_{A,G}(v)$$
$$h_{A,E}(v) \sim freq_{A,E}(v)$$

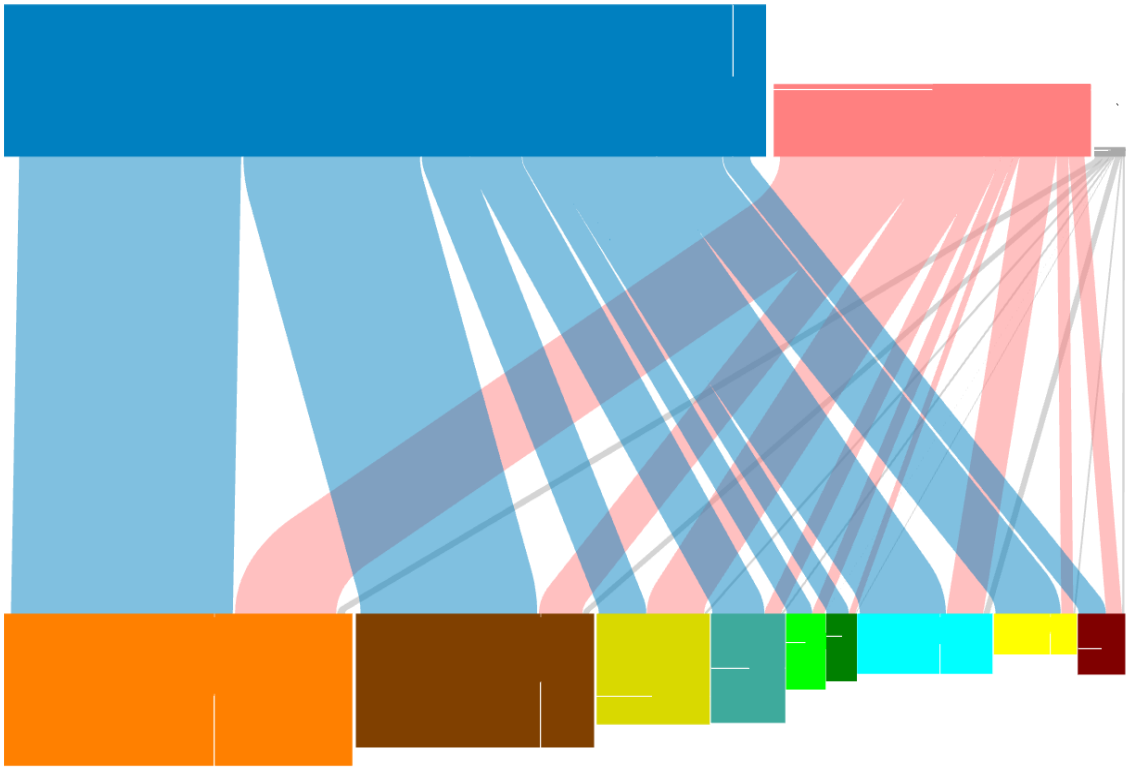

Fig. 48: Incorporating entity information in parallel sets of relational data.

This helps seeing how the entities are distributed in the categories of each variable [Fig. 48]. Additionally, the resulting boxes have different aspect ratios, which helps finding which category has higher a number of relations per-entity (computed as $\frac{freq_{A,G}(v)}{freq_{A,E}(v)}$).

Furthermore, to compare the number of relations per entity in each category and the average number of relations per entity (computed as $\frac{|G|}{|E|}$), a line can be shown in each box which indicates this average ratio as follows:

- In case the box has a higher than average relations/entities ratio, a vertical line shows the number of relations which would have made the ratio equal to the average
- Otherwise, a horizontal line shows the number of entities which would have made the ratio equal to the average.

Besides seeing the frequencies of categories in entity sets, it is also interesting to see how the relations represented by a stripe are distributed between the entities in the connected categories. We perform this by showing the graph of the function between the stripe relations and the entities, which maps each relation to its corresponding entity. The relations are ordered so that the function is non-decreasing. The function is visualized by rendering its graph with sub-pixel resolution to a $Freq_{A_1,A_2,X}(v_1, v_2) \times Freq_{A,E}(v)$ canvas. Then each pixel is assigned an opacity based on the number of sub-pixels in it. The canvas is drawn with a white background in the corresponding location in the box.

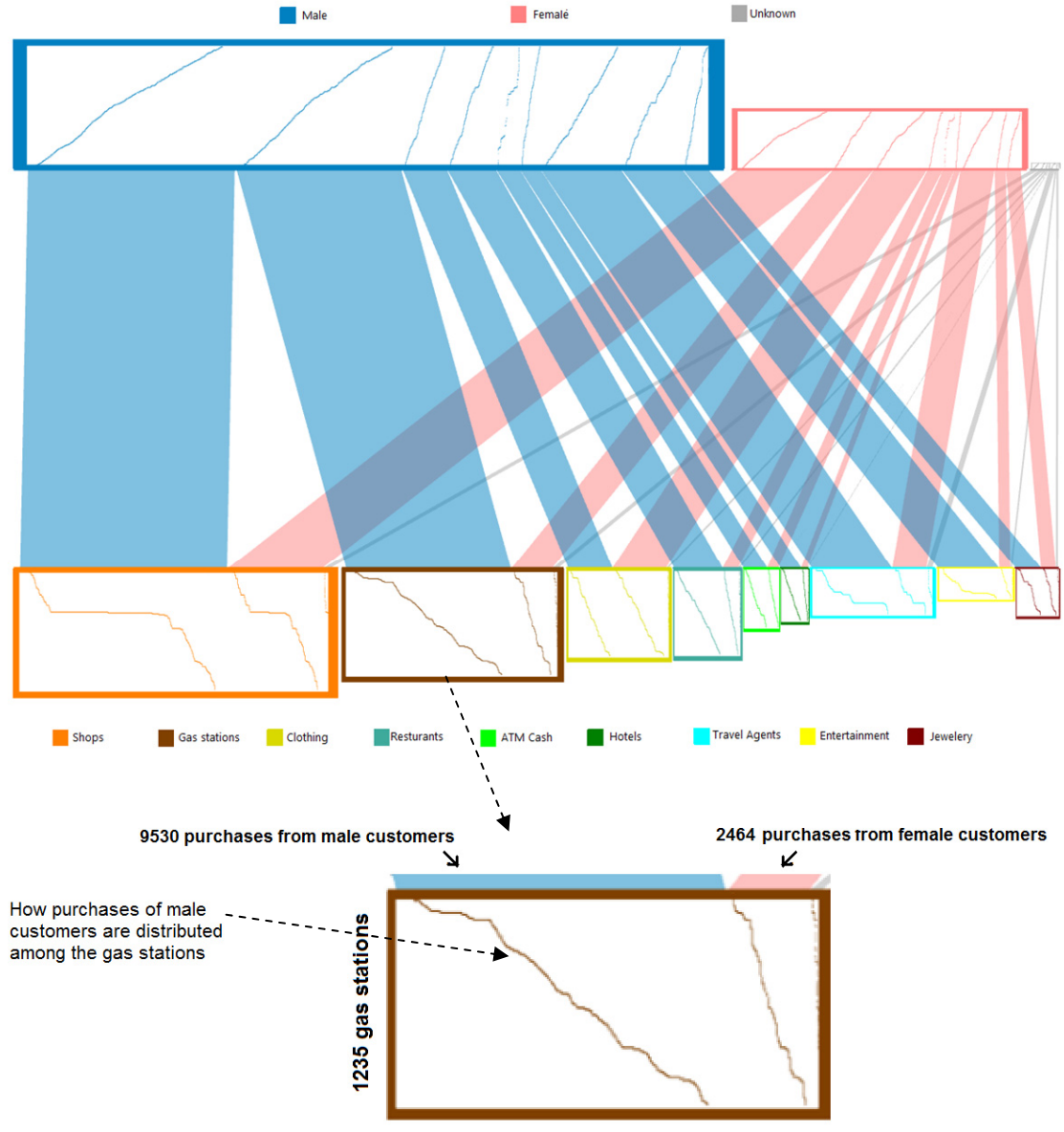

Fig. 49: Visualizing the relation-to-entity mapping in parallel sets.

Fig. 49 shows an example for visualizing the relation-to-entity mapping. Fig. 50 shows three interesting cases spotted in Fig. 49. In Fig. 50-a one can recognize that certain entities (20 out of 1579) in the shops category have received a very large part of the relations (48% of purchases from male customers and 37% of purchases from female customers). The same observation can be made in purchases in travel agents and entertainment categories in Fig. 49. In contrast, the purchases in restaurants where almost evenly distributed (Fig. 50-b). Also, one can notice that remarkably many customers did not use ATM Machines (Fig. 50-c).

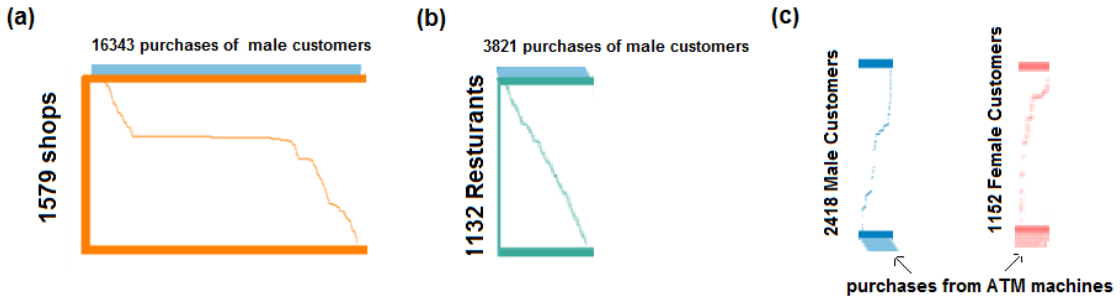

Fig. 50: Interpreting the relation-to-entity mapping. (a) A curve with varying slope: non-uniform distribution of the relations among the entities. (b) A curve closer to a straight line: the distribution is more uniform. (c) Discontinuity in the curve: several entities do not receive relations.

An attribute from the entity type of the box category can be mapped to the color of the sub-pixels of the graphs of the relation-entity mappings in it. In this case, the entities in each box (distributed in the height) are ordered by this attribute so that equal values are placed next to each other. The corresponding relations are reordered to preserve the monotonicity. This enables seeing how the stripe relations are distributed according to the selected attribute.

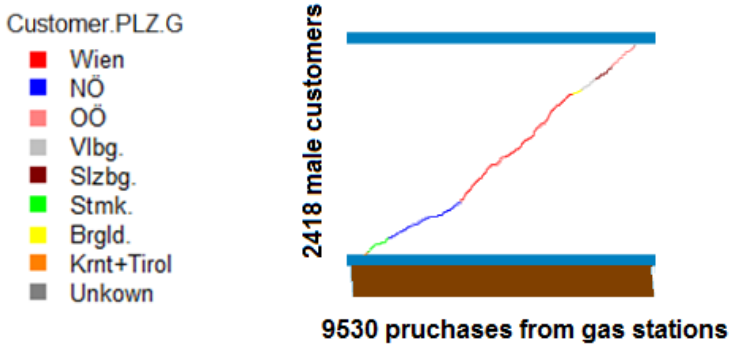

Fig. 51: Coloring the relation-to-entity mapping graph by a user-defined entity attribute.

In Fig. 51 the attribute “Customer.PLZ.G” which represents the provinces in Austria where customers live, is mapped to the graph color in the customer categories. One can recognize that male customer from the province NÖ* have higher per-customer purchases from gas stations than other customers (since the curve had low slope for these purchases). The mapping of the relations to the different categories can be better emphasized by a space-filling visualization as explained in section 7.3.1.3.

Lower Austria

5.2. Attribute-Relation Association

In the relations set $G \subseteq E_1 \times E_2$, we would like to know if the distribution $dist_S(A)$ of a relation attribute A, derived from one of the entity sets E_1, in the subset S defined as:

$$G \supset S = \{x : ID_{E_2}(x) = id\}$$

deviates considerably from its global distribution $dist_{E_1}(A)$ in this entity set. The set S represents relations in G to a specific entity of the other entity set E_2, whose identifier $ID_{E_2}(x)$ (defined in section 1.2) is equal to id.

As mentioned earlier in the chapter overview, the local analysis of this type of associations is applied to a specific entity of the set E_2, which is selected by the user (e.g. by specifying its id) while the global analysis is applied to all entities in this set. In this work, the local analysis can be performed by means of a conditional histogram and / or graph exploration views, while the global analysis can be performed by means of aggregated attributes or a view we specifically designed for this purpose called the "entity wheel".

5.2.1. Local analysis

Conditional histograms are suitable to examine the distribution of an attribute in the multiset of entities related to a specific entity, in the same way as they were employed to perform local attribute-attribute association analysis. Instead of specifying ranges for other attributes, the user only needs to select the desired entity to analyze its related entities. In fact, since the identifier of the entity is an attribute itself, selecting the entity can be done in the same way as a range is selected. In Fig. 52, a histogram of the product identifier in the relation is created, where besides each product the number of its purchases is written. A histogram of customer year of birth in the purchases of one of these products is created and visualized both in absolute, relative to the global distribution, and per-entity modes.

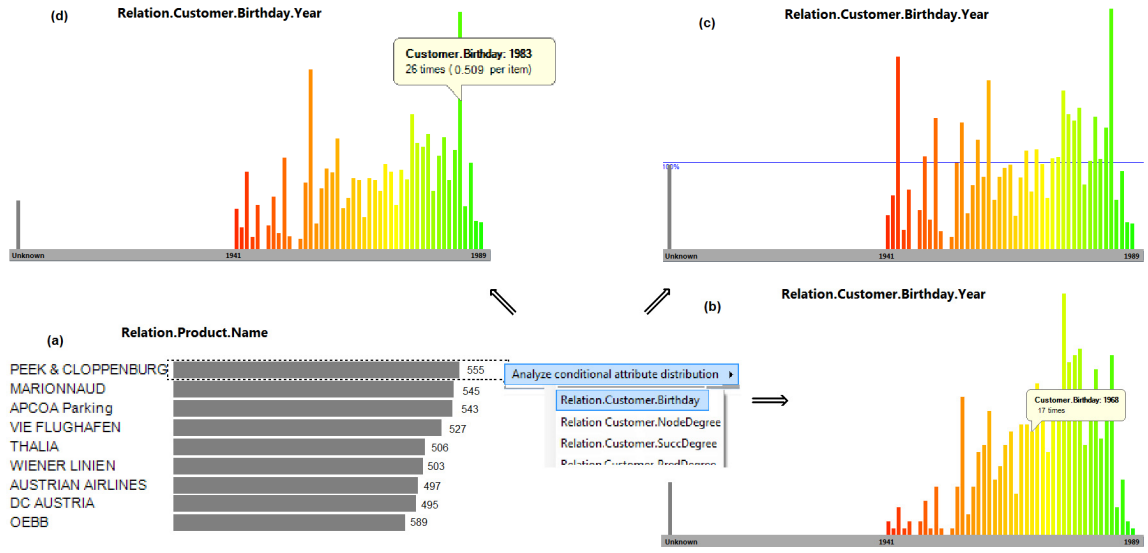

Fig. 52: Histogram for local attribute-relation association analysis. (a) Choosing a product. (b) Birthday distribution of customer birthday in its purchases (c) Relative distribution (d) Per-entity distribution.

In the relative and per-entity histograms [Fig. 52-c and Fig. 52-d], one can notice that young customers have higher per-capita purchases from the selected shop, compared to older ones. To assert this fact and make sure the increase is not due to outliers (few young customers with very high purchase rates), a graph exploration view is created with the selected product as the central entity. The customers are sorted by birthday and placed in the first ring, and the purchases for each customer are also visualized as dark green triangles [Fig. 53]. The purchases show roughly the same density as depicted in the histograms with the additional information of showing how the purchases are distributed among single customers (which is compromised in the histograms).

As can be seen in Fig. 53, the purchases among the customers is rather more uniform among younger customers which means that the corresponding empirical histogram densities are closer to the true unknown densities.

