

# Visualization of Online Sales Databases

Clickstream Visualization and  
Parallel Coordinated Hierarchies

eingereicht von:  
Alexander Brandstätter

## DIPLOMARBEIT

zur Erlangung des akademischen Grades  
Magister rerum socialium oeconomicarumque  
Magister der Sozial- und Wirtschaftswissenschaften  
(Mag. rer. soc. oec.)

Fakultät für Informatik der Universität Wien und  
Fakultät für Informatik der Technischen Universität Wien

Studienrichtung: Wirtschaftsinformatik

Begutachter:

a.o. Univ.-Prof. Dr. M. Eduard Gröller

Wien, im Februar 2007



## **Eidesstattliche Erklärung**

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.



## Abstract

Information Visualization today offers a wide range of techniques and approaches that can be used to visualize and analyze business data. For the task of visualizing online sales databases of a hardware vendor, this thesis discusses two fields of interest and presents exemplary solutions to certain objectives of analysis. On the one hand it will deal with the visualization of users' interaction in an online shop (*Clickstream Visualization*), presenting means of usage-data collection, its analysis and preparation as well as visualization. Thereby, a prototype is developed that generates node-link diagrams of users' clickstreams. On the other hand this thesis introduces the novel approach of *Parallel Coordinated Hierarchies* – a linked view consisting of a *Hierarchical List* and *Parallel Coordinates* plots for every hierarchy, visualizing sales data like turnover and contribution grouped by attributes like time and location. As the evaluation of the concepts showed, both prototypes' capabilities allowed greater insights in business data than tools already in use and facilitated advanced business analyses to a great extent.



## Abstrakt

Der heutige Stand der Informationsvisualisierung stellt eine Fülle an Mitteln und Techniken zur Verfügung, die für die Visualisierung und die Analyse von Geschäftsdaten herangezogen werden können. Die vorliegende Arbeit beschäftigt sich mit deren Anwendung auf die graphische Darstellung von Online-Verkaufsdatenbanken eines Hardware-Händlers. Dabei werden für zwei ausgewählte Anwendungsbereiche exemplarische Lösungen entwickelt, die sich bestimmten Analysezielen widmen. Zum einen befasst sich die vorliegende Arbeit damit, Benutzerpfade auf den Webseiten eines Online-Shops (*Clickstream Visualization*) darzustellen und zu analysieren. Dabei werden neben der eigentlichen Visualisierung sowohl die Erfassung der Daten als auch deren Analyse und Vorbereitung zur graphischen Darstellung beleuchtet. Der zweite Bereich dieser Arbeit ist die Darstellung und Analyse von Verkaufsdaten – wie beispielsweise Umsatz und Deckungsbeitrag – gruppiert nach Merkmalen wie Zeitraum oder Ort. Dabei wird der neue Ansatz der *Parallel Coordinated Hierarchies* vorgestellt, der die Daten einer Hierarchischen Liste mit deren Darstellung in Parallelen Koordinaten verbindet. Wie die Evaluierung der Konzepte gezeigt hat, ermöglichen beide Prototypen einen tieferen Einblick in die vorhandenen Geschäftsdaten, als dies mit bereits bestehenden Mitteln möglich ist.





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation and Focus . . . . .	2
1.1.1	Company and Business Data . . . . .	2
1.1.2	Objectives . . . . .	4
1.1.3	Focus of this Work and Limitations . . . . .	7
1.2	Outline . . . . .	7
<b>2</b>	<b>Related Work</b>	<b>9</b>
2.1	Clickstream Visualization and Analysis . . . . .	9
2.1.1	Clickstream Information Uses . . . . .	9
2.1.2	Data Collection . . . . .	10
2.1.3	Data Organization and Path Reconstruction . . . . .	13
2.1.4	Visualization Approaches for Website Structure and Usage . . . . .	16
2.2	Parallel Coordinated Hierarchies . . . . .	21
2.2.1	Hierarchical Lists . . . . .	21
2.2.2	Parallel Coordinates . . . . .	22
<b>3</b>	<b>Clickstream Visualization</b>	<b>31</b>
3.1	Motivation . . . . .	31
3.2	Data Collection . . . . .	32
3.3	Path Reconstruction and Data Preparation . . . . .	35
3.3.1	Path Reconstruction . . . . .	35
3.3.2	Data Preparation . . . . .	36
3.4	Clickstream Visualization . . . . .	40
3.4.1	Graph Drawing . . . . .	42