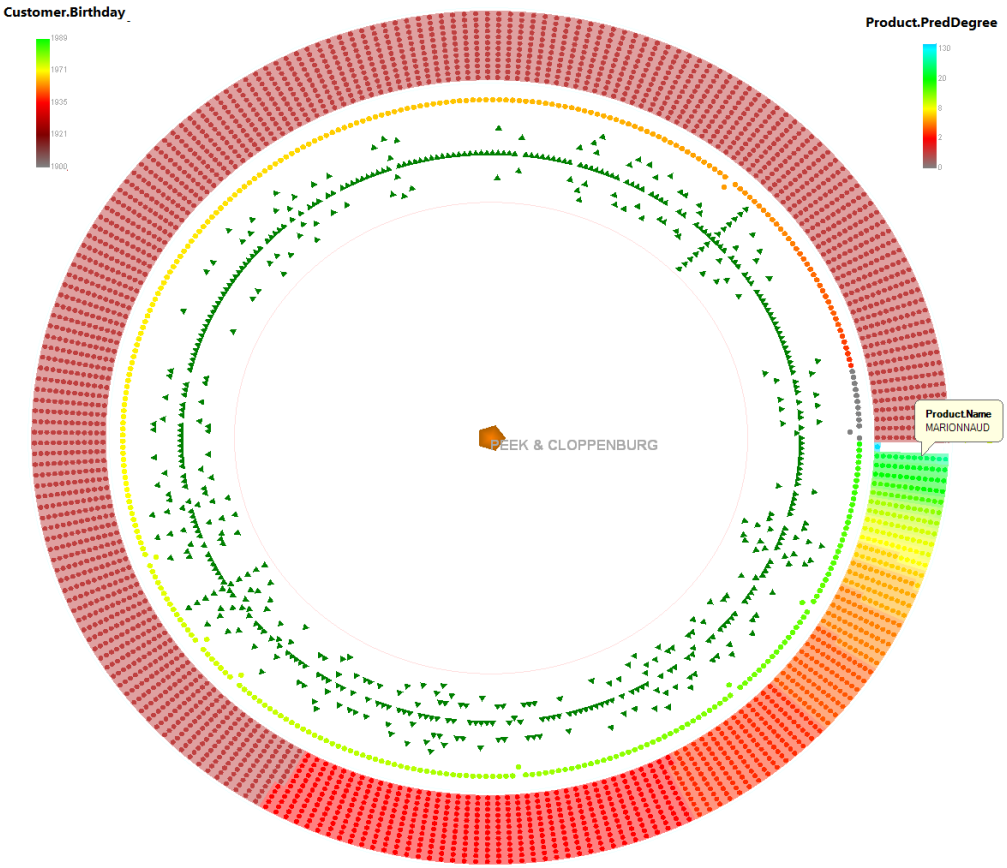

Fig. 53: Detailed relations exploration.

Using graph exploration views to view the relations is not only useful for verifying the histogram densities, but also enables applying all the features mentioned in chapter 4 to the visualized subgraph. This includes viewing related entities in farther-away levels, how common they are related, what attribute they have, selection, sorting, viewing links, and other interaction functionalities in these views.

5.2.2. Aggregated attributes

An aggregated attribute, defined in section 3.2.4, associates with each entity an aggregation of the values of an atomic (non-aggregated) attribute in the related entities. We assume the atomic attribute is categorical. In this case, the aggregation of the values using the mode value and the corresponding distinctivity of this value (both defined in section 3.2.4) help finding for each entity if there is a dominant category in entities related to it.

The analysis is performed by first selecting a subset of entities whose relations are to be analyzed and then creating a nodes-only navigation view of this subset. This view shows the selected node as the result of a query. In this view, the desired categorical, aggregated attribute is mapped to the nodes' visual properties, by mapping the color of the mode category in each entity to its node color and the certainty value to the node saturation. Additionally, the nodes are split into groups according to the resulting categories, and in each group, the nodes with high certainty are placed outwards. In Fig. 54 the customers who bought from the bookshop "Thalia" are colored according to their mostly purchased product category. The customer nodes near the center have small certainties for their dominant category.

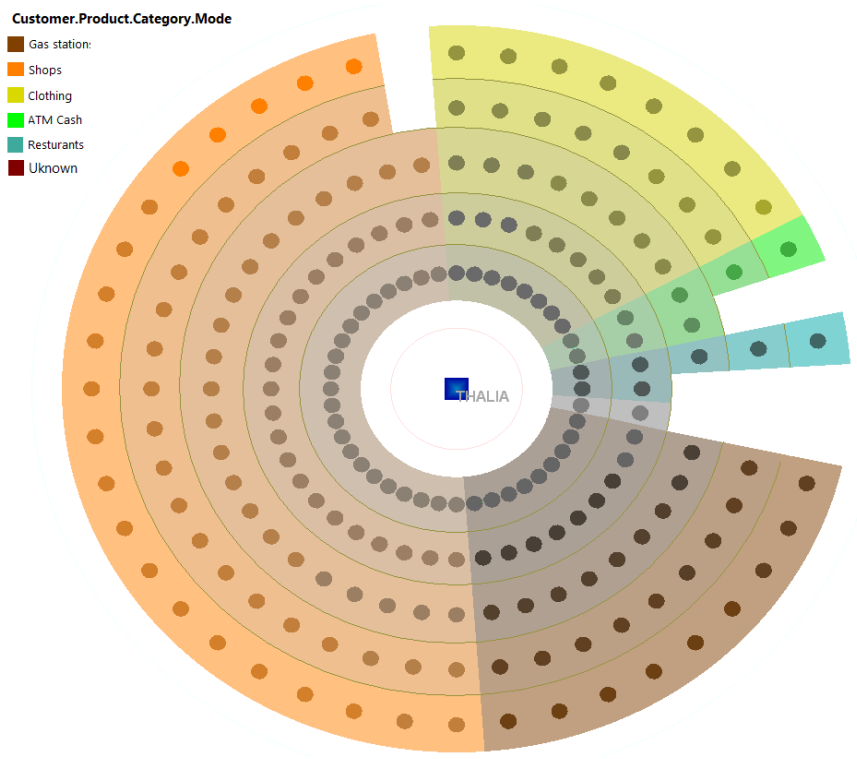

Fig. 54: Aggregated attributes for attribute-relation association analysis.

This method does not handle the case of more than one dominant category (which all have high frequencies in comparison with the other categories). A more elaborate view for attribute-relation association, with dedicated layout and interaction functionalities, is presented in the next section.

5.2.3. Entity wheel

This view is dedicated to find attribute-relation association for a categorical attribute. It is intended to show for each entity, how it is related to the categories of this attribute. In the relation set $G \subseteq E_1 \times E_2$, a conditional histogram showing the frequencies $freq_{S(e_j)}(A)$ of the categorical attribute "$E_1.A$" of the entity sets E_1 is associated with each entity $e_j \in E_2$ from the related entity set, where:

$$G \supset S(e_j) = \{x : ID_{E_2}(x) = id_j\}$$

It is not straightforward to produce a readable visualization of all the values in all of these histograms at once. Luckily, in the context of entity-relation association, not all of these values need to be incorporated in the visualization, but rather the ones that exhibit a significant association between an entity and a category k which is present frequently enough among its related entities. The strength of this association can be defined by means of the distinctivity of k in the values of "$E_1.A$" in $S(e_j)$, which has been defined in section 3.2.4. A threshold α determines the significance of this association:

$$distinctivity_{S(e_j)}(k) \geq \alpha : 0 < \alpha \leq 1 \ .$$

Likewise, the normalized distinctivity can be used instead [section 3.2.4].

The visualization is performed in this view by assigning a sector in a ring for each of the categories of the attribute "$E_1.A$". Entities from E_2 are placed in these sectors by means of the geometric mapping, and assigned visual properties by means of the visual mapping as will be explained in the next sections.

5.2.3.1. The geometric mapping of entities

Initially, for each entity selected for visualization from E_2, a graphical node is placed in the sector of its mode category "$E_2.E_1.A.Mode$". The radial location of the entity in the sector depends on the distinctivity of its mode category: nodes with distinctivity = 1 (all related nodes fall in the sector category) are placed on the outer boundary, and nodes with distinctivity = 0 (have equal frequency in all the categories) are placed on the inner boundary. The nodes with other distinctivity values are placed on rings which radii are interpolated linearly between the two rings. The distinctivity is also mapped linearly to the node size.

The nodes in a sector are spread angularly to prevent overlapping without changing their radial positions. This is achieved by drawing the nodes in descending order of their distinctivities (from outwards to inwards). When drawing a node, the next free space for its disk is searched which lies on the ring that corresponds to its distinctivity and which is as close to the median of the sector as possible. After finishing the placement of all nodes in all sectors, the nodes are scaled down to re-fit nodes possibly placed outside their sectors (due to unavailable free space inside the sector) back into their sectors. The algorithm for spreading the nodes is presented in Appendix II.

Fig. 55 shows an example entity wheel. A sector is created for each category in the customer attribute "Customer.PLZ.G". These categories represent the provinces in Austria where the customers live. Each product is placed in the sector where customers did more per-capita purchases from this product than customers from other sectors (which corresponds to the normalized mode category "Product.Customer.PLZ. G.NMode"). Only products with large enough support (NodeDegree > 20) are shown.

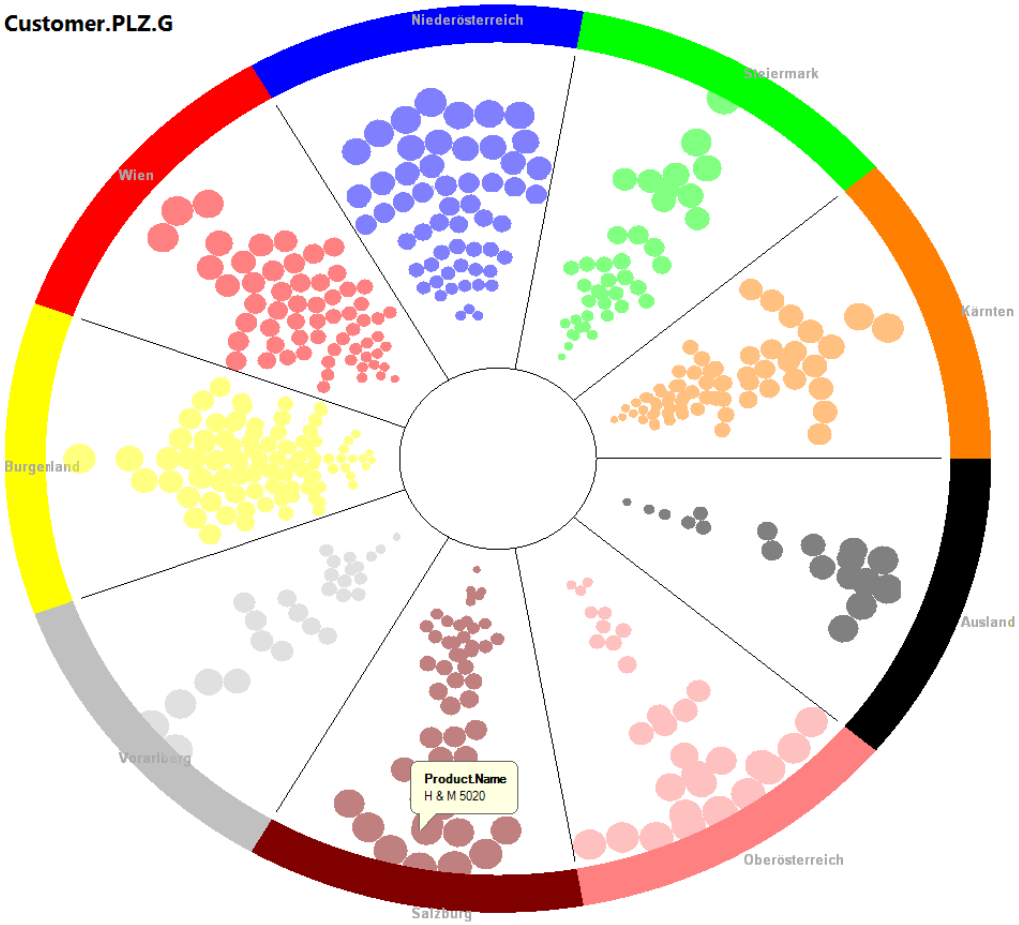

Fig. 55: An entity wheel showing distribution of products according to customer provinces.

In this view it is easy to see which products have particularly high association with one of the categories: these products have larger nodes and are close to the sectors outer boundary. Also it is easy to identify products that have a low distinctivity factor and hence have histograms close to a constant at the average ($nfreq_{S(e_j)}(k) \simeq |S(e_j)|/|categories|$).

Until now, only the association of the entity with its mode category has been visualized. Still, there can be other significant associations between the entity and other categories (with $discinctivity_{S(e_j)}(k) \geq \alpha$). However these additional associations might exist only in nodes with mode distinctivity lying in the range $[\alpha, 1-\alpha]$: $\alpha \leq 0.5$. For example, when $\alpha > 0.5$, at most one category (which is mode category) can satisfy the distinctivity condition. This means that the user can control the number of additional associations (other than with the mode category) to incorporate in the visualization, by controlling the value of α.

Incorporating these additional associations in the visualization resembles the multi set membership problem which was also encountered in graph drawing with links: a level-2 node has connection with multiple level-1 nodes. The same solution as the one used in the graph drawing problem can be applied to the entity wheel, namely, the node-repetition-and-linking technique introduced in section 4.3.2.

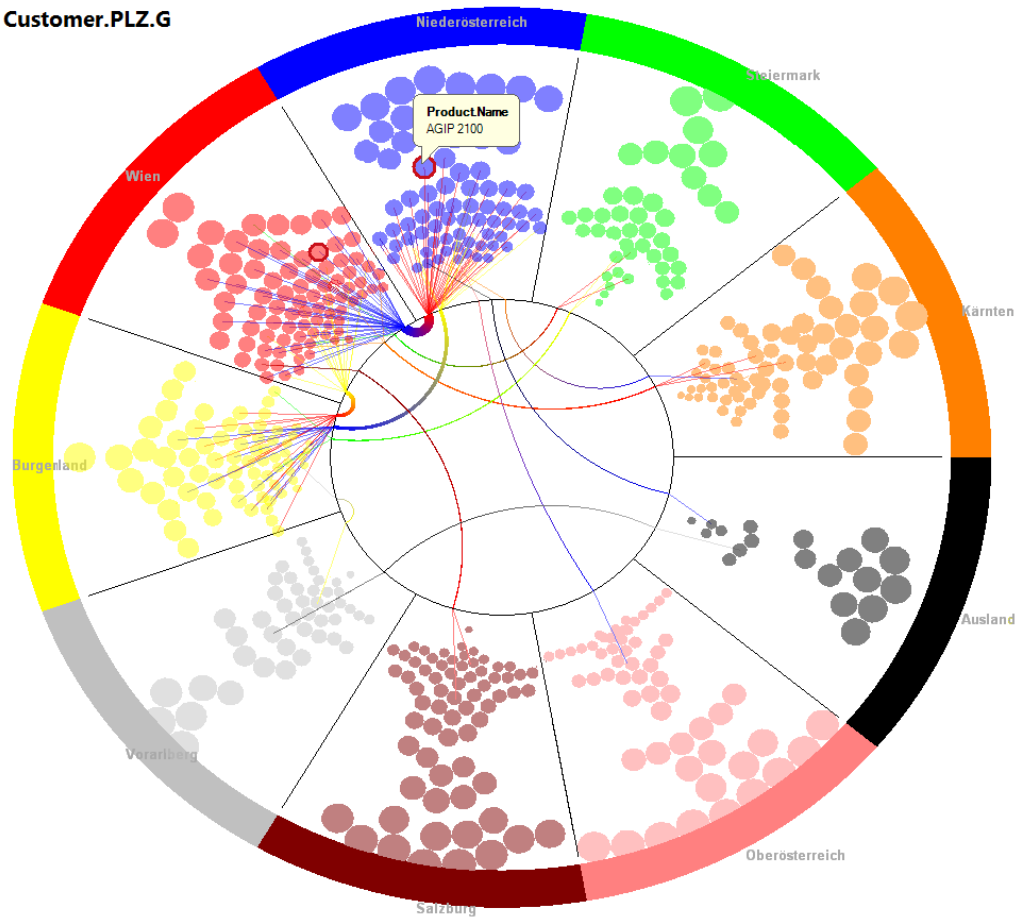

Fig. 56: Node-repetition-and-linking in the entity wheel.

Using this technique, for the entity which has significant associations with multiple categories (including the mode category), a separate graphical node instance is created in each sector of these categories. This node is layouted in the same way as the other nodes in the sector: the radial placement and node size are based on the distinctivity of the sector category in the related entities of this entity.

All the node instances of the same entity are linked using the bundling-by-arc technique explained in section 4.3.2.1, as depicted in Fig. 56. In this figure, one can notice that many products achieve high distinctivity in two Austrian provinces, which often happen to share a geographical border such as Wien-NÖ, NÖ-Burgenland, Vorarlberg-Ausland and Burgenland-Ausland (where the group “Ausland” contains customers mostly from Germany and Hungary). Many of these products represent regional branches of shops or service providers. In fact, visualizing the links not only helps recognizing the nodes that have multiple distinctive categories, but also reveals which two categories are highly interrelated.

5.2.3.2. The visual mapping of nodes and sectors

To distinguish sectors from each others in the entity wheel, a stripe on the outer boundary of the sector is assigned the color of the sector category and the category names can be placed next to the sectors. The area of the stripe can be chosen to code the frequency of the entities in the sector category [Fig. 57-a].

A link between two instances of the same entities is assigned color from the two sectors it connects in the same manner the colors are assigned to links in section 4.3.2. The nodes colors are mapped from a user defined attribute from their entity type.

To enhance readability of the resulting visualization, the nodes can be sorted according to the chosen attribute and placed so that similar values lie close to each other, in a similar manner as in the graph exploration view in section 4.2.1. However, in the placement of sorted nodes, only the angular position of the nodes in the sector can be used to indicate the order or grouping, since the radial position should code the distinctivity.

In case the nodes are sorted by a numerical attribute, a simple approach to place them is to draw the nodes in the same manner explained in the previous section, but using the user-defined sorting attribute for determining the node drawing order instead of the distinctivity attribute. In case the sorting is descending, this method results in placing nodes with high values for the sorting attribute in the middle of the sector, and the further a node is placed from the center, the lower is the attribute value. Fig. 57-b shows “Product” nodes sorted by their “NodeDegree” attribute in a descending order. The NodeDegree is also mapped to the node color.

In case the nodes are sorted by a categorical attribute, a simple approach to place them is to order the categories of each sector descendingly by their frequencies in the sector, and draw their nodes in this order. When looking for a drawing space for a node along the ring of its radial distance, instead of searching the location as close as possible to the sector median, the location is searched as close to the sector right boundary as possible. The intention of this placement is to place the nodes of the same category closer to each other.

Fig. 57-a shows product nodes sorted and colored by their categories. One can easily recognize that the majority of products which are bought by female customers more often than by male customers are clothing shops, while the majority of products which are bought from male customers more often, are services of gas stations. In contrast to parallel sets, that treat the category as the item of analysis, the wheel can show gender-bias for single entities which enabled spotting the male clothing shops (which would be impossible in parallel sets).

The resulting view bears similarity with a stacked-histogram. It might however, place the nodes of the same category in visually unconnected regions. An alternative approach to place nodes sorted by a categorical attribute would be to divide the sector into subsectors which correspond to the categories of the sorting attribute. This solves the visual discontinuity drawback, but is, however, not as space efficient as the stacking approach and might results in smaller nodes (due to scaling nodes down to fit them in their sectors) with large unoccupied areas.

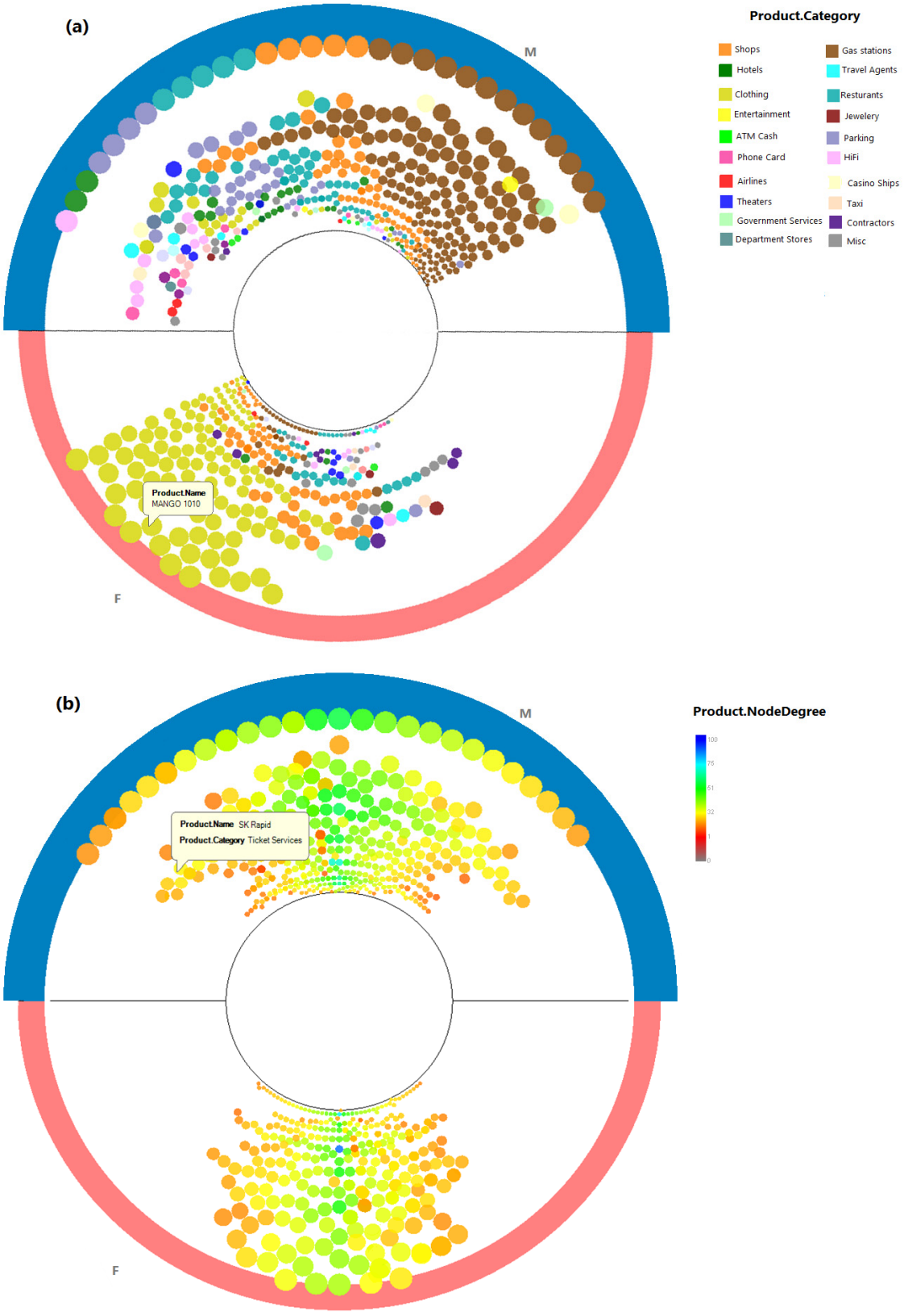

Fig. 57: Assigning node color and altering node location according to an attribute.

5.2.3.3. User interaction

The user can choose the attribute which will be mapped to the wheel sectors, as well as the attributes for sorting and coloring the wheel nodes, and define the minimum support for the nodes to be included in the view and the minimum distinctivity α for additional instances of the nodes. Also, the user decides whether to show links between multiple instances, the coloring scheme of these links and the blending degree based on the link multiplicity (same as in section 4.2.2).

Any node in the wheel can be clicked to see how it is associated to the categories of the other sectors. The association between the node entity and the sector category is defined by the distinctivity of the category in the selected entity relation (defined in section 3.2.4). These associations are visualized using the star coordinate system (proposed by Kandogan [K00]). The axes of this system are the sector medians, and the association of the selected entity in a category is piecewise-linearly mapped to the category axis based on the distinctivity as follows:

- $distinctivity \geq 0$: The coordinate is located inside the sector between the boundaries, with $distinctivity = 0$ mapped to the inner circle and $distinctivity = 1$ mapped to the outer circle.
- $distinctivity < 0$: The coordinate is located inside the inner circle with the smallest distinctivity $\frac{1}{1-|categories|}$ mapped to the wheel center and $distinctivity = 0$ mapped on the inner circle.

In Fig. 58-a, the sectors represent the customer age groups, and the products are colored by dominant gender of customers who bought them. A product entity in the sector of the age group “38-45” is selected and its associations are visualized. One can notice that it has a positive association in the group “32-37” and totally negative association (which means no relations) with the groups “54-62”, “63-70” and “70+”.

By double-clicking the node, it will be selected as the root node in the graph exploration view. In this view, the user can verify the association between the node entity and the category of its sector, and perform further analysis using the implemented view functionalities. Fig. 58-b shows the graph exploration view of the selected entity in Fig. 58-a.

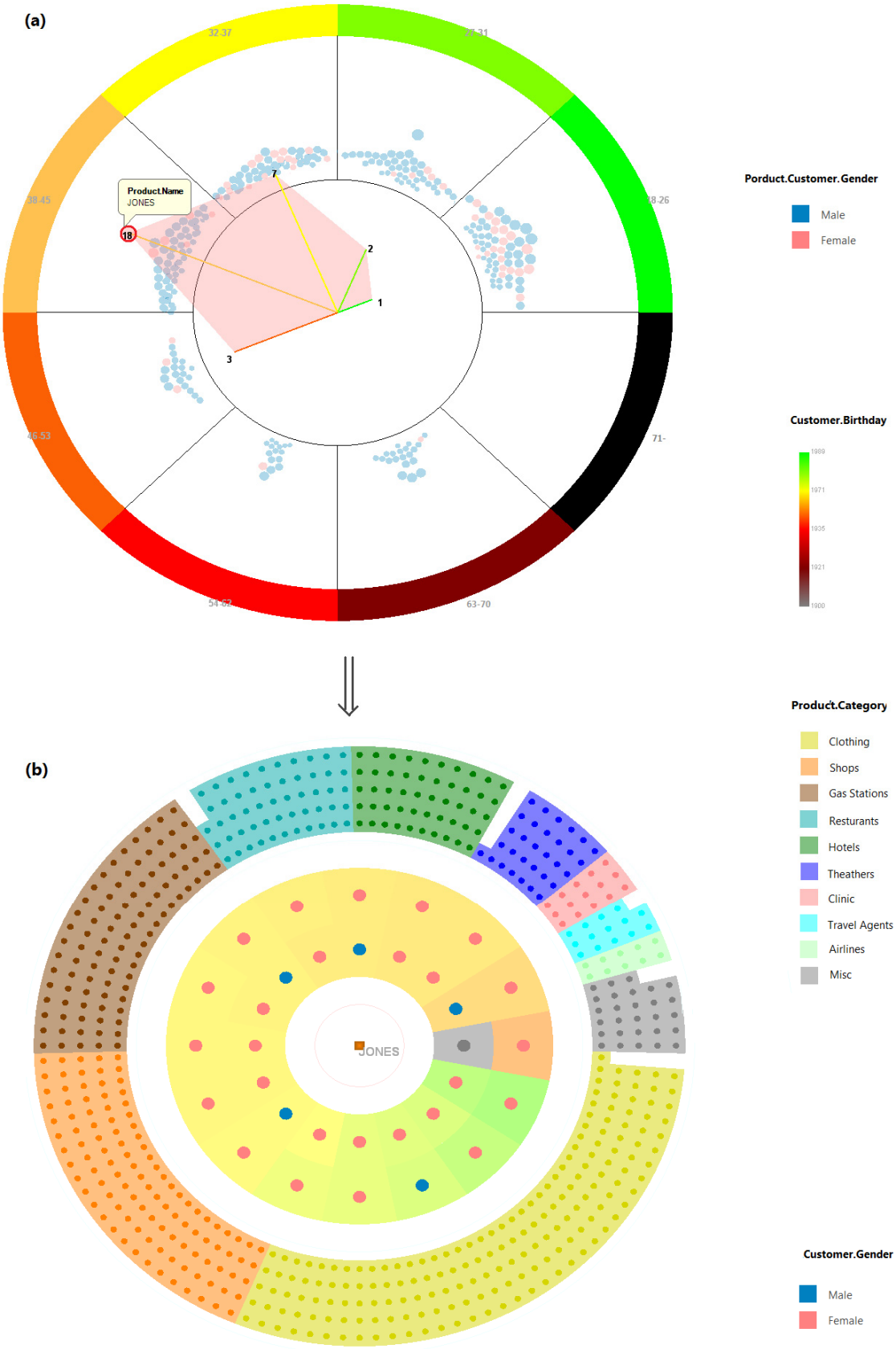

Fig. 58: Interaction with the entity wheel: (a) Viewing the associations of the selected node. (b) Exploring the selected node in the graph exploration view.

5.3. Relation-Relation Association

The last type of association analysis addressed in the work is to find whether the entities related to one entity are often related to another entity as well. For example for a set of products, we want to see if customers who bought a specific product, have frequently bought some products and rarely bought other ones.

No dedicated view has been implemented for this type of association. Instead, the graph exploration view, along with the attribute analysis view can be used to analyze existing relation-relation associations.

5.3.1. Local analysis

The level-2 nodes in the graph exploration view represent the entities that are related with a number of entities that are related to the central entity. This number is denoted by the layout attribute “Entity.CPredDegree”.

To figure out which entities share a lot of relations with a given entity, a nodes-only graph exploration view is created with this entity at the center, and the level-2 nodes are sorted and colored by the attribute “Entity.CPredDegree”.

Fig. 33 shows the customers who bought many of the products bought by a given customer. These customers are placed close to each other and are assigned green to blue colors according to the depicted color scale.

5.3.2. Overall analysis

In a given set of entities, to find which two entities share a lot of relations, these entities can be visualized as query results in a graph exploration view which uses node-repetition-and-linking for showing level-2 nodes. This view visualizes an edge between each two entities in the query results based on the number of relations they share. The thickness and opacity of the edge can be chosen to code this number, which helps visually finding high relation-relation associations in the desired set of entities.

Fig. 59 shows an example where 34 products have been selected as query results and visualized using the graph exploration view with node-repetition-and-linking. Inter sector links are drawn as separate arcs in Fig. 59-a, and bundled as B-spline curves using the product category as the hierarchy for edge bundling in Fig. 59-b.

The above view can fit only a relatively small number of level-1 nodes, while producing a clear visualization. Hierarchical clustering is required to visualize associations in larger graphs.

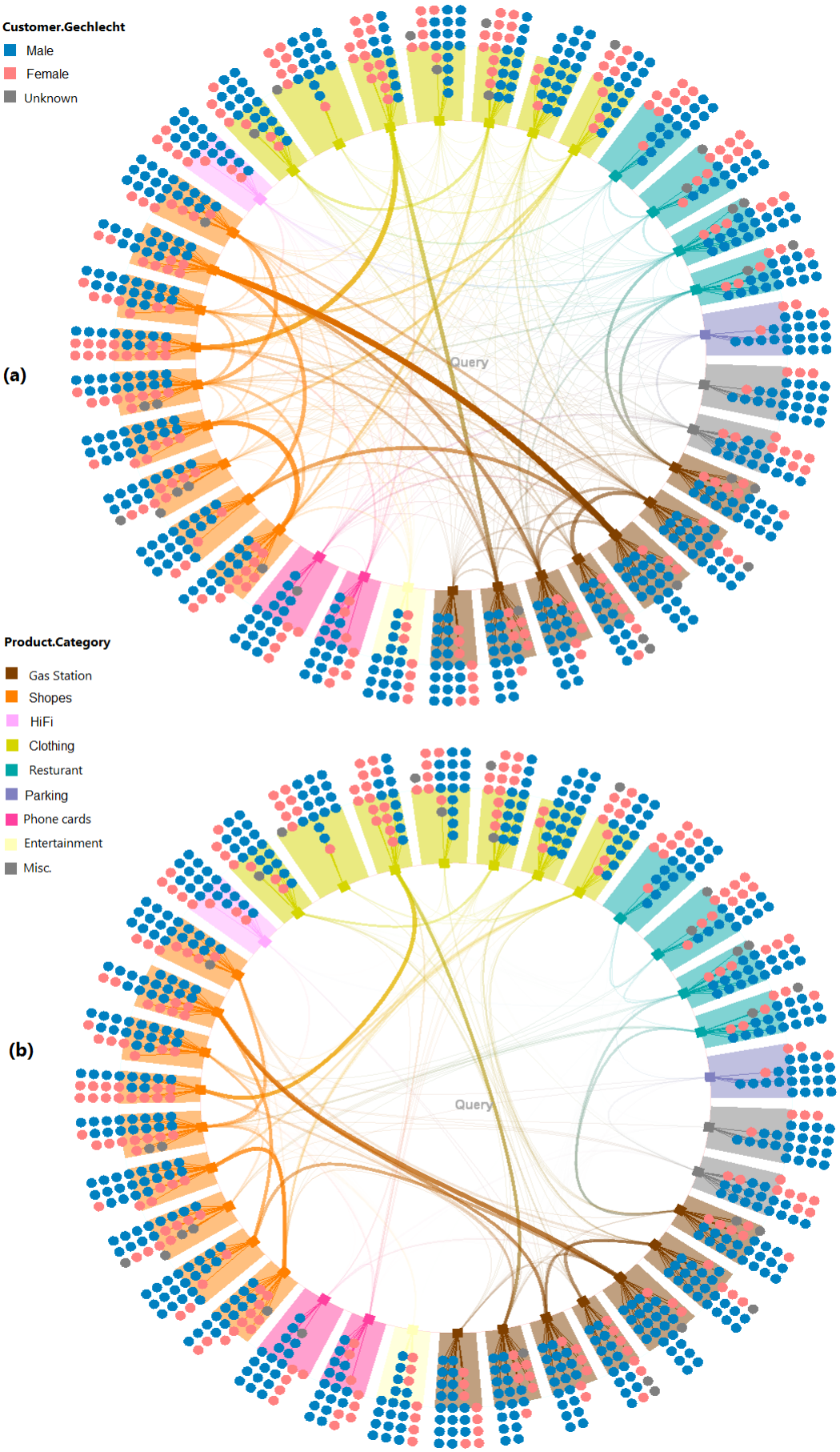

Fig. 59: Relation-relation association analysis.

5.3.3. Attribute-based relation-relation association analysis

This type of association analysis aims to find if entities that are related to entities which have a specific attribute value, are also frequently related to entities which have another value for this attribute. For example we want to see if customers who bought from a specific product category (e.g. parking coupons), have frequently bought from some other categories (e.g. gas stations).