<b>4</b>	<b>Parallel Coordinated Hierarchies</b>	<b>45</b>
4.1	Choice on Hierarchical List and Parallel Coordinates . . . . .	45
4.2	Domain of Interest . . . . .	46
4.3	General Approach . . . . .	49
4.3.1	Systematics of Creation . . . . .	49
4.3.2	Exemplary Displays . . . . .	51
4.4	Interaction and Features . . . . .	52
4.4.1	Section G: General Features . . . . .	53
4.4.2	Section H: Highlighting . . . . .	55
4.4.3	Section B: Brushing and Zooming . . . . .	59
4.4.4	Section C: Changing Hierarchies and Axes . . . . .	63
4.5	Implementation . . . . .	67
<b>5</b>	<b>Evaluation</b>	<b>69</b>
5.1	Interpretation and Evaluation of Clickstream Visualization . . . . .	69
5.1.1	Interpretation . . . . .	70
5.1.2	Functional Evaluation . . . . .	75
5.2	Evaluation Parallel Coordinated Hierarchies . . . . .	77
5.2.1	Test Setup and Methods . . . . .	77
5.2.2	Procedure . . . . .	78
5.2.3	Scenarios . . . . .	78
5.2.4	Current Analysis Tools . . . . .	79
5.2.5	Execution and Outcomes . . . . .	80
5.2.6	Survey . . . . .	83
5.2.7	Test Summary and Conclusions . . . . .	84
<b>6</b>	<b>Conclusions and Future Work</b>	<b>85</b>
6.1	Summary . . . . .	85
6.2	Conclusions . . . . .	86
6.2.1	Clickstream Analysis and Visualization . . . . .	86
6.2.2	Parallel Coordinated Hierarchies . . . . .	87
6.3	Future Work . . . . .	88
6.3.1	Clickstream Analysis and Visualization . . . . .	88

6.3.2 Parallel Coordinated Hierarchies . . . . .	90
<b>Acknowledgments</b>	<b>93</b>
<b>Bibliography</b>	<b>94</b>

## *Contents*

# List of Figures

2.1	Common Log Format [Pro] . . . . .	10
2.2	Subsessions [AGJ <sup>+</sup> 00]. A user-session consisting of four clicks is divided into six subsessions that partly overlap. . . . .	14
2.3	Path Graph [DK01b]. Nodes represent the accumulated traversals, SON is the start of navigation, EON the end of navigation, respectively. . . . .	15
2.4	WebCanvas [CHM <sup>+</sup> 00a]. Clickstreams are clustered according to their similarity. Each line in each cluster is a single user's path. Different pages are color coded. . . . .	17
2.5	Forward oriented, popular paths [DK01b]. Users' popular paths through a website using a hierarchical layout algorithm. Paths traversed more frequently are colored red. . . . .	18
2.6	ClickViz main window [BB01]. Overall flows in the online shop. Clickstreams are separated by gender, where men are colored blue and women are colored red. . . . .	20
2.7	<i>Hierarchical List</i> (Tree List) showing the folder-structure of a filesystem [XCAC05]. . . . .	22
2.8	<i>Parallel Coordinates</i> [ID90]. . . . .	23
2.9	Aggregating data by averaging [Sii00]. . . . .	25
2.10	Curves in <i>Parallel Coordinates</i> [GK03]. . . . .	26
2.11	Extruded Parallel Coordinates [WLG97]. . . . .	28
2.12	Three-Dimensional Parallel Coordinates [WLG97]. . . . .	28
2.13	Higher Order Parallel Coordinates [The01]. . . . .	29
3.1	Exemplary user session. . . . .	36
3.2	Transition Probability Matrix. . . . .	37

## List of Figures

3.3	Overall Transition Probability Matrix. . . . .	37
3.4	Frequent transitions. . . . .	39
3.5	Data analysis execution time. . . . .	40
3.6	User flows in November 2005 . . . . .	41
3.7	Basic structure for edge drawing on the left and an application of the greedy algorithm traversing this structure on the right. . . . .	42
4.1	<i>Parallel Coordinated Hierarchies</i> – default view. . . . .	50
4.2	<i>Parallel Coordinated Hierarchies</i> with expanded hierarchies. . . . .	51
4.3	(G1) Information on axis’ scale via text tooltip. . . . .	53
4.4	(G4) Right click menu for <i>Hierarchical List</i> . . . . .	54
4.5	(H1) Highlight node on “mouse pointer over” in the <i>Hierarchical List</i> . . . .	56
4.6	(H2) Highlight related node on “mouse pointer over” in <i>Hierarchical List</i> . .	57
4.7	(H4) Highlight focussed polylines in <i>Parallel Coordinates</i> . . . . .	57
4.8	(H5) Highlight focussed node and the corresponding polyline. . . . .	58
4.9	(H6) Highlight focussed and related node and polylines. . . . .	59
4.10	(B1) Brushing polylines and nodes according to the mouse-pointer position. .	61
4.11	(B2) Brushing axes segments by mouse-pointer selection. . . . .	62
4.12	(B4) Zooming axes segments. . . . .	63
4.13	(C1a) Exchanging hierarchies by dropping one PCP on another one. . . . .	64
4.14	(C1b) Changing hierarchies by dropping one PCP besides another one. . . .	65
4.15	(C2) Changing axes by dropping one label on another one. . . . .	66
5.1	User flows in March 2005. . . . .	71
5.2	User flows in July 2005. . . . .	72
5.3	Search for Shuttle. . . . .	74
5.4	Answering questions in scenario 1. Answering questions not only takes existing tools but the user’s hand to point out dependencies with current tools (left). On the right, the user performs a change of dimensions’ position (feature C2) to visually display dependencies in underlying data with the prototype. . . . .	80