The entity wheel, proposed in section 5.2.3, can give insight in this type of associations with the following parameters:

- Node degree threshold = 0: this allows all nodes to be incorporated in the view, regardless to their degrees (support).
- Distinctivity threshold = $-\infty$: this allows all entities that have a relation with the category of a sector to have a node instance in this sector.
- Adjust the thickness and opacity of an inter-sector link based on the number of nodes common between the two sectors that represent the two categories v_1 and v_2 and on the number of nodes in both sectors, as follows:

$$thickness_{E_i,A}(v_1, v_2) \sim \frac{\left| linked_{E_i,A}(v_1) \cap linked_{E_i,A}(v_2) \right|}{\max\left(\left| linked_{E_i,A}(v_1) \right|, \left| linked_{E_i,A}(v_2) \right| \right)}$$

Where:

$$linked_{E_1,A}(v) = \{e \in E_2 : (\exists e_1 \in E_1 : (e_1, e) \in G \wedge A(e_1) = v)\}$$

$$linked_{E_2,A}(v) = \{e \in E_1 : (\exists e_2 \in E_2 : (e, e_2) \in G \wedge A(e_2) = v)\}$$

The link thickness represents in this case how often the two linked categories appear together in the relations of the entities of the other entity set.

The nodes can be filtered (by defining a visual query) to retain nodes with higher support or higher distinctivity, and observe how the inter-sector associations change accordingly. Fig. 60 shows how the entity wheel is used to find attribute-based relation-relation associations. In this figure, the wheel sectors represent selected product categories. The wheel nodes represent customers, and each sector has a node for each customer who bought a product of the sector category. The inter-sector links are assigned thickness and opacity as explained above. One can notice that the shopping centers are highly associated with clothing shops, gas stations and parking coupons. Also, gas stations are highly associated with parking coupons. On the other hand jewelry shops and hotels have generally less association with other categories. As mentioned above, the associations can also be analyzed in a subset of customers.

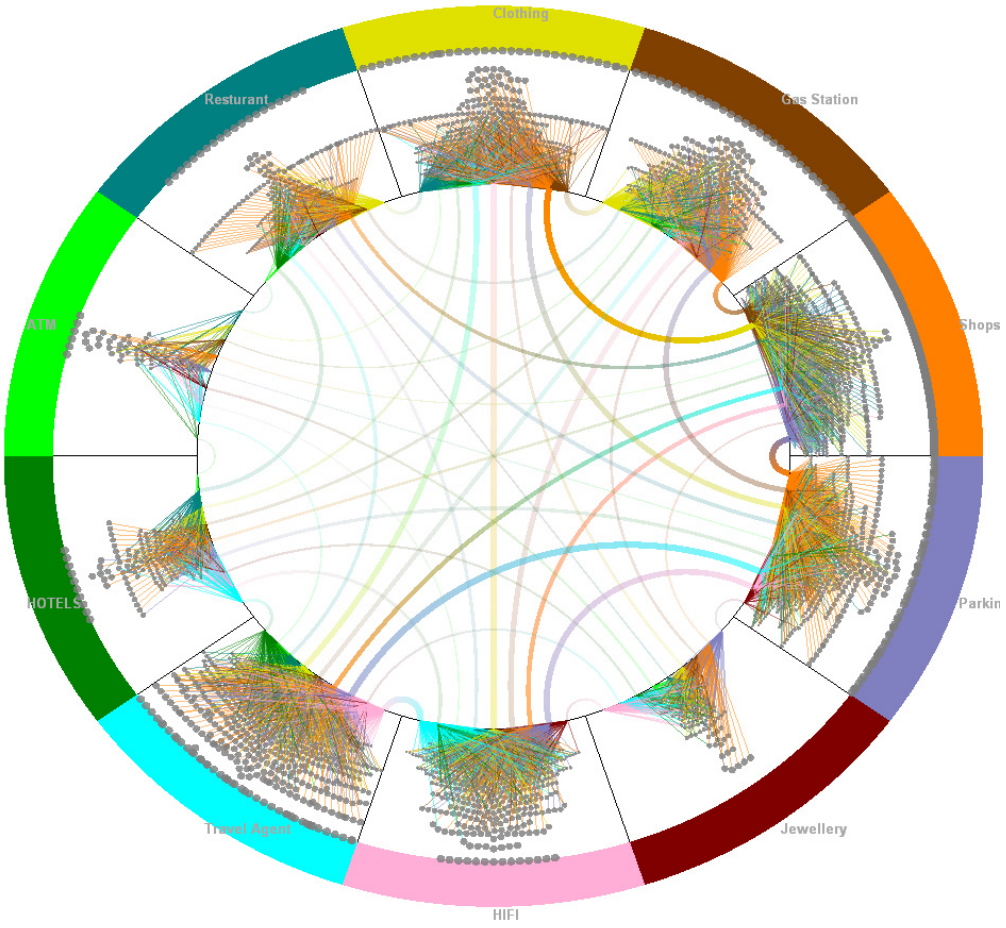

Fig. 60: Attribute-based relation-relation association analysis.

5.3.4. Ordered relations analysis

In some applications, an ordering can be defined on the relations between one entity and the entities related to it. In the graph of such entities, it is of interest to see if the ordered sequences of relations that belong to different entities exhibit some patterns.

In this work we implemented a simple approach to visualize ordered relations, which was intended to visualize the ticket flow in the ticket assignment application. In this application, a support ticket is initially assigned to a support office, and might be reassigned to multiple offices before being resolved. We are interested in seeing from which offices the tickets originate, and in which offices they are closed, and the paths they follow between their first and last assignments.

For this purpose, we created a graph exploration view where level 1-nodes are the support offices and level 2 nodes are the tickets. Links are shown using the node-repetition-and-linking with hierarchical edge bundling technique, with three modifications made to the link drawing:

1. The link between two instances of a support ticket is drawn only if the support offices the two instances belong to are consecutive in the ordered relation of the support ticket, i.e. if the ticket was forwarded from the first to the second office.

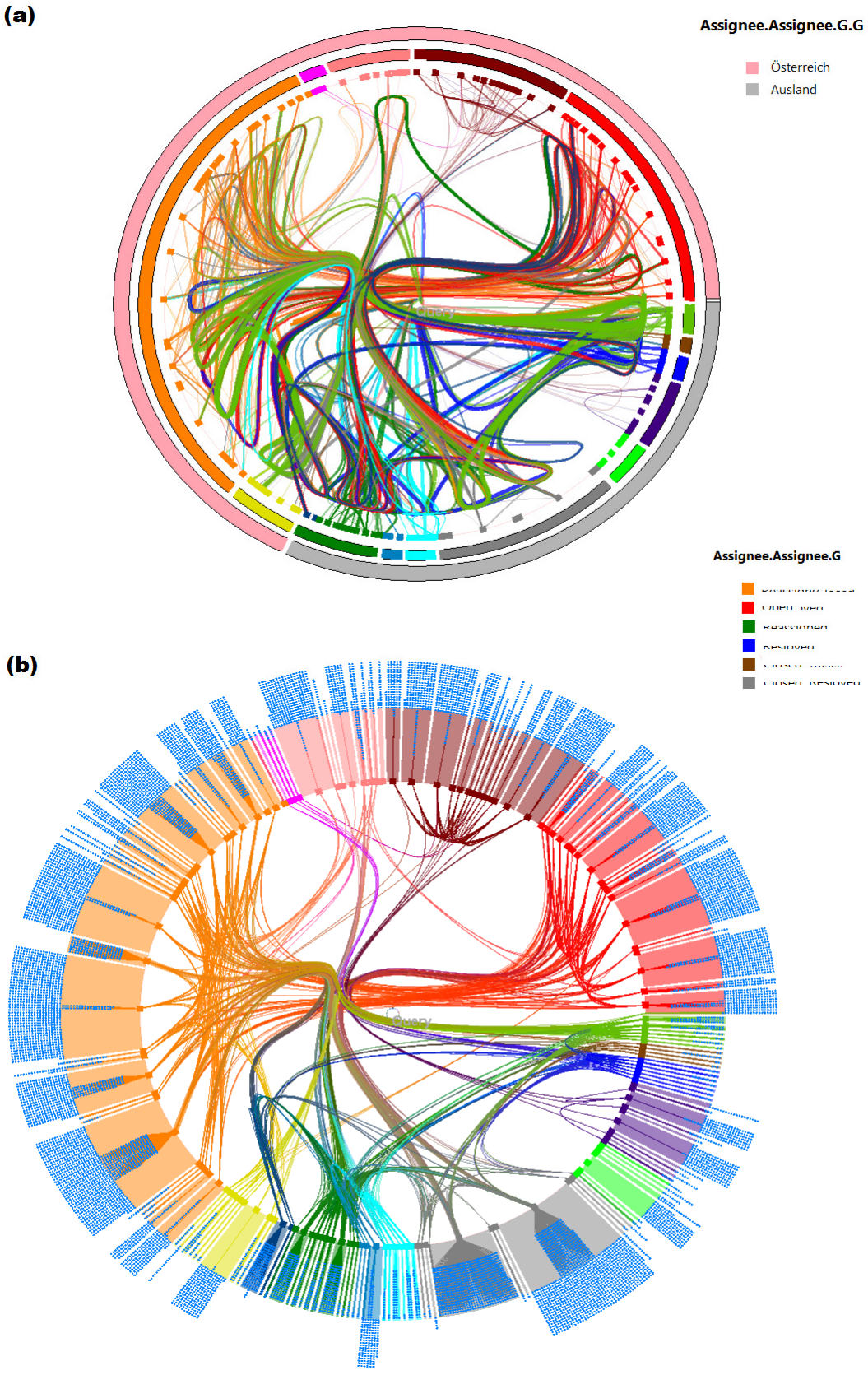

Fig. 61: Ordered vs. unordered relation visualization.

2. An end of an inter-sector link is drawn originating from the sector node only if this node is the first or the last one in the relations sequence. Otherwise, the link does not reach the node, but is moved inwards by down-scaling the radial component of the polar coordinates of the sector node. This intends to show the sequence as a continuous curve which meets only the first and the last node, and approaches nodes between them without meeting them (since otherwise it will not be perceived as a continuous curve).
3. A link is drawn thick and with higher opacity if it is part of a long series.

These modifications are designed to focus on showing the flow of the support tickets between the nodes. Fig. 61-a shows an example visualization with these modifications, while Fig. 61-b shows the standard inter-sector relations (without accounting for relation order). Basically, the links in Fig. 61-a are a subset of the links in Fig. 61-b, but with different edge placement and coloring policies. The resulting view has the advantage of showing offices that very frequently forward tickets that are initially assigned to them, or offices that frequently resolve tickets or offices that mostly forward the tickets forwarded to them.

Chapter 6

Implementation Details

"A good idea is about ten percent and implementation and hard work, and luck is ninety percent" - Guy Kawasaki

This work has been implemented as a relational data analysis module within the Senactive InTime™ event-based system. Nevertheless, the implemented techniques are generic and can be applied in other domains which need relational data analysis. In fact, by manually writing a data description script, the current implementation allows analysis of generic relational data stored in any relational database.

This chapter will provide an overview of the implementation environment, and how the data to be analyzed are defined by the user. The implementation modules used for the visualization are listed, and finally, technical details are provided.

6.1. Environment

The working environment, Senactive InTime™, implements the concept of typed attributed events and is divided mainly in three modules:

- **The Modeling Studio**

  Using this module, the user can specify an event-based application by defining the event types it processes along with their attributes, the correlations between them, the event evaluation graph which defines the desired event-processing flow, and various other aspects such as event persistence, event adapters, simulation models and response services. A set of diagrams supports the modeling process by showing graphical elements for the events and correlations between them, and for the components of the evaluation graph and the processing flow between them.

- **The Runtime Module**

  This module is responsible for runtime execution of the user defined event-based application. It performs the sense-and-respond cycle depicted in Fig. 3.

- **The Event Analyzer**

  This module offers various data analysis tools to get insight in the events received or generated in the system. They encompass tools for clustering, similarity search, correlation analysis, and time-line analysis to mention a few.

6.2. Data Modeling and Acquisition

As mentioned in section 1.2, the concept of events might bear a low level abstraction, which might make direct analysis for relations between them as entities rather complicated. Therefore, in this work, entities have been implemented as standalone analysis concepts that encapsulate and aggregate the information from events. This separation brings the advantage of making the concept of entities generic.

In this work, entity types and relations between them can be defined in a visual editor which was implemented for this purpose as an extension in the Modeling Studio module of Senactive InTime™. The definition is performed by specifying the name and the set of typed attributes for each entity type and for each relation defined between two entity types [Fig. 62]. The attributes can be of the following implementation types: int, long, float, double, date and string.

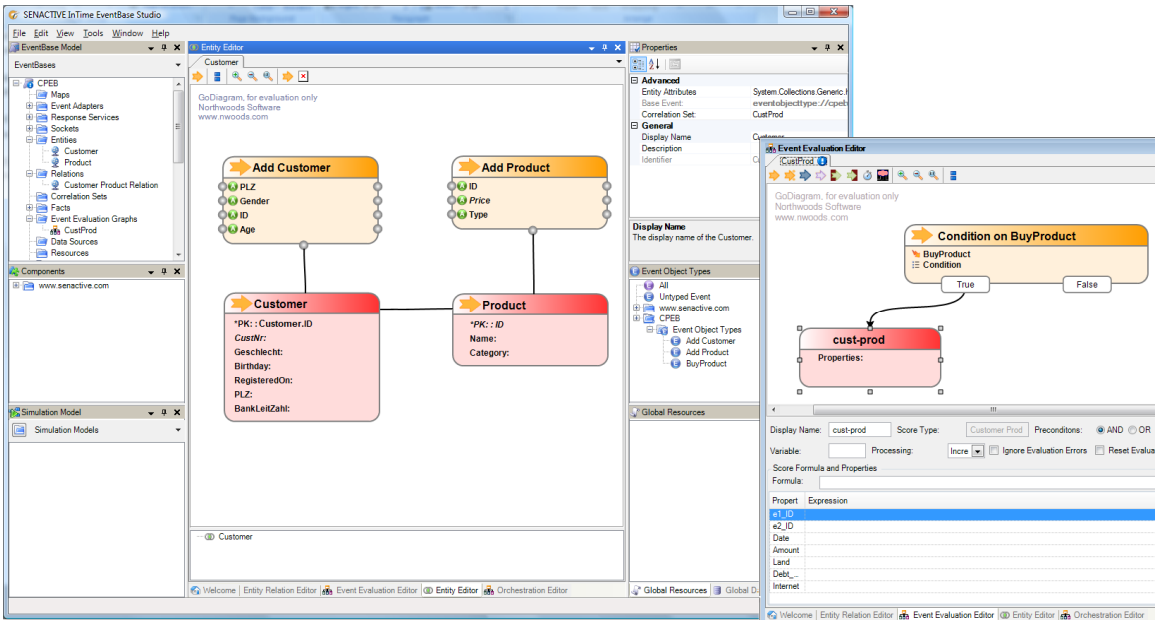

Fig. 62: Modeling entities and relations in Senactive InTime™ Modeling Studio.

The attributes of entity types and relations can derive their values from the attributes of event types as explained in section 3.1.1. The user can define an expression for this purpose which computes the entity or relation attribute value based on values of multiple attributes in multiple correlated events by means of the Event Access [EA] expressions facility implemented in Senactive InTime™ [SRRS07]. The defined entity types and relations are saved within their event-base (the application), and the corresponding database tables are automatically created or updated upon saving.

Based on the above definition, entity and relation instances are created and their attributes are assigned values at runtime. The runtime module in Senactive InTime™ was extended for this purpose, so that the entities are created / updated upon receiving a new event, and relation instances are created when the defined conditions in the event evaluation graph are met.

6.3. The Visualization Modules

The visualization modules in this work have been implemented as sub-modules within the Event Analyzer. The Event Analyzer provides a convenient windowing system that allows the creation and management of multiple windows, and defining *perspectives* which in turn divide the viewing area into predefined customizable regions. The implementation is divided into three sub-modules:

- **Graph loader and attribute analysis module**

  In this module, the user selects a relation to analyze from one of the available event-bases (applications). The entities and relation instances of the selected relation are then loaded into the memory and a set of transfer functions for the intrinsic and network attributes is instantiated and initialized based on the loaded data. This makes the attribute analysis module ready to use.

  In this module, the user can customize the transfer functions and assign the desired colors and opacities by creating a histogram as explained in section 3.4.1. Additionally, the user can define hierarchical groupings by means of grouping attributes, and define aggregated attributes as well. Transfer functions for these additional attributes can also be defined.

  This view also implements conditional histograms and visual query definitions. A query definition is implemented by a set of constraints on the attributes. This set is passed to other views for the purpose of filtering, highlighting or viewing the visual query results.

- **Graph exploration module**

  This module implements all the graph drawing algorithms presented in chapter 4. The module comprises two separate windows one for rendering the graph drawing results and performing interaction with them, and one for specifying the layout parameters and performing other operations on the view. The implemented view is used to explore the graph, show visual query results, and to perform some of the association analysis methods mentioned in chapter 5.

- **Entity wheel module**

  This module implements the detailed attribute-relation association analysis. Like the graph exploration module, it comprises of a drawing window and a control panel.

- **Parallel-sets module**

  This module implements the detailed attribute-attribute association analysis. Like the previous two modules, it comprises of a drawing window and a control panel.

6.4. Technical Details

- The work has been implemented in the Microsoft .NET development environment using the C# 3.0 programming language.
- The graphical library used is MDX 1.1 (Managed DirectX). The methods used to draw graphical elements are all implemented in one class, and are used by all the modules. This separation makes it easier to replace the graphical library.
- The user interfaces are built using the Infragistics 7.0 user interface components.
- The persistence is implemented using the NHibernate ORM framework which performs object-relational mapping and all the required functionalities for database connectivity with different database dialects.
- The diagrams for entity and relation modeling are implemented using the Northwoods GoDiagram library

Chapter 7

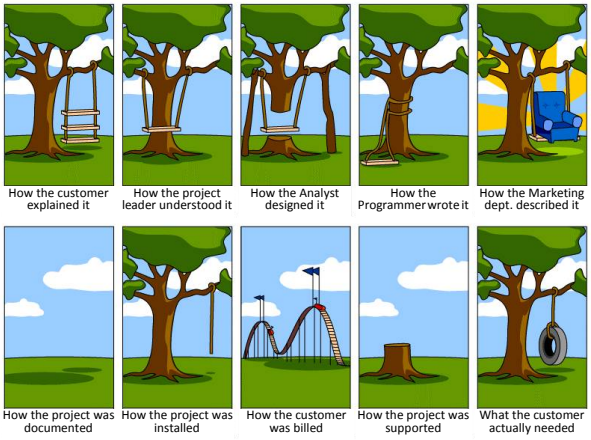

Fig. 63: IT Project Life Cycle [anonymous source].

Evaluation Results

"Only damaged people want good things to happen to them through visualization. They want something for nothing". Edwin H. Land

This work has presented several approaches for attributed graph visualization, attribute distribution analysis and association analysis in relational data. In what follows, the presented techniques will be evaluated in terms of their readability and usability, and their limitations will be investigated. The work has been developed as a prototype to investigate opportunities to perform analysis in relational data derived from event-based systems. Datasets from two real-world applications have been used to perform the evaluation: the customer-product relationships and the support tickets assignment.

A major part of the evaluation was performed by the author, with feedback mainly collected from the development and marketing team at Senactive Inc. Additionally, general feedback about the visualization was collected from the visualization group at the Vienna University of Technology, and several other colleagues. The evaluation was by no means exhaustive, due to limited availability of the target users, and due to the fact that the work has been developed as an experimental prototype, independently from the mainstream application.

The first section will discuss the strengths and limitations of each implemented visualization module, based on experimental results as well as user feedback. An evaluation of the system functionality as a whole and a summarization of the evaluation results of the single modules are presented in section 7.2. Section 7.3 presents how these modules can be applied to solve real-world problems in the two example applications, and section 7.4 compares the presented analysis tools with other approaches to solve similar problems. Section 7.5 discusses future improvements and section 7.6 concludes the work.

7.1. Evaluation of Single Modules

In the following subsections, the evaluation results for each of the presented concepts and visualization techniques in this work will be discussed. For each view, the readability, ease of use, available / missed features, and scalability will be evaluated.

7.1.1. Attribute model and distribution analysis

As explained in chapter 3, the attribute model has incorporated several extended attributes beside the intrinsic ones. The network attributes and grouping attributes in the attribute model were generally intuitive to interpret and use by the users. They were easy to treat like the intrinsic attributes. The users varied in their response to layout-based attributes. While users with previous experience in network analysis found this extension straightforward, other users needed more time to precisely understand what these attributes represent and for which purposes they can be used. Likewise not all users found the aggregated attributes intuitive and useful.

Apart from understanding views involving additional attributes, the users also varied in their ability to use these additional attributes on their own, to model certain problems or queries. To improve the user experience with these concepts, several examples are needed to demonstrate how these attributes can be employed in the analysis.

The global histograms were found easy for users to create, interpret and interact with. Also, the method for creating conditional histograms and defining queries was understandable by the users, though not particularly easy to use. The users were different in their ability to interpret conditional histograms in the four available modes (absolute values, sub-histograms, ratio growth, and per-entity frequencies).

Sub-histograms were intuitive to interpret by most of the users and effectively emphasized that a sub-histogram represents frequencies in a subset of the data. However, in the current implementation, no special scaling was implemented for sub-histograms that have frequencies substantially smaller than their global counterparts, which limits the usefulness of this mode.

Histograms depicting ratio growth were useful for showing disproportionalities. Yet, not all users could interpret them correctly. The depicted disproportionalities are based on statistical supports which are mapped to the bar opacities, nevertheless, some users were unable to read the desired information. Per-entity histograms were rather easy to interpret by the users.

Currently, little information is provided to help user orientation in the query definition process. The view should provide information about the subset selected for the conditional histogram (by a textual description of the conditions or by visualizing these conditions as intersecting subsets).

7.1.2. The nodes-only graph exploration view

The nodes-only graph exploration module provides a simple overview of the neighboring subgraph of its central node. This view, accompanied with its interaction facilities, was found efficient at producing large nodes in this subgraph and at providing insight into the values of the attributes in these nodes (thanks to node sorting). However, ordering nodes of a numerical attribute in the 2-dimensional polar space was in some cases judged as unintuitive.

The animated navigation implemented in this view, has improved the perception during the navigation and added a satisfactory feeling of continuity to the navigation, compared with abrupt view changes in a static navigation. It effectively shows how many of the current nodes remain visible in the next view.

The majority of interaction facilities in this view such as defining the layout and sorting parameters, the visual mapping, and animation and selection parameters, were generally found easy to understand and intuitive to use. The selection of multiple nodes is, however, less intuitive.

One missing feature which the users expected to see was node annotation. The current view can show the attributes of one node by hovering the mouse over it or by clicking it. Drawing annotation text is by no means a straightforward task: similar to how drawing links as straight lines can considerably reduce the readability, drawing annotation texts besides nodes without incorporating them in the logic of the layout might also make both the annotations and the graph unreadable. The absence of the annotation slows down the comprehension of the view, and hinders a quick comparison of the nodes (since the user has to manually figure out which entity is represented by each of the nodes individually, using the mouse).

Also, the absence of links in this view limits a global comprehension of the relations in the graph. However, this is compensated in the view dedicated to show the links, and partly by using the layout attributes “PredDegree” and “SuccDegree”.

In a 1000x1000 pixels screen area, the view could accommodate 10000 nodes with large enough sizes to select them and comprehend their visual attributes. This number is sufficient to analyze two levels (other than the center) of medium-sized or large graphs which branching factor is smaller than 100. This is sufficient for many practical problems. For larger graphs, a clustering need to be employed which is beyond the scope of this work.

7.1.3. Graph drawing by random-parent strategy

When testing the method explained in section 4.3.1 for graph drawing with links, the results have shown significant inconsistency. Not only was it hard to find the nodes that are related to a level-1 node, but also, the links were found hard to follow and comprehend by the majority of the observers. One user commented that following the links is as hard as path-finding puzzles. The randomness has not only caused the inconsistency in the view but also made it hard to attach meaningful semantics to the edges.

7.1.4. Graph drawing by node-repetition-and-linking

Another approach to draw edges in the graph was to draw an instance of each level-2 node in the sector of every parent node, and then link the instances via an inter-sector link. The resulting view has considerably better readability than the random-parent view and it allowed finding the children for each level-1 node quickly. Also, having these children in one place enabled sorting them by an attribute.

When bundled as arcs, the inter-sector links effectively exhibited which two sectors were highly related (or unrelated). When using hierarchical edge bundling, the inter-sector links effectively exhibited how the categories of a level-1 attribute are inter-related within the set of level-1 nodes. Also, it has been found to perform a more effective bundling, with less-cluttered views and to have the ability to handle larger graphs than the bundling by arcs technique.

The technique has produced a readable view with about 200 level-1 nodes and 5000 level-2 node, using hierarchical edge bundling. However, the repetition of nodes was misleading for some users despite the linking and the concurrent highlighting of all instances of a node when it is selected. Possibly, the visual representation of a node should be changed to emphasize that there are multiple instances that belong to the same node. Though the number of nodes in a sector, and the relative sizes of sectors correctly express the concepts they represent, due to node repetition, the visual representation of the level as a whole does not express the number of nodes in it. The user should be aware of this fact to avoid drawing wrong conclusions.

7.1.5. Parallel sets

Parallel sets is a well established technique for visualizing categorical variables, and has been shown to be effective in providing insight into the associations between these variables (see evaluation results by Hauser et al. in [HKB06]). An important feature of parallel sets is that the view complexity depends only on the number of categories of the variables in the analysis, and is independent from the number of items in the data set, thanks to being a space-filling approach.

The extension of this technique to analyze relational data was straightforward, since this data is also represented by a set of attributes. The visual enhancement which adjusts the stripe thickness was found intuitive and useful by most of the users. The other extensions (assigning different heights to the categories boxes, indicating the number of relations per entity in each box and showing the relation-to-entity mapping) were also found useful, though some users reported an experience of being overwhelmed with a lot of details that negatively affected the focus and the cognition. These users preferred to see a more flexible and zoomable view with multiple levels of details, which in effect resembles a drill-down in the data (a commonly used technique in the analysis of multidimensional data). They expected the ability to zoom into each of the boxes, and see closely how the relations are distributed among the entities. The current implementation lacks this ability and it does not provide a level-of-detail analysis.

7.1.6. Entity wheel

The user impression about the entity wheel view was generally positive. It was easy to interpret by the users, and helped gaining quick insight into particularly high disproportionalities.

The same issues of node repetition discussed in section 7.1.4 are also present here. In fact, node repetition in the entity wheel is a more problematic issue since the repetition is not inherent in the relations and does not represent an intrinsic relation between a node instance and its sector. It rather represents a high enough association between an entity instance and the corresponding sector category. The distinctivity threshold controls whether a designated node instance will be created or not. Nevertheless, after working for some time with the view, the users could get used to the visual metaphors and were able to draw the correct conclusions from the view.

The sorting of the nodes was found rather limited since the usage of the node radial position to indicate the degree of association constrains the node placement and might results in placing nodes of the same category in noncontiguous regions. Also node annotation and a more comprehensive and intuitive interaction were demanded by the users.

The entity wheel has the potential to produce a readable visualization of the entity-relation associations in middle-sized or even large graphs, for the following reasons:

- The number of sectors is independent of the number of items in the entity set whose attribute is chosen to be analyzed by the view. It depends only on the number of categories of this attribute. Even if the number of categories is large, it is possible in many cases to define a grouping attribute that can substitute the original attribute and yield fewer categories that are still relevant and sufficient for the analysis.
- The nodes which have a small support (i.e., number of relations), are filtered out by a user-defined threshold (the node support threshold).
- Usually, just a few number of nodes show considerably high associations with the categories, while a large number of nodes have low associations with most of the categories. This has two advantages:
    - A small number of multiple instances will possibly be created especially with a high enough distinctivity threshold.
    - Since the node size corresponds to its association with its sector category, small node sizes will be assigned to a large number of nodes, and larger sizes will be assigned to a small number of nodes that exhibit high association with specific wheel categories.

This view has successfully been used to quickly and interactively find the most relevant attribute-relation associations present in the relational data. This is aided with the adjustable node support threshold and distinctivity threshold, and the visual metaphor which maps the association degree to the node size and radial location.

The inter-sector links were found straightforward to interpret. With a growing number of sectors and of multiple instances, the clutter increases in the link visualization and reduces the readability. Luckily, this clutter can be reduced by blending links with low multiplicity, which gives focus to links that represent significant associations between the sectors they connect.

7.1.7. Association analysis in graph drawings

As explained in section 5.2.1 the attribute-relation associations for a given entity can be performed using the nodes-only graph exploration view for this entity. This view was found intuitive and effective in showing these associations. In case of a categorical attribute, the readability depends mainly on the number of categories, since each category will be assigned a sector in the corresponding ring, and these sectors should be large enough to produce a readable view.

The use of aggregated attributes to find attribute-relation associations in a given set of entities was not found as effective. The use of color saturation to denote the degree of association was not intuitive for many users, and the current association-to-saturation mapping for different colors was not successful to produce consistent and intuitive results. The entity wheel has been more successful in performing this type of association analysis.

The use of the nodes-only graph-exploration view for relation-relation association analysis for a given entity, explained in section 5.3.2, was able to produce intuitive results and highlight such associations. However, the process of creating the view and highlighting the nodes involves several steps in the current implementation, which was found rather difficult.

The relation-relation association analysis for a set of entities, explained in section 5.3.3, was also able to produce readable results and the creation of the view involves fewer steps. The readability of this view depends, however, both on the number of nodes in the set and on the number of categories they belong to. Associations between about 200 nodes that fall in about 20 categories were successfully visualized using this view (with hierarchical edge bundling). The small number of categories helped the hierarchical edge bundling technique to produce more compact and less cluttered visualizations. To discover relation-relation associations between a larger number of entities, the standard data-mining tools are probably more efficient, and can be extended to more complex associations (e.g. using the Apriori algorithm [M67] to find association rules).

Finally, the visual analysis of ordered relations, explained in section 5.4 has been found largely inadequate and offered less information as desired. This is mainly because of the clutter caused by long edges. The current view, however, has the potential to provide the desired information if the clutter issue is handled.

7.2. Evaluation of the Overall Framework

This section presents and overall evaluation of the framework, in terms of consistency in the visual metaphors used, and how multiple visualization modules are linked to perform otherwise unfeasible analysis tasks. Then a summarization of the evaluation of single modules is presented.

7.2.1. Consistency of the visual metaphors

The present visual analysis methods have similar basic ideas and visual metaphors, but they employ them differently. In particular, we can find the following similarities between the method for graph drawing with links, the entity wheel, and parallel sets:

- **Existence of primary and secondary items of analysis**
  a. In the graph drawing view, the primary items are level-1 nodes, visualized by graphical nodes along with their sectors. The secondary items are level-2 nodes (nodes connected to level-1 nodes) visualized as nodes in these sectors.
  b. In the entity wheel, the primary items are the categories of the wheel attribute, visualized as sectors. The secondary items are the entity instances of the related entity type, visualized by graphical nodes in these sectors.
  c. In parallel sets, the primary items are the categories of each variable, visualized as boxes. The secondary items are the entities that belong to each category, and their quantity can be visualized by means of the box height.
- **Drawing links between the primary Items**
  a. In the graph drawing view, two sectors are connected with links that represent multiplicity of paths of specific length (1, 2 or 3) between them.
  b. In the entity wheel, two sectors are connected with a link that represents how many entity instances have significant distinctivity in both sectors.
  c. In parallel sets, two boxes are connected with a stripe that represents the frequency of items that fall in the connected categories.
- **Drawing the relations between the primary and the secondary items**
  a. In the graph drawing view, the links between a sector and its nodes show how the inter-sector links of this sector are distributed among these nodes.
  b. In the entity wheel, the links between a sector and its nodes show which of these nodes have other instances in other sectors.
  c. In parallel sets, the graph of Fig. 49 shows how the relations that fall in a category are distributed among the entities that fall in this category.

7.2.2. View linking

The main focus of this work has been on implementing the analysis tools presented in the above section. The linking of these views, and the analysis use cases that involve multiple views have not been thoroughly investigated.

A diagram which explains how the analysis views are linked is depicted in Fig. 2. As can be seen in this diagram, the attribute distribution analysis module lies in the center of the analysis and is linked with the other modules. This module is responsible for speciying a subset of the data by means of defining visual queries. These queries are then sent to one of the other views along with the command to execute (highlight existing nodes, visualize the query results, filtering, etc). Additionally, the entities in the association analysis views can be visualized directly in the graph exploration view to closely examine their neighboring subgraphs.

These functionalities enable modest linking between the views. Though for users experienced with the framework many analysis tasks can be performed using these functionalities, for most of the users, the linking is not easy to use and has considerable shortcommings.

A better view linking should enable the user to define views she wants to link, and enable realtime highlighting of entities in all of these views, based on user-defined selections in any of them. Also, frequent use cases that involve several interaction steps in multiple views can be provided as new functionaltities that combine these steps.

7.2.3. Summary of the modules evaluation

Table 1 provides a summary of the evaluation of the visualization modules.