5.5	User's initial difficulties in scenario 2. Looking for the best performing payment group the user brushes an upper axis region (feature B2) without changing hierarchies (feature C1) first (left). The result is unsatisfying (right): instead of the highest aggregate return-on-investment the list shows the best performers at a monthly aggregation level. . . . .	81
5.6	The right way through scenario 3. On the left the user marks the initial list node in <i>Parallel Coordinates</i> (feature B3). In the middle he zooms regions around the initial data on different axes (feature B4). On the right the result of highlighting similar nodes is displayed. . . . .	82

## *List of Figures*



# 1 Introduction

Scientific visualization and information visualization have the task to generate appropriate visual representations of a given set of data in order to (enable and) support effective as well as efficient data-analysis [SM00]. Thereby, it is an important – and at the same time crucial – aim, to depict facts the way they really are. Only then information visualization allows gaining insights in underlying data, reveals hidden relations and helps understanding and interpreting reality.

To meet these requirements, information visualization today provides a wide range of techniques to display various data at many levels of detail and abstraction, respectively. Finding the approximate display showing relations the best for a given set and kind of data then may become a challenging process – especially if there are several approaches that seem to be suitable. On the other hand, just looking at the databases of an on-line hardware-vendor, one will find a plenty of different data that demand sophisticated analyses: Apart from details on products, customers and suppliers or transactions like purchases and sales, data for example concerning website-usage are stored. Having the visualization techniques on the one hand and domain-specific data on the other, this work will try to merge these two dimensions with the aim to supply domain specific information visualization. Thereby this thesis does not claim to either provide a complete visualization tool for all data from all points of view, nor will it be able to produce the perfect display. What it can and will do is to give exemplary solutions to selected scopes of visualizing certain aspects of online sales-databases.

Before doing so, this chapter will report on the background and motivation for this work in the next paragraphs before giving an overview on the structure of the following sections. Apart from the evaluation sections in Chapter 5 where it was explicitly male

users that were asked for their feedback, any occurrence of the term “user” in this thesis refers to both female and male users respectively, although – for reasons of readability – the text may refer to the user as “he”.

### 1.1 Motivation and Focus

The focus of this thesis is the visualization of online sales databases. The motivation behind this aim is an existing hardware vendor in Vienna who was highly interested in analyzing business-data in a way that goes beyond the scope of statistics concerned for example with the number of units sold or the development of prices. As can be expected, data collected in online sales databases immanently offer a wide range of questions for possible analyses. In order not to exceed the scope of this work and to deepen certain aspects of visual analysis, this thesis will not develop a general approach enabling users to explore all data stored. We will have to concentrate on certain domains of interest for which exemplary solutions will be presented.

The following section will therefore be organized as follows: First, the company itself and its business-data will be introduced to gain further insight in the background and basis of the current work. After that, some possible analyses will be discussed before finally focussing on two different fields of interest.

#### 1.1.1 Company and Business Data

DiTech Daten und Informationstechnik GmbH, in the following referred to as DiTech, is a vendor of computer hardware components and configured systems, selling its products online (via <http://www.ditech.at>) and in its main branch in Vienna as well as in its subsidiaries in Graz, Klagenfurt/Celovec and since September 2006 also in Austria’s largest shopping-mall SCS. Customers are consumers and resellers. In 2005 the online store had about 120,000 average monthly visitors, causing more than a million page requests. The yearly turnover for all stores – online as well as in the subsidiaries – in 2005 is estimated at 24 million Euros, made by selling nearly 500,000 products in more than 130,000 orders.

About one seventh of the orders was shipped to the customers, the rest was picked up in either of the stores.

### Data and Current Analyses

In order to give an overview of the environment the work will set up on, the following section will take a look at the current state concerning the database and its tables as well as current analysis that DiTech already runs.

**Data.** DiTech – referring to the online shop as well as to the four stores – currently offers about 3,000 different products. The main administration of stock was realized with a Paradox database until June 2006, whereas the online shop has always worked with Microsoft<sup>®</sup> SQL Server 2000<sup>™</sup>, the database that in the meantime also is used for main stock administration. The synchronization between the online shop database and the main stock application takes place every ten minutes, concerning goods sold, number of goods in stock and each good’s price. At the moment, the relevant tables in the online shop database hold records as shown in Table 1.1, where table “Orders” stores all shopping-carts that are ordered, “Order Items” holds all items sold in these orders, “Shopping Cart” stores all transactions that did not proceed to checkout, “Pricelist” holds all products currently available, “Pricehistory” stores each product’s price per day (approximately for the last half year), “User” and “Distributor” contain all currently registered users and distributors and finally “Click-Log” stores each click every single user – whether registered or not – performs on the web site.

Apart from the database responsible for administrating the online shop, DiTech runs databases in each of its stores, holding the complete product-, purchase- and sales-data. Depending on whether an order from the online shop is picked up by the customer in either of the stores or shipped, orders are forwarded to the particular store or the main branch in Vienna, respectively. Whatever location, the main databases are all organized in the same tables: Tables for products, for distributors, customers, sales- as well as purchase-transactions, transaction items referencing transactions and products, payment-

## 1 Introduction

Database Table	Approx. Number of Records
Orders	82,000
Order Items	170,000
Shopping Cart	85,000
Pricelist	3,000
Pricehistory	520,000
User	35,000
Distributors	3,200
Click-Log	1,000,000 per month

Table 1.1: *Online store database tables and number of current records*

and shipping-cases. In addition to the price list for the online sales database, the main stock administration manages about 3,000 products at the moment.

**Analyses.** Concerning business-analysis and data-visualization, DiTech at the moment only marginally makes use of data stored. Online statistics mainly concentrate on web site usage – such as the number of page requests or unique IPs per time period – and development of their products’ sales prices. Sales analyses on the other hand are currently run in terms of (partly predefined) database queries aggregating for example main sales characteristics per time period, product groups, payment or shipping methods. Visualization as such – apart from bar charts of website usage and point diagrams displaying development of sales prices – has been left aside so far. More complex analyses, like users’ path through the online shop or content of shopping carts for example, were not made at all, although all data necessary for these analyses exist. These facts, and the conviction that more complex analysis and visualization definitely will add value to running daily business, are motivation and mission to develop the tools presented in this thesis.

### 1.1.2 Objectives

As we have mentioned above, the aim of information visualization and therefore the objective of any approach to be developed, is to enable users to easily explore data visually in certain fields of interest. In order to understand information needs at first, the following paragraphs will discuss potential users’ objectives as well as possible fields of interest – deduced from the target group’s aims of analysis.

- **Marketing and Sales.** Possible aims of analysis for the marketing and sales division may include statistical information on sales, the influence of products' prices on sales as well as analyses about the efficiency of certain marketing activities. Analyzing for example sales per region or sales per time period the efficiency of campaigns may then be assessed.
- **Executive Management.** Executive management on the other hand will possibly be interested in overall turnover or the development of certain product groups for a given time period evaluating executive strategies.
- **Product Management.** In turn, product managers might want to see how systems, that are sold pre-configured, are accepted by costumers or whether their configurations are changed frequently.
- **Online-Shop Engineers.** Looking at products bought together more often than might be expected by chance can be interesting for online shop engineers, as new features – like recommending products that complement articles in the shopping cart – might be integrated. Moreover, displaying the website's traffic might reveal weaknesses concerning navigation or ease of use.

Having these four groups of potential users, five analysis objectives that seemed most interesting have been picked and are discussed in the following paragraphs.

**Characteristics of Sales.** The first field that has to be covered is certain characteristics of sales, taking a look at sales per time-period, per region, per salesperson, per shop or per product. Means of data abstraction – which is aggregating data at different levels of detail – and visual abstraction – showing different levels of details without changing the underlying data – will be considered. To do so, a meaningful set of analyses will be provided, allowing the user to first select a subset – such as sales per time – and then zoom into data for example in terms of product groups or pick-up points. Viewing underlying data has to be provided at all times.

**Products Bought within the Same Transaction.** Secondly, considering the frequency with which two different products are purchased within the same transaction, the application will have to allow analyzing statistical significance of certain product combinations, where concentrating on twovalent coherences here should be sufficient. To get an overview, users could be provided a matrix-based representation, aggregating coherences according to product groups. Clicking on one combination of product groups, a more detailed matrix could show dependencies between single products in this group for example. Further zooming may cause the system to change to a node-link diagram, representing all significant coherences for a certain product, labeling vertices as well as edges. Again, underlying data has to be accessible for all displays.

**Pre-Configured Systems.** As pre-configured systems are sold, it will be interesting to take a look at changes customers make when ordering such a system. If for example many costumers choose a more powerful CPU (Central Processing Unit) than originally provided in the system, it might make sense to change the offered system. Here, appropriate ways of analyses as well as such of graph drawing will have to be found.

**Compatibility of Products Bought within the Same Transaction.** As compatibility of individual hardware components is known, the application may evaluate shopping carts in terms of the compatibility of components bought together. Doing so, hotline inquiries as well as complaints could probably be reduced, and the presentation of products might change. As for the pre-configured systems, an efficient way of analyzing data and visualizing information seems crucial for fulfilling this task.