Visualization technique / features	**Attribute Distribution Analysis**	**Graph Exploration (nodes-only)**	**Graph Exploration (with-links)**	**Parallel Sets**	**Entity Wheel**
Functions	Distribution analysis, visual mapping, defining queries	Exploration Local association analysis.	Relation-relation association analysis	Attr-attr association analysis	Attr-relation association analysis
Intuitiveness / readability issues	Difficulty with ratio-growth	Sorting by a numerical attribute not always intuitive	Node repetition sometimes misleading	Good overall readability	Node repetition and sorting sometimes misleading
Usability issues	Limited updating of conditional histograms	Minor multi-selection issue	No link selection or sector ordering	No category ordering	No category ordering or multi selection
Scalability with respect to nodes	Independent	Up to 10.000 nodes	Up to 200 nodes in level-1	Independent	Controllable w. two thresholds
Scalability with respect to categories	Independent	Up to 100 categories in a ring	Up to 20 categories in level-1	Up to 20 categories per axis	Up to 20 categories (wheel sectors)
Missing Features	Textual/graphical query description	Annotations	Annotations	Drill-down in each box	Annotations, multi-selection

Table 1: Evaluation of the visual analysis techniques

7.3. Case Studies

As mentioned in section 1.4, the visualization framework has been tested to analyze data of two applications: the customer-product relationships and the support tickets assignment. This section explains how analysis tasks are performed using the implemented views.

7.3.1. Customer-product relationships

Various analysis tasks of this application have already been presented and used to explain the developed approaches throughout this work, and most of the illustrations were produced from the dataset of this application. Here, we will present how the analysis tasks mentioned in section 1.4.1 are implemented using the developed approaches. The analysis tasks are concerned with association analysis. Use-cases both for the association discovery and verification are provided.

7.3.1.1. Clothes and jewelries are bought more frequently by women than by men

This association can be found by creating a parallel-sets view of the purchases dataset and plot the product category against the customer gender. Fig. 64 shows the parallel-sets view of customer gender vs. product category in the purchases. The box which represents these purchases shows that clothes and jewelries are bought more frequently by women, and these associations can be verified in the conditional histogram of the categories of the purchases of female customers in Fig. 45.

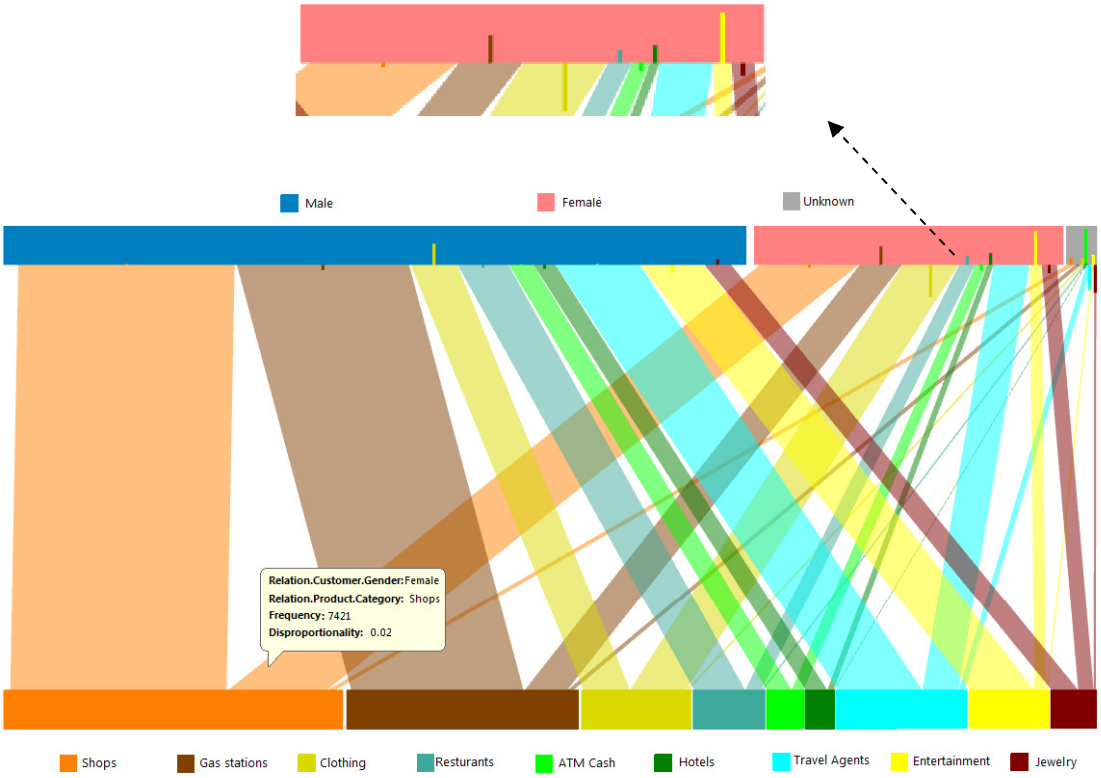

Fig. 64: Disproportionalities of products in purchases of female customers.

7.3.1.2. Travel agents are used more frequently by older people than by younger ones

Likewise, this association can be found by creating a parallel-sets view of the purchases dataset and analyzing the product category against the customer age group [Fig. 65]. The dominance of older people in the purchases from travel agents as well as gas stations can be easily spotted. This association can be verified by visualizing a query which selects the travel agents in a graph exploration view and analyze the distribution of customers' birthdays in the level-2 of this view. Alternatively, it can be verified by creating a conditional histogram of the customers' birthdays in the subset of purchases which has gas stations as product category.

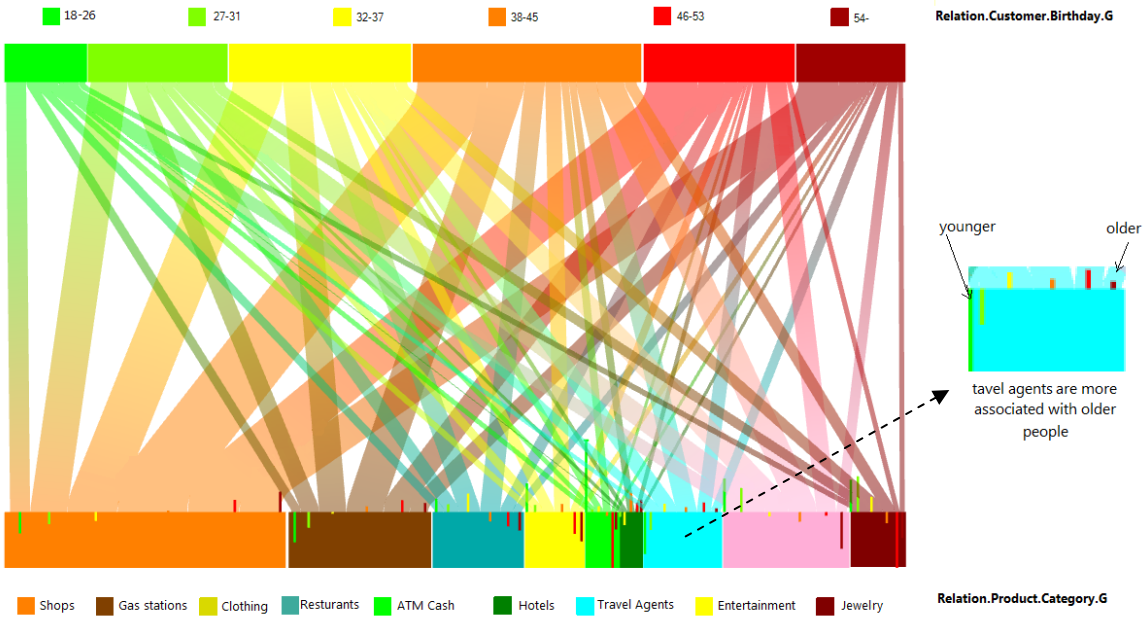

Fig. 65: A parallel-sets view of customer age group vs. product categories in the purchases.

7.3.1.3. Customers from Styria who used gas stations are mostly men

This association can be discovered by creating a parallel-sets view of the customers' provinces vs. product categories, and choosing the customer gender to color the relation-to-entity mapping in the postal address categories (which represent the Austrian provinces). Fig. 66 shows the box which represents purchases of customers from "Styria". It is noticeable in this view, that the purchases from gas stations in this box have been performed largely by male customers.

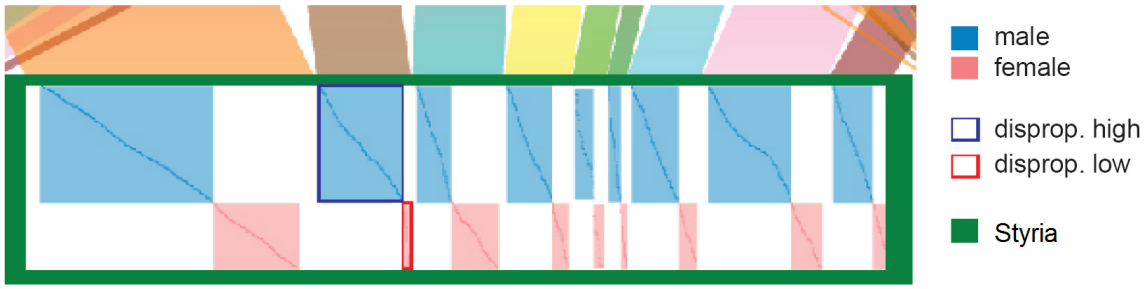

Fig. 66: Purchases of customers from Styria, divided according to product category and gender.

This association can be verified by querying the purchases for gas stations and customer postal address in the group “Styria”. A conditional histogram of the customer gender in the resulting subset is created and the ratio growth is computed for each gender.

7.3.1.4. Hotel reservations by customers from a specific province in Austria are more likely to be done in specific months

This association can be discovered by creating a parallel-sets view of purchases which fall in the “Hotels” category and choosing the attributes “Relation.Customer.PLZ.G” and “Relation.DateTime.Month.G” for the axes. The first attribute represents the provinces in which the customers live, and the second one represents the months in which the reservations were done (the added grouping aims to make the attribute categorical).

In each box representing a province, the vertical lines show in which months the reservations from this province were higher or lower than the average. As can be seen in Fig. 67, the reservations of customers from “NÖ” (Lower Austria) peaked in May, June and December.

The associations can be verified by selecting the subset of purchases in Hotels and a selected living province of customers. The conditional histogram of the relation attribute “Relation.DateTime.Month.G” in this subset is created with the absolute-values mode, which shows in which months the reservations peaked. Using the ratio-growth histogram mode, we could verify that the peaks in May and June are specific to hotel reservations and are not according to a general trend in the purchases.

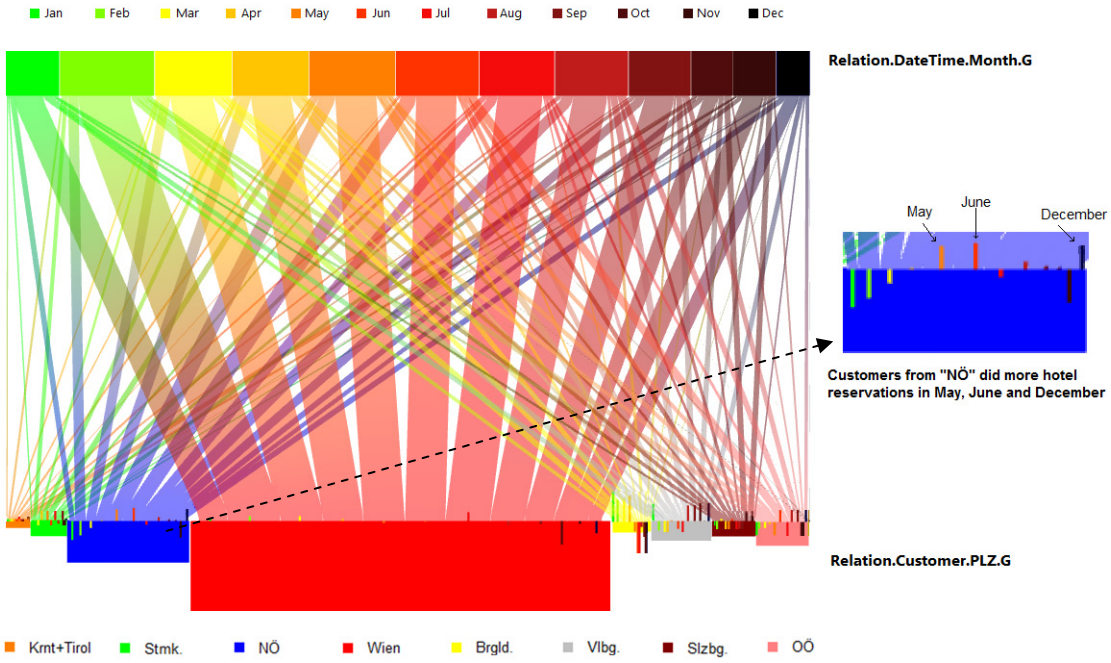

Fig. 67: Analysis of hotel reservations by month and customer living place.

7.3.1.5. Customers who used gas stations are more likely to buy parking coupons

This is an attribute-based relation-relation association. It can be discovered by creating an entity wheel whose sectors represent the product categories, and assign the parameters as explained in section 5.3.3. Fig. 60 shows the entity wheel for a subset of product categories. It shows that gas stations are highly associated both with shopping centers and with parking coupons.

The verification of this association can be performed by creating a query of products which fall in the gas stations category, and visualizing its results in a nodes-only graph exploration view. The level-2 nodes in this view represent the customers who ordered one or more of these services. Level-3 nodes contain other products they have bought. By creating a conditional distribution of products whose "NodeLevel" attribute is equal to three, one can see that parking coupons are disproportionally high [Fig. 68].

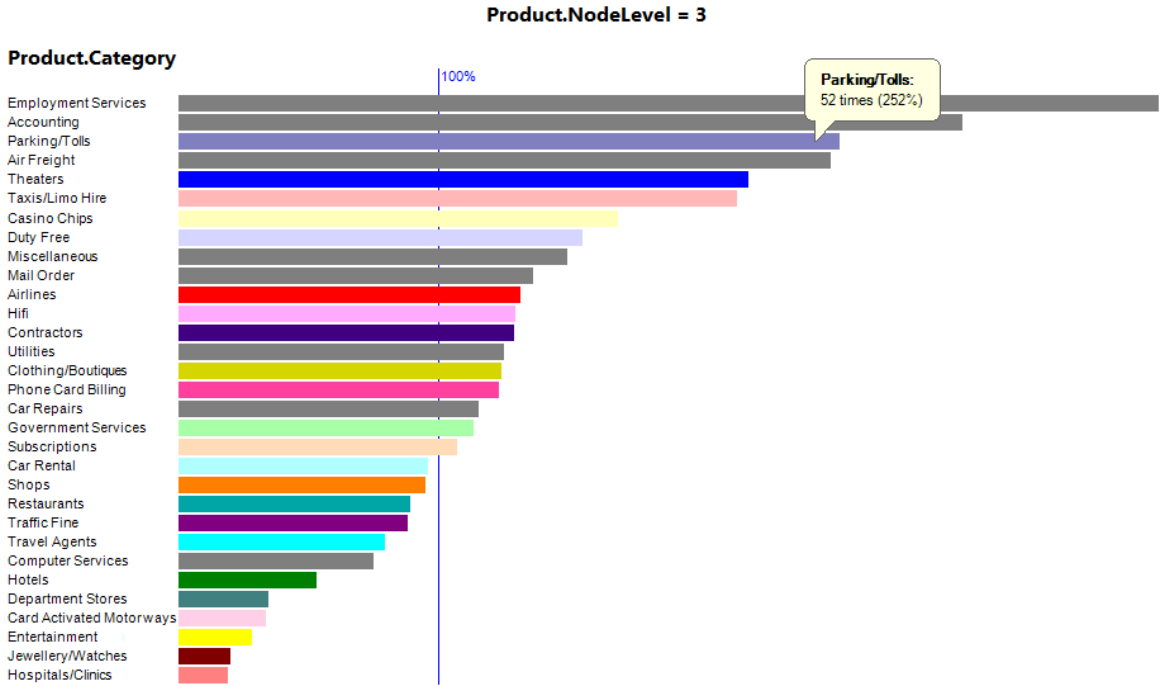

Fig. 68: Verification of the association by a conditional histogram.

7.3.1.6. Young women who bought jewelries are more likely to buy luxury watches

Currently, there is no single view that helps discovering this association directly. A brushing step is needed to select a subset of the customers, and to see what relation-relation associations exist in their purchases. This can be performed by creating an entity-wheel view whose sectors are the product categories, and then selecting the subset of young female customers in the attribute analysis view, and restricting the wheel inner nodes to these customers. The resulting wheel links represent mutual purchases of these customers, and show existing associations between the product categories in these purchases.

7.3.2. Support tickets assignment

In this application, the basic analysis which aims to explore distributions and finding the assignee of each ticket and vice-versa was straightforward based on the implemented tools.

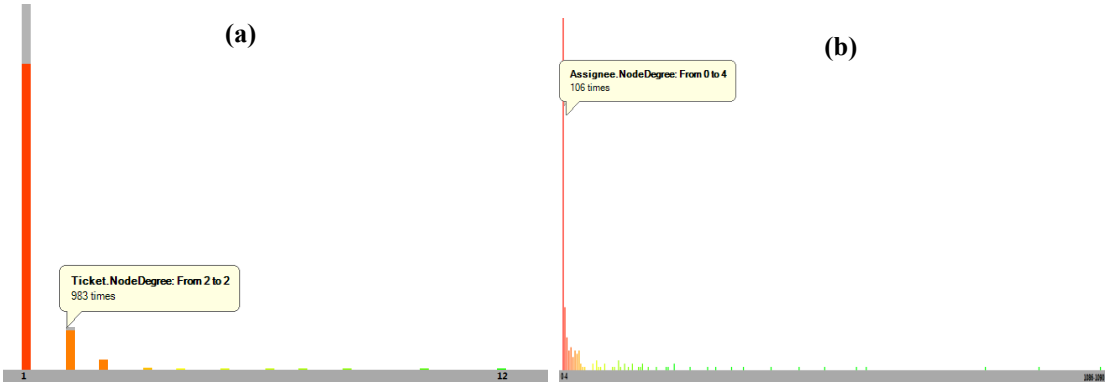

Fig. 69: (a) Conditional histogram of the tickets NodeDegree (sub-histogram represents resolved tickets). (b) Histogram of the assignees NodeDegree.

From the distribution of the NodeDegree attribute of the tickets [Fig. 69-a], one can conclude that most tickets were assigned to one support office only, and the majority of them were resolved by it. The ratio of resolved issues increases with increasing number of support offices assigned to the ticket. Also, from the distribution of the NodeDegree attribute of the support offices, one can see that most support offices have processed 0-4 tickets [Fig. 69-b], while there are offices that processed as much as 1090 tickets.

The graph exploration view has been used successfully to show which tickets have been processed by each support office, how many steps the processing took for each ticket, and what the final status of this ticket at this support office was (resolved, closed, forwarded, open). Also, the user could quickly click a ticket to see other support offices which participated in processing it. Fig. 70-a shows an example of this view, where the relation instances were colored according to the ticket status upon processing the corresponding ticket event, and sorted according to their creation times.

The relation-relation association view could also be successfully used to show the communication between support offices. Fig. 70-b shows the communication between support offices which processed more than ten support tickets. The sector colors denote the hierarchical grouping of the support offices. The view shows that some offices have not processed tickets exclusively (since they have no ticket nodes beyond the sector boundaries), and some offices have frequently processed tickets of high (resp. low) priorities. Fig. 70-c shows the same offices, but uses the above level of hierarchical grouping to color the sectors and the links between them. This grouping enables seeing the communication between Austrian and non-Austrian branches. In both figures, one can see that one non-Austrian support office has processed a large number of tickets together with two Austrian offices, and another non-Austrian office has also high association with two other non-Austrian offices.

The ordered-relations view suggested in section 5.3.4 to analyze the flow of tickets between the offices was unable to provide appropriate insight into the flow.

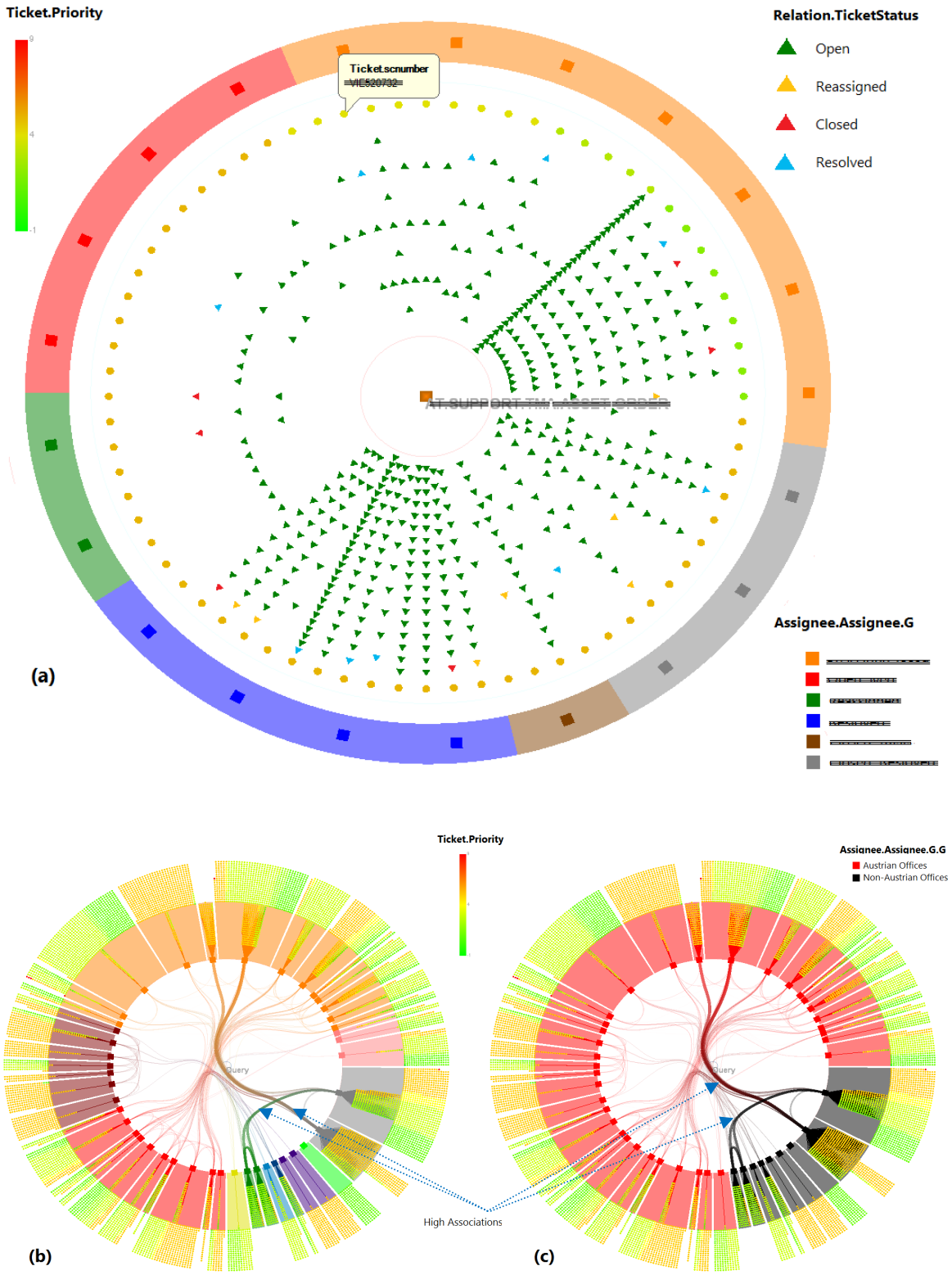

Fig. 70: Ticket assignment analysis. (a) Tickets assigned to a support office. (B, c) communication between a subset of support offices.

7.4. Comparison with Other Approaches

It is always difficult to compare the performance of two systems, especially if they were using considerably different methods to solve the problem. Also, the lack of access to the other systems makes an objective evaluation unfeasible. Nevertheless, this section tries to discuss similarities and differences between the implemented methods in this work, and other approaches to solve similar problems.

7.4.1. Comparison with other visualization methods

The graph drawing using node-repetition-and-linking has advantages over the Anchor Maps of Misue [M08]. In our work, since all the nodes related to an anchor node are placed in its neighborhood, the inner area is dedicated to links (no nodes are placed along links), and the links are bundled to reduce clutter and have clearer semantics (inter-sector relations). However, node repetition might be misleading when trying to establish a global understanding of the drawing.

Also, when used for drawing the neighboring subgraph of the central node, our method incorporates all the adjacency relations between the first and the second level while the radial layout method of Yee et al. [YFDH01] draws only the edges of a spanning tree in the graph (drawing all the edges causes significant clutter in their method). However, links in further radial levels are totally ignored in our work.

Most of the non-node-centric methods for navigation in graphs employ force-directed methods to layout the graph, and hence enjoy many of the aesthetics criteria of graph drawing [section 2.1]. The more even distribution of nodes is an important criterion to create readable annotations and map complex visual properties to the nodes, such as image icons. These two features are important in social-network analysis systems, and hence many of these systems, such as VizSter [HB05], use force-directed methods for network navigation. Yet, these methods do not always produce consistent drawings, and do not allow a node-centric navigation which causes that not all the nodes related to a node are guaranteed to lie in its neighborhood. Also, the clutter increases significantly with a growing number of edges since they are usually drawn as line segments. All the above mentioned methods fit a relatively small number of nodes, and need a level-of-detail exploration to handle large graphs.

A thorough comparison of the parallel-sets method with other approaches for analyzing categorical variables can be found in the work of Hauser et al. [HKB06]. The extensions provided in this work for analyzing relational data proposed a sub-division of category boxes [Fig. 66], which resembles the method of mosaic displays. To the best of knowledge of the authors, no other work has employed parallel sets for relational data or suggested the combination with mosaic displays.

One way to visualize the attribute-relation associations addressed by the entity wheel is to use parallel coordinates. The categories represented by sectors in the wheel, are represented by the parallel axes instead, and each inner entity instance in the wheel is visualized instead by a polyline connecting the coordinates that represent its distinctivity in each axis category. The wheel has the advantage of assigning a two-dimensional sector for each category rather than a one-dimensional axis, which provides

more space to visualize the entities in each category. The usage of radial placement enables a natural scaling to emphasize high associations. This is also an advantage for the wheel over the usage of a scatter plot. Furthermore, the placement in the wheel allows routing the links between sectors in the inner circle, which would be cumbersome in a standard scatter-plot or in parallel coordinates.

More elaborate methods exist for defining visual queries compared to the one implemented in this work, such as VizQL, the visual query language introduced by Hanrahan [Ha06] and employed in the Polaris data visualization system*. Also, many social-network analysis systems attach additional attributes with the network nodes (such as Pajek [NMB04]).

7.4.2. Comparison with non-visual approaches

More often, conventional approaches that do not rely on visual representations are used to perform relational data analysis. Not only do they exist longer than visual methods, they are also still considered more reliable and practical to use, and do not suffer from many of the limitations found in the visual analysis methods.

Exploration of relational data is typically performed in a tabular view using data grids whose rows represent the items of one entity set along with their attributes. For each item, the user can explore entities related to it in a subgrid below it [Fig. 71]. These grids allow data editing and offer typical spreadsheet functions. Many systems have incorporated basic information visualization techniques into these grids (such as coloring the grid cells according to their contents).

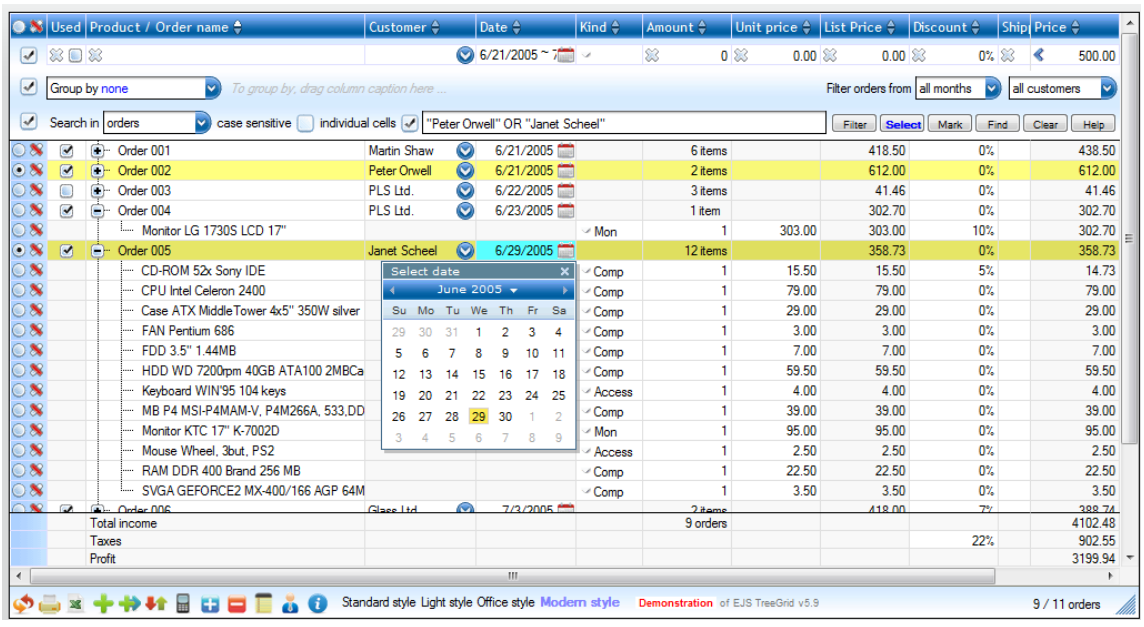

Fig. 71: A tree-grid. [TreeGrid]

Tabular views are efficient at showing detailed information of multiple data items, and allow intuitive and rich interaction operations with these data items. However, they

* Later, called Tableau Software.

can show only a small portion of the data at once, and hence do not give insight into the whole data. While they are suitable for a detailed exploration of items related to one item, they cannot provide information about how multiple items are inter-related. To compensate for these limitations, these tables are often enriched with other analysis tools such as charts. Also, they can be linked with more elaborate visual analysis methods to combine advantages of both approaches.

A variety of data mining methods exist to compute several types of associations in relational data. These approaches have been able to discover complex associations in which several attributes are involved. This offers more advanced association analysis than the one in this work. However, as discussed in section 2.3.2, interactive visual analysis still has the advantage of incorporating user knowledge and cognitive power in the analysis process. This helps in providing the user with an overall comprehension and awareness of the data, and enabling her to discover patterns not addressed in the data mining algorithms being used. In addition, the visual methods allow a quick verification of whether the associations found are inherent in the data, or are due to outliers.

7.5. Future Work

Various future extensions are possible to increase the insight gained from the developed tools, and make them more usable. First of all, more elaborate evaluation and user studies need to be performed. Currently, the author is preparing for an evaluation of the visualization methods through online information visualization communities.

In the graph exploration view, an attribute-based as well as a graph-theoretic hierarchical clustering can make the exploration more intuitive, enable abstraction of the view, and make it possible to analyze larger amounts of data. Also, the view needs more elaborate zooming and panning, as well as node annotation. Also, a non-node-centric exploration view based on force-directed methods is worth being investigated. Additionally, a tabular exploration of the relational data, linked with the graphical exploration, will increase the ability to explore the entities in greater details.

The parallel sets can be further enriched by enabling a detailed exploration of the relation-to-entity mapping in each category box as part of a drill-down in the data. Also, the overlapping resulting from adjusting the stripes thickness should be further addressed to reduce the overlapping artifacts more robustly.

A more elaborate and intuitive view linking is needed to ease finding knowledge which cannot be found in either view alone. Ordered relations, time attributes, and geographic attributes need dedicated views to benefit from their special natures. The edge bundling technique for visualizing ordered relations should be further researched to produce a more readable visualization.

Also, there are many possibilities for enhancements in the attribute analysis and visual mapping modules. The mapping of categories to color should benefit from well-studied color models, to alleviate color conflict when analyzing multiple attributes (such as in parallel sets).

Finally, the current implementation is by no means optimized, and there are many possibilities for performance optimization. Shader-based, indexed shapes need to be

utilized to accelerate the rendering. Also, the spreading of the nodes in the entity wheel was implemented by a brute force algorithm to search for free spaces. More advanced computational-geometry algorithms exist that perform considerably faster.

7.6. Conclusion

This work has presented a framework for interactive visual analysis of attributed relational data and their corresponding graphs, which comprises multiple analysis modules. One module enables node-centric navigation in the graph to explore the attributes of its components and how they are related. Another module is used to perform analysis of global and conditional attribute distributions and define visual queries. The remaining set of modules are used to perform the analysis of three types of associations, namely, attribute-attribute, attribute-relation, and relation-relation associations.

An extensible attribute model was introduced, which extends the intrinsic attributes with network-based and layout-based attributes, hierarchical-grouping attributes, and attributes aggregated over the relations. These extended attributes can be used in the same way as intrinsic attributes (e.g. in attribute analysis, visual mapping, node sorting and association analysis).

Two modes for the node-centric exploration of the graph were proposed. The first mode views only the nodes of the neighboring sub-graph of the central node in successive levels in concentric rings around this node. This mode is designed to show which attributes these nodes have and to visualize the results of visual queries. The other mode displays the edges between the nodes in the first and the second level in the radial placement, and aims at finding relation-relation associations.

The parallel sets approach was employed to perform attribute-attribute association analysis in the relations, and extended to incorporate the mapping between the relations and the entities. Also, a novel view has been proposed to perform attribute-relation association analysis.

The presented approaches were applied to real-world datasets, and have been able to provide informative navigation, and to discover useful associations in both of them. However, the usability of the provided tools is still insufficient, and there are still limitations and lacking features for a practical usage of the implemented framework. By overcoming current usability flaws, the presented approaches will be a valuable extension for business intelligence applications.

Appendix I

Adjusting Stripe Thickness in Parallel Sets

As explained in section 5.1.2, the stripes of the parallel sets are drawn as the inner area between two offsets of a C^2 curve. The curve is composed of arcs and line segments and is perpendicular to the parallel axes.

Here, a mathematical description of these offsets is presented. Fig. 72 shows an example stripe between the corresponding intervals in two boxes, where:

- R is the width of either interval.
- H is the distance between the corresponding parallel axes.
- D is the distance between the left sides of the intervals.