**Users' Path Through Online Shop.** The last analysis objective that could be covered is concerned with analyzing and displaying users' paths through the online shop. As the shop application logs every single click users perform on the web site, a huge amount of data is collected. Analyzing these data will require an efficient algorithm at first. For visualization on the other hand appropriate graphical representations and abstractions have to be found as the online shop consists of thousands of pages.

### 1.1.3 Focus of this Work and Limitations

Analyzing, engineering, implementing, testing and documenting the five tasks mentioned above may exceed the scope of this thesis. In a discussion with DiTech’s CEO and an informal talk with the main responsible programmer of the online shop we therefore limited the objectives to those two that seemed most interesting – from a viewpoint of analysis as well as considering visualization.

First, the thesis will deal with the analysis and visualization of users’ paths through the online shop. Following Brainerd et al. [BB01] this task will further be referred to as *Clickstream Visualization*. Secondly, we will concentrate on the visualization of sales data, limited to sales per month and the kind of payment. Working on the first task will not introduce new concepts but will be based on approaches that already have been presented. We will apply and discuss certain techniques with data collected from DiTech’s online shop. Visualizing sales data in turn will be done by introducing the novel approach of *Parallel Coordinated Hierarchies*, a linked view that displays hierarchically structured data in a *Hierarchical List* on the one hand, on the other hand makes use of *Parallel Coordinates* to visualize data in different levels of aggregation and provides interaction functionality respectively.

## 1.2 Outline

The rest of this thesis is organized as follows: In Chapter 2, Related Work, existing approaches for both of the tasks will be presented. *Clickstream Visualization* will be split up in its three main phases, namely data collection, data preparation and finally visualization approaches. The state of the art considering *Parallel Coordinated Hierarchies* on the other hand will first introduce the familiar concept of *Hierarchical Lists* in a compact fashion. After that, this section will concentrate on approaches with respect to drawing and interacting with *Parallel Coordinates* and possible conceptual extensions.

## 1 Introduction

Chapter 3, will discuss the concept of *Clickstream Visualization* and its application to data collected by DiTech's online shop. According to the structure of the related work chapter, we will first introduce data collection and preparation as realized in the prototype before turning to data visualization as such.

*Parallel Coordinated Hierarchies* will be introduced in Chapter 4, where the general approach and the prototype's features will be presented.

The evaluation of both the *Clickstream Visualization* and the *Parallel Coordinated Hierarchies* can be found in Chapter 5, where systematics and outcomes of the user studies carried out will be presented.

In Chapter 6 we finally will give an overview on what has been done, draw conclusions on both approaches realized and discuss future work on concepts and prototypes presented for *Clickstream Visualization* as well as *Parallel Coordinated Hierarchies*.



## 2 Related Work

After having introduced and defined the domain of interest as well as the concepts that will be realized, the following section will give an overview on the state of the art considering *Clickstream Visualization* – including data collection and preparation before giving a short introduction on *Hierarchical Lists*, one of the two basic concepts needed for the realization of *Parallel Coordinated Hierarchies*. The last section within this chapter will finally introduce existing approaches considering *Parallel Coordinates*, i.e. interaction techniques and conceptual extensions.

### 2.1 Clickstream Visualization and Analysis

For the task of visualizing users' paths through DiTech's online shop, different solutions and approaches concerning collecting, preparing, analyzing and visualizing logged data exist. The following section will introduce basic concepts, non-commercial approaches presented by the graph drawing community and existing commercial applications that deal with the focus of this thesis. First, approaches from scientific visualization concerning data collection and analysis as well as visualization techniques will be presented. Before doing so, the following paragraph will take a brief look at the motivation for clickstream analysis and visualization.

#### 2.1.1 Clickstream Information Uses

Clickstream data collection and analysis have been done even before the world wide web was introduced, mostly in order to analyze users' behavior in interacting with applications.

## 2 Related Work

According to Guzdial et al. [GSB<sup>+</sup>94] the following main goals are aimed at: First, usability measures can be done, evaluating the effectiveness of a given interface. Secondly, log files are used to characterize usage patterns. Inferring knowledge or expertise is the third aim, where log files are used to show a user's degree of knowledge or expertise about the application he is using. Additionally, Andersen et al. [AGJ<sup>+</sup>00] name the support of automated personalization, the identification of user groups, analyzing banner efficiency and customer interests, or identifying so called "killer-pages" – leading users to terminate their session by exiting a website – as clickstream information uses.

### 2.1.2 Data Collection

Whatever goal has to be reached, the most fundamental part of clickstream analyses may be reliable data collection. Analogous to the basic architecture of a client-server- or a 3-tier-system respectively, collecting data can be done at the server, the client or at an instance in between. Therefore – and reflecting this structure – the following section will give an overview of the most widespread approaches.

**Standard Webserver Access Logs.** Quite commonly, webserver standard access logs are used for analysis of users' behavior [CHM<sup>+</sup>00b, CHM<sup>+</sup>00a, HS99, Die01, DK01b]. A list of tools containing more than 50 commercial and freeware applications is available [Uni]. Webserver standard logs are quite popular in tracking users' site usage because they can easily be created and accessed on the one hand. On the other hand these logs supply a rich set of data.

```
127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326
```

Figure 2.1: Common Log Format [Pro]

The ease of creation and use originates from the fact, that nearly any webserver application provides this function. For example the Apache HTTP-Server Project [Pro] gives an overview on how to create and configure the logging capabilities of the Apache HTTP Server Version 1.3. As shown in Figure 2.1 the data logged for each access in the so called *Common Log Format* contains the IP address of the client (127.0.0.1) and a hyphen for

the fact that the identity of the client is not available. *frank* stands for the user-ID that was authenticated by a log-in event in advance and *10/Oct/2000:13:55:36 -0700* gives the exact time of the request including information on the server's time zone (-0700). "*GET /apache\_pb.gif HTTP/1.0*" is the request itself – in this case the client wanted the *apache\_pb.gif* graphic to be sent (*GET*). *200* is the status code for the request according to the HTTP specification of the W3C *Request for Comment 2616* [W3C], where *200* means that the request was successful. *2326* finally stands for the size of the object in bytes sent to the client. Additional to that, Apache's *Combined Log Format* logs further details on the referrer of a request and the client's application used for accessing information on the internet. Like shown for example by Diebold et al. [DK01b] where an approach for analyzing user interaction is presented in section 2.1.3, reconstructing a user's path with these data works well by sorting them according to IP address or user-ID and according to time an access was registered.

Although this approach can be realized easily, it has some weaknesses. Faisal et al. [FSMB99] point out, that server-sided user-tracking causes the problem, that not all usage is being taken into consideration. Cache hits and proxy hits for example have to be ignored because pages a user has seen before are not requested from the server but supplied by the user's cache or proxy. As a consequence session data may become incomplete as single hits within a sequence are not logged. Concerning reliable logging, further problems caused by the use of proxy-servers were mentioned by Diebold et al. [Die01]. If a proxy does not have a valid copy of a website cached, it may request the site from the webserver. Instead of the actual user's IP-number, the webserver now logs the proxy's IP-number. Another problem arises, if internet service providers (ISP) balance their proxies' load by having more than one proxy handling requests. It is impossible then to reliably tell, what session a click belongs to as consecutive requests from the same user may be forwarded by different proxy-servers – with different IP-addresses. Another problem mentioned by Shahabi et al. [SFKF00] is the accuracy of timespans that are associated with page views. As the webserver cannot exactly know when a user gets a requested website displayed completely, server sided time capturing can only provide information about when a certain side was requested. A more detailed overview on problems with analyzing standard webserver logs has been given by Davidson [Dav99].

## 2 Related Work

**Remote Agent Tracking.** To overcome weaknesses of server-sided tracking, Faisal, Shahabi et al. [FSMB99, SFKF00] propose the employment of user-agent that keeps track of each page’s viewing time, the viewing sequence, likelihood or frequency of a user’s visiting a page, the usage of hyperlinks and an aggregate hit count. Realized as a Java Applet it is send to each user that requests a website. During browsing of the webpage by the user the agent logs all user interactions as specified above. As soon as users leave, a string containing the attributes recorded before is sent to an analysis server. Similarly, Cunha et al. [CBC95] present a modified version of the browsing software Mosaic that keeps track of every user interaction while browsing the internet.

Although client-side logging-approaches provide complete data on how users browse a given set of websites, it goes beyond what is desirable for an online shop. As can be expected, users are not willing to install any kind of agent when they – for example – just want to browse the product catalogue. Moreover a browsing software that constantly keeps track of what the user is doing and reports data may only work within a test environment but – due to privacy concerns – hardly in practice.

**Controller and Proxy Based Logging.** The third kind of logging is presented for example by Winckler and Pontico [WP06], who introduce a *Navigation Controller* for processing user interaction. Whenever a user clicks a hyperlink the controller records all parameters encoded at each transition before handling the actual request. Reconstructing complete paths through a website here is simplified by the fact, that having the navigation controller processing all requests, even browser events like refreshing a page or using the back button are logged.

Similar to this approach, Waterson et al. [WHS<sup>+</sup>02] describe *WebQuilt’s* proxy-based logger. Designed to work within a usability test environment, the proxy-based logger is realized as an intermediary to the client and the server and intercepts each user’s page request. When the server returns the requested content to the proxy, it caches a copy of the content and logs the performed action including links clicked on each page and transaction ordering information for the recreation of the clickstream. Unfortunately, this also only works in a test environment as the use of specific proxy servers that log a single user’s interactions cannot be guaranteed in reality.