The two offsets are symmetric and each one consists of a long circular arc, a line segment and a shorter circular arc. We have chosen the shorter arc to have a radius equal to R and the longer arc to have radius equal to $2R$.

To preserve the perceived thickness, the distance between the two offsets should be made equal to R along the entire stripe. To achieve this requirement, the following two conditions should be met:

- The two line segments (each from an offset) should be parallel to each other, have equal length, and have distance between them equal to R.
- The offset curves should be C^2.

To fulfill these conditions, the arcs are assigned the angle which makes the arcs radii perpendicular to the line segments at meeting points, which makes the part of the stripe bounded by the line segments a rectangle.

From Fig. 72, one can see that the following relation holds:

$$\tan(\theta) = \frac{D - 3R\ (1 - \cos(\theta))}{H - 3R \sin\ (\theta)}$$

This relation simplifies to:

$$\mathrm{H} \sin(\theta) + (3R - D) \cos(\theta) = 3R$$

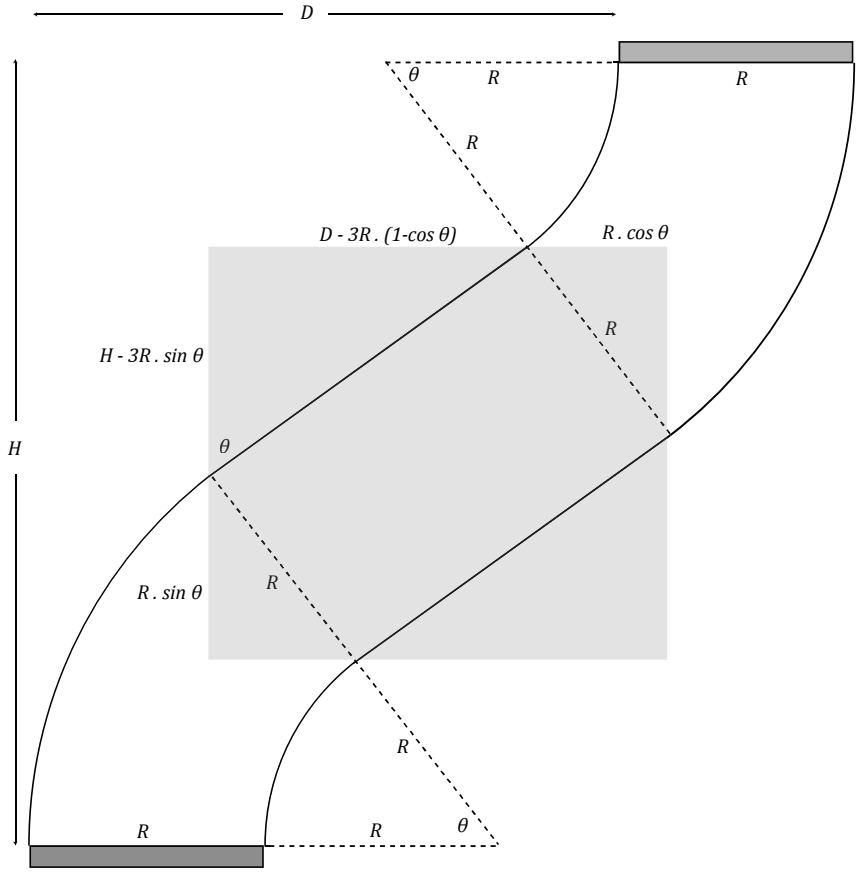

Fig. 72: Adjusting stripe thickness in parallel sets.

The solution[1] to this equation (with $0 < \theta < \frac{\pi}{2}$) is:

$$\theta = \begin{cases} 2.\tan^{-1}\dfrac{\sqrt{D\,(D-6\,R)+H^2}-H}{D-6\,R} & : \quad D \neq 6\,R \\ 2.\tan^{-1}\dfrac{D}{2\,H} & : \quad D = 6\,R \end{cases}$$

1. The solution was simplified from the one computed using the Wolfram Alpha answer engine (accessible at the time of writing this work at www.wolframalpha.com)

Appendix II

Spreading the Nodes in the Entity Wheel

As explained in section 5.2.3.1, the wheel nodes are layouted so that the radial position of the node corresponds to its distinctivity in its sector, no node overlapping occurs, and the node sizes are as large as possible. This is achieved by computing the angular positions for nodes in each sector relative to the sector median, and assigning them initial sizes between minSize and maxSize (depending on the distinctivity). Then the nodes are rescaled in size and in angular position, in order to fit the sectors in the wheel ring.

In the following algorithms $\rho(n)$ and $\theta(n)$ denote the radial and angular coordinates of a node, while $r(n)$ denotes the node size (the radius of the circle enclosing its shape). The algorithms compute these values for each node to satisfy the above mentioned criteria.

Algorithm I – LayoutWheelNodes(wheel)

Step 1: accumulate un-scaled sector angles

$TotalAngle \longleftarrow 0$ *// this variable accumulates the unscaled angles of the sectors*

for all sector $s \in sectors(\text{wheel})$ **do**

SpreadNodesInSector(s) *// compute unscaled start angle and end angles of the sector*

$TotalAngle \longleftarrow TotalAngle + (EndAngle(s) - StartAngle(s))$

end for

Step 2: rescale the computed sizes and angular coordinates of the nodes

$SectorAngle \longleftarrow 0$ *// this variable accumulates the scaled angles of the sectors*

for all sector $s \in sectors(\text{wheel})$ **do**

for all node $n \in nodes(s)$ **do**

$r(n) \longleftarrow \frac{2\pi}{TotalAngle}\, r(n)$

$\theta(n) \longleftarrow \frac{2\pi}{TotalAngle}\, \theta(n) + SectorAngle$

end for

$SectorAngle \longleftarrow SectorAngle + 2\pi\, \frac{(EndAngle(s) - StartAngle(s))}{TotalAngle}$

end for

Algorithm II - SpreadNodesInSector(s)

$P \leftarrow \phi$ *// the set of processed sector nodes*
$StartAngle(s) \leftarrow 0$ *// the start angle of the sector*
$EndAngle(s) \leftarrow 0$ *// the end angle of the sector*
for all node $n \in nodes(s)$ **do**

Step 1: Assign the radial coordinate $\rho(n)$ and size $r(n)$ of the node

$\rho(n) \leftarrow InnerRing + Distinctivity(n).(OuterRing - InnerRing)$
$r(n) \leftarrow minSize + Distinctivity(n).(maxSize - minSize)$

Step 2: Find all angular ranges at radial coordinate $\rho(n)$ occupied by nodes from P

$OccupiedRanges \leftarrow \phi$
for all node $p \in P$ **do**
if $|\rho(n) - \rho(p)| < r(n) + r(p)$ **then**
$\alpha \leftarrow \cos^{-1} \frac{\rho(n)^2 + \rho(p)^2 - (r(n)+r(p))^2}{2\,\rho(n)\,\rho(p)}$
$range = [\theta(p) - \alpha, \theta(p) + \alpha]$ *// range occupied by p at $\rho(n)$*
$NonIntersectingRanges \leftarrow \{r \in OccupiedRanges: r \cap range = \emptyset\}$
$IntersectingRanges \leftarrow \{r \in OccupiedRanges: r \cap range \neq \emptyset\}$
// combine the range, $range$, with all ranges that intersect it into one range:
$combinedRange \leftarrow range \cup \bigcup_{R \in IntersectingRanges} R$
$OccupiedRanges \leftarrow NonIntersectingRanges \cup \{combinedRange\}$
end if
end for

Step 3: Set $\theta(n)$ to be as close to the sector median as possible

// set $\theta(n)$, the angular coordinate of the node, initially to zero (the sector median)
$\theta(n) \leftarrow 0$
// if the sector median falls in a filled range (is occupied by another node), set $\theta(n)$ to be equal to one of the two boundaries of this range (the closer to the median).
for all $range = [start, end] \in FilledRanges$ **do**
if $start < 0 < end$ **then**
$\theta(n) \leftarrow \min(|start|, |end|)$
$StartAngle(s) = \min(StartAngle(s), \theta(n))$
$EndAngle(s) = \max(EndAngle(s), \theta(n))$
break
end if
end for
$P \leftarrow P \cup \{n\}$
end for

List of Figures

Bibliography

[AIS93] R. Agrawal, T. Imieliński, and A. Swami: “Mining Association Rules between Sets of Items in Large Databases”. In *Proceedings of the ACM SIGMOD international Conference on Management of Data,* pp 207-216, 1993.

[B09] S. Borgatti: “2-Mode Concepts in Social Network Analysis”. In *Encyclopedia of Complexity and Systems Science*, *ISBN: 0-38775-888-7*, *Springer Publisher,* 2009.

[D03] P. Domingos: “Prospects and Challenges for Multi-relational Data Mining”. In *ACM SIGKDD Explorations Newsletter 5, 1,* pp. 80-83, 2003.

[D07] S. Diehl: “Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software”. *ISBN: 3-54046-504-9, Springer Publisher*, 2007.

[DL01] S. Džeroski and N. Lavrač: “Relational Data Mining”. *ISBN: 3-54042-289-7, Springer Publisher*, 2001.

[Dž03] Sašo Džeroski: “Multi-Relational Data Mining: An Introduction”. In *ACM SIGKDD Explorations Newsletter 5, 1*, pp. 1-16, 2003.

[DD97] L. Dehaspe and L. De Raedt: “Mining Association Rules in Multiple Relations”. In *Proceedings of the 7th international Workshop on inductive Logic Programming*, pp. 125-132, 1997.

[E00] S. Eick: “Visualizing Multi-dimensional Data”. In *ACM SIGGRAPH Computer Graphics 34, 1*, pp. 61 – 67, 2000.

[EGKNW01] J. Ellson, E. Gansner, L. Koutsofios, S. North and G. Woodhull: “Graphviz - Open Source Graph Drawing Tools”. In *Proceedings of the 9th International Symposium on Graph Drawing,* pp. 483-484, 2001.

[E84] P. Eades: “A Heuristic for Graph Drawing”. In *Congressus Nutnerantiunt, 42,* pp. 149–160, 1984.

[FGW01] U. Fayyad, G. Grinstein and L. Wierse: “Information Visualization in Data Mining and Knowledge Discovery”, *ISBN:1-55860-689-0, Morgan Kaufmann Publishers Inc.*, 2001.

[FPS96] U. Fayyad, G. Piatetsky-Shapiro and P. Smyth: "From Data Mining to Knowledge Discovery: An Overview". In *Advances in knowledge discovery and data mining*, *ISBN: 0-26256-097-6, the MIT Press*, pp. 1-34, 1996.

[F98] M. Friendly: “Extending Mosaic Displays: Marginal, Partial, and Conditional Views of Categorical Data”. In *Journal of Computational Statistics and Graphics, issue 8,* pp. 373-395, 1998.

[FR91] T. Fruchterman and M. Reingold: “Graph Drawing by Force-Directed Placement”. In *Software: Practice and Experience 21/11*, pp. 1129-1164, 1991.

[GGK00] P. Gajer, M. Goodrich and S. Kobourov: “A Fast Multi-Dimensional Algorithm for Drawing Large Graphs”. In *Proceedings of the Graph Drawing Conference*, pp. 211--221, 2000.

[GK06] E. Gansner, and Y. Koren: “Improved Circular Layouts”. In *Proceedings of the 14th International Symposium on Graph Drawing*, pp. 386-398, 2006.

[GN98] E. Gansner and S. North: “Improved Force-Directed Layouts”, In *Proceedings of the 6th International Symposium on Graph Drawing*, pp. 364 - 373, 1998.

[H01] M. L. Huang: “Information Visualization of Attributed Relational Data”. In *Proceedings of the Asia-Pacific symposium on Information visualization*, *9*, pp. 143 – 149, 2001.

[H06] D. Holten: "Hierarchical Edge Bundles, Visualization of Adjacency Relations in Hierarchical Data". In *IEEE Transactions on Visualization and Computer Graphics*, pp. 741-748, 2006.

[Ha06] P. Hanrahan: “VizQL: a Language for Query, Analysis and Visualization”. In *Proceedings of the ACM SIGMOD international conference on Management of data,* pp. 721 - 721*,* 2006.

[HB05] J. Heer and D. Boyd: “Vizster: Visualizing Online Social Networks”. In *Proceedings of the the IEEE Symposium on Information Visualization*, pp. 32-39, 2005.

[HK81] J. Hartigan and B. Kleiner: “Mosaics for Contingency Tables”. In *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, pp. 268-273, 1981.

[HK00] D. Harel and Y. Koren: “A Fast Multi-Scale Method for Drawing Large Graphs”. In *Proceedings of the Working Conference on Advanced Visual interfaces*, pp. 183-196, 2000.

[HKB06] H. Hauser, R. Kosara and F. Bendix: “Parallel Sets: Interactive Exploration and Visual Analysis of Categorical Data”. In *IEEE Transactions on Visualization and Computer Graphics*, pp. 21-29, 2006.

[HM00] I. Herman and M. Marshall: “Graph Visualization and Navigation in Information Visualization: A Survey”. In *IEEE Transactions on Visualization and Computer Graphics*, pp. 24-43, 2000.

[HSPR06] S. Havre, A. Shah, C. Posse, and B. Robertson: “Diverse Information Integration and Visualization”. In *Proceedings of the International Society for Optical Engineering*, 2006.

[J87] J. Johnson: “UCINET: A Software Tool for Network Analysis”. In *Journal of Communication Education*, *36, 1*, pp. 92-94, 1987.

[JJJ08] S. Johansson, M. Jern and J Johansson: “Interactive Quantification of Categorical Variables in Mixed Data Sets”. In *Proceedings of the 12th International Conference on Information Visualization*, pp. 3-10, 2008.

[K00] E. Kandogan: "Star Coordinates: A Multi-dimensional Visualization Technique with Uniform Treatment of Dimensions". In *Proceedings of the IEEE Information Visualization Symposium*, pp. 4-8, 2000.

[K01] D. A. Keim: “Visual Exploration of Large Data Sets”. In *Communications of the ACM, 44, 8*, pp. 38-44, 2001.

[K02] A. Keim, “Information Visualization and Visual Data Mining”. In *IEEE Transactions on Visualization and Computer Graphics*, pp. 1-8, 2002.

[KK89] T. Kamada and S. Kawai: “An Algorithm for Drawing General Undirected Graphs”. In *Information Processing Letters, 31*, pp. 7-15, 1989.

[M67] J. MacQueen: "Some Methods for classification and Analysis of Multivariate Observations". In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281-297, 1967.

[M08] K. Misue: “Visual Analysis Tool for Bipartite Networks”. In *Proceedings of the 12th international Conference on Knowledge-Based intelligent information and Engineering Systems*, pp. 871-878, 2008.

[NMB04] W. Nooy, A. Mrvar and V. Batagelj: “Exploratory Social Network Analysis with Pajek”. *ISBN: 0-52160-262-9, Cambridge University Press,* 2004.

[QE01] A. Quigley and P. Eades: “FADE: Graph Drawing, Clustering and Visual Abstraction”.
In *Proceedings of the 8th International Symposium on Graph Drawing*, pp. 197-210, 2001.

[RSS07] S. Rozsnyai J. Schiefer and A. Schatten: “Concepts and Models for Typing Events for Event-Based Systems”. In *Proceedings of the inaugural international conference on Distributed event-based systems*, pp. 62-70, 2007.

[SRRS07] J. Schiefer, S. Rozsnyai, C. Rauscher and G. Saurer: “Event-driven Rules for Sensing and Responding to Business Situations”. In *Proceedings of the inaugural international Conference on Distributed Event-Based Systems*, pp. 198-205, 2007.

[SZ00] J. Stasko and E. Zhang: "Focus+context Display and Navigation Techniques for Enhancing Radial Space-filling Hierarchy Visualizations". In *Proceedings of the IEEE Symposium on Information Visualization*, pp. 57-65, 2000

[T63] W. Tutte: "How to Draw a Graph". In *Proceedings of London Mathematical Society 13/52*, pp. 743-767, 1963.

[TreeGrid]: Screenshot from COQSoft TreeGrid online demo (retrieved from http://www.treegrid.com).

[V07] A. Vana: "Using Two-mode Network Analysis on Product Ownership to Improve Marketing Communications". In *Proceedings of Applications of Social Network Analysis Workshop*, 2007.

[W97] G. Wills: "NicheWorks - Interactive Visualization of Very Large Graphs". In *Proceedings of the 5th International Symposium on Graph Drawing*, pp. 190-212, 1997.

[WP1] Wikipedia Image (retrieved from http://en.wikipedia.org/wiki/Scatter_plot).

[YFDH01] K. Yee, D. Fisher, R Dhamija and M. Hearst: "Animated Exploration of Dynamic Graphs with Radial Layout". In *Proceedings of the IEEE Symposium on Information Visualization*, pp. 43-50, 2001.

[Z09] C. Ziemkiewicz: "Forked Parallel Sets". *Ongoing project in Charlotte Visualization Center, University of Carolina*, accessible at http://webpages.uncc.edu/~caziemki/ , 2009