### 2.1.3 Data Organization and Path Reconstruction

It is not only logging interaction but also regenerating clickstreams that can be done in certain different ways. As mentioned for standard webserver logs in section 2.1.2, a complete clickstream can be reconstructed by simply sorting logged data by user and time. When no further operations are applied, the retrieved path may contain loops. For the aims of this thesis this basic approach will be sufficient as analyses are restricted to overall flows between certain (sub-)pages and by that to aggregate transition frequencies between single pages respectively, resulting in a 1<sup>st</sup> order *Markov model*. In case more complex analyses are made, paths have to be reorganized as presented for *Fact Schemes* and *Popular Paths* in the following paragraphs.

**Markov Models.** Basically a *Markov model* is defined by three parameters  $\langle A, S, T \rangle$ , where  $A$  is a set of possible actions (here the set of links on every single website),  $S$  is the set of all possible states the model is built for (here all accessible pages) and a  $|S| \times |A|$  transition probability matrix (TPM)  $T$  giving a system's transition probabilities between its different states [DK01a]. Given the state a system is in, the *TPM* in a 1<sup>st</sup> order *Markov model* then gives the probabilities for passing on to certain other states. Respectively, the *TPM* in a 2<sup>nd</sup> order *Markov model* gives the probabilities for transition to other states, given a pair of states the system was in before. Any *TPM* in a  $k^{th}$  order *Markov model* then gives the probabilities for transition in any other state given the last  $k$  states. As this approach is used for predicting a system's respectively a user's behavior, higher-order *Markov models* are widespread. However, for our goals of analysis it will be sufficient to build a 1<sup>st</sup> order *Markov model*.

**Fact Schemes.** *Fact Schemes* were introduced by Andersen et al. [AGJ<sup>+</sup>00], who basically distinguish two different kinds of data-organization for analysis: The *Click Fact Schema* and the *Session Fact Schema*. The *Click Fact Schema* stores every single click with its dimensions URL, date, time, and a session where the session-dimension captures a session key, the user's IP-address, the start and the end-page. The *Session Fact Schema* on the other hand supplies aggregated information in terms of complete sessions instead

## 2 Related Work

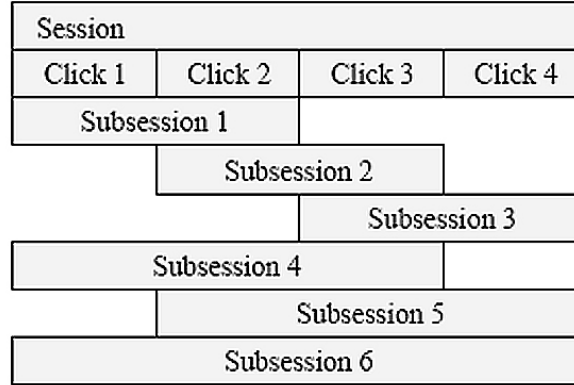


Figure 2.2: Subsessions [AGJ<sup>+</sup>00]. A user-session consisting of four clicks is divided into six subsessions that partly overlap.

of single clicks. Reconstructing a user’s path in the first schema may – similar to the standard logs presented above – be done by (resource demanding) sorting operations on clicks by time. The *Session Fact Schema* on the other hand already stores complete sessions, however leaving aside information on single clicks.

To overcome weaknesses like resource demanding sorting-operations and incomplete information, Andersen et al. [AGJ<sup>+</sup>00] introduce their *Subsession Fact Schema*, where every possible sequence of successive clicks is modeled. As shown in Figure 2.2, a session originally containing four clicks is transformed into six different subsessions that consist of at least two clicks and partly overlap. Analyses concerning users’ paths thereby can be computed easily and in reasonable time. Extracting single clicks on the other hand turns out to be difficult, as all clicks are only stored within their sequences and subsequences. Moreover problems with performance might arise, as the number of subsequences grows exponentially with the number of clicks in a single session. To get this problem fixed, Andersen et al. recommend to use an upper limit for the maximum number of clicks in a subsession, which – from a certain length onwards – limits the growth of subsessions to a linear expansion.

**Popular Paths** In order to investigate the most traveled paths through a website, Diebold and Kaufmann [DK01b] use a maximum forward paths approach [WYB98] to reconstruct a set of paths that all start at the user’s site entry, are unique concern-

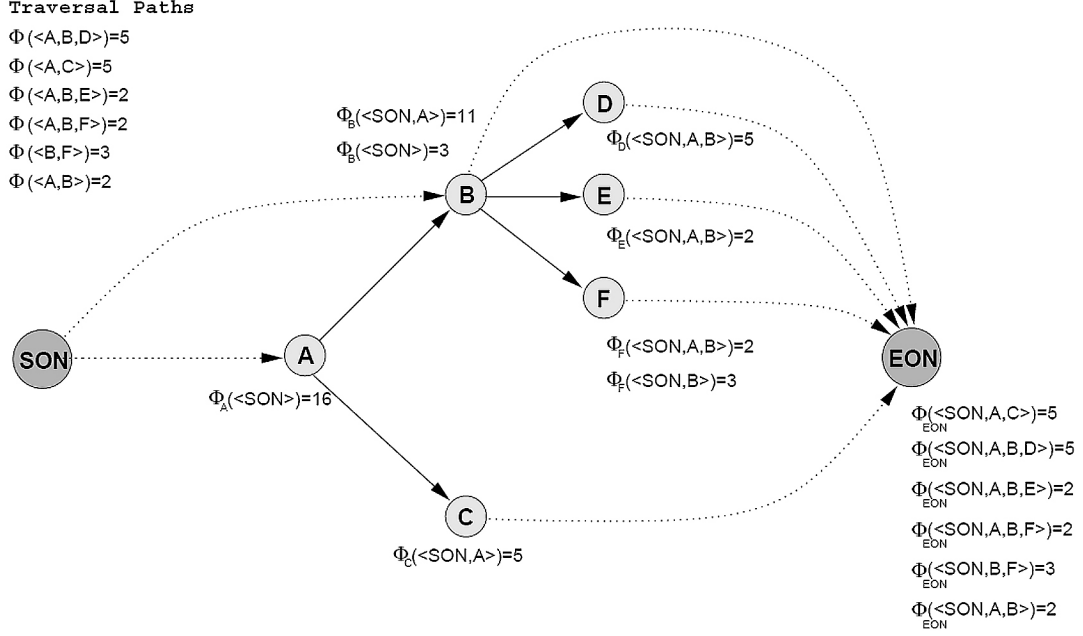


Figure 2.3: Path Graph [DK01b]. Nodes represent the accumulated traversals, SON is the start of navigation, EON the end of navigation, respectively.

ing sequence and number of visited pages and do not have any loops. A traveled path  $P = \langle A, B, C, B, D \rangle$  with nodes  $A$  to  $D$  visited is, for example, then split into two forward paths  $P_1 = \langle A, B, C \rangle$  and  $P_2 = \langle A, B, D \rangle$ . Whenever a path is found that already has been found before, its weight – the number of times a path was traveled – is increased. Having this set of traversal paths, extracting the most popular path then means to first create a path graph that captures the accumulated traversals where each node represents the sum of all paths' weights that enter this specific node. Starting from the absolute entry point (in Figure 2.3 the SON-node), the next node visited is always the one with the highest weight (first node  $A$  in Figure 2.3 with a total weight of 16, from node  $A$  to node  $B$  with a total weight of 11, finally from node  $B$  to node  $D$  with a total weight of 5). A transition to the end-state is made if none of the other potential succeeding nodes has a higher weight than the weight stored for the path between the current node and the end-state. After finishing traversal, the paths' weight is subtracted from all nodes visited including the end state – in our example the total weight is 5, denoted in the end-state. The most popular path has been found. As weights have been adopted, the graph can now be traversed again to find the next most popular path.

## 2 Related Work

Although this approach is one of the best documented [WYB98, Die01, DK01b], it does not seem to be appropriate for our work. First of all, it does neither support self-references nor loops as it concentrates on forward directed paths. For the kind of analysis we will perform, loops are an essential component as they may reveal weaknesses in navigation. As shown above, a sequence  $P = \langle A, B, C, B, D, E \rangle$  is always split into  $P_1 = \langle A, B, C \rangle$  and  $P_2 = \langle A, B, D, E \rangle$ , assuming that it contains two forward directed navigational goals. For our analysis on the other hand, this path could imply for example that either node  $C$  contains important information needed for goals that will be reached later or that the navigational structure as such is misleading. If only forward directed paths are used, this information gets lost.

### 2.1.4 Visualization Approaches for Website Structure and Usage

Concerning mapping and visualization of log-file data, Guzdial et al. [GSB<sup>+</sup>94] distinguish six different classes of displays that are as follows. The first type described is *Scalar* representations, meaning the retrieval of a quantitative value such as expertise level or command frequencies from the log file. The main goal is the performance of a quantitative analysis. The second category listed is *1-D* representations, dealing with the question of what events follow other events. In this context 1-D refers to a one dimensional timeline, on which user events are marked. The next class mentioned is the *1-D Color and Dynamic* representation where, additionally to the *1-D*-class, color and displaying events on the timeline in a dynamic fashion is used. The fourth class presented is called *2-D Abstract*, that includes various abstract two-dimensional representations like simple tables of frequencies for pairs of events or Markov-network charts, displaying Markov-models as presented in Section 2.1.3. *2-D Color and Dynamic*, the last but one representation, adds – just like before – color and dynamics to the basic 2-D-version. This class of representation will be the one used for the application to be developed. The last type of representation includes complex, photo-realistic three-dimensional images and therefore is called *3-D*.

Concerning visualization of website structure and usage, several approaches have been presented by the scientific visualization community. For example *WebPath* [FS98], tries



## 2.1 Clickstream Visualization and Analysis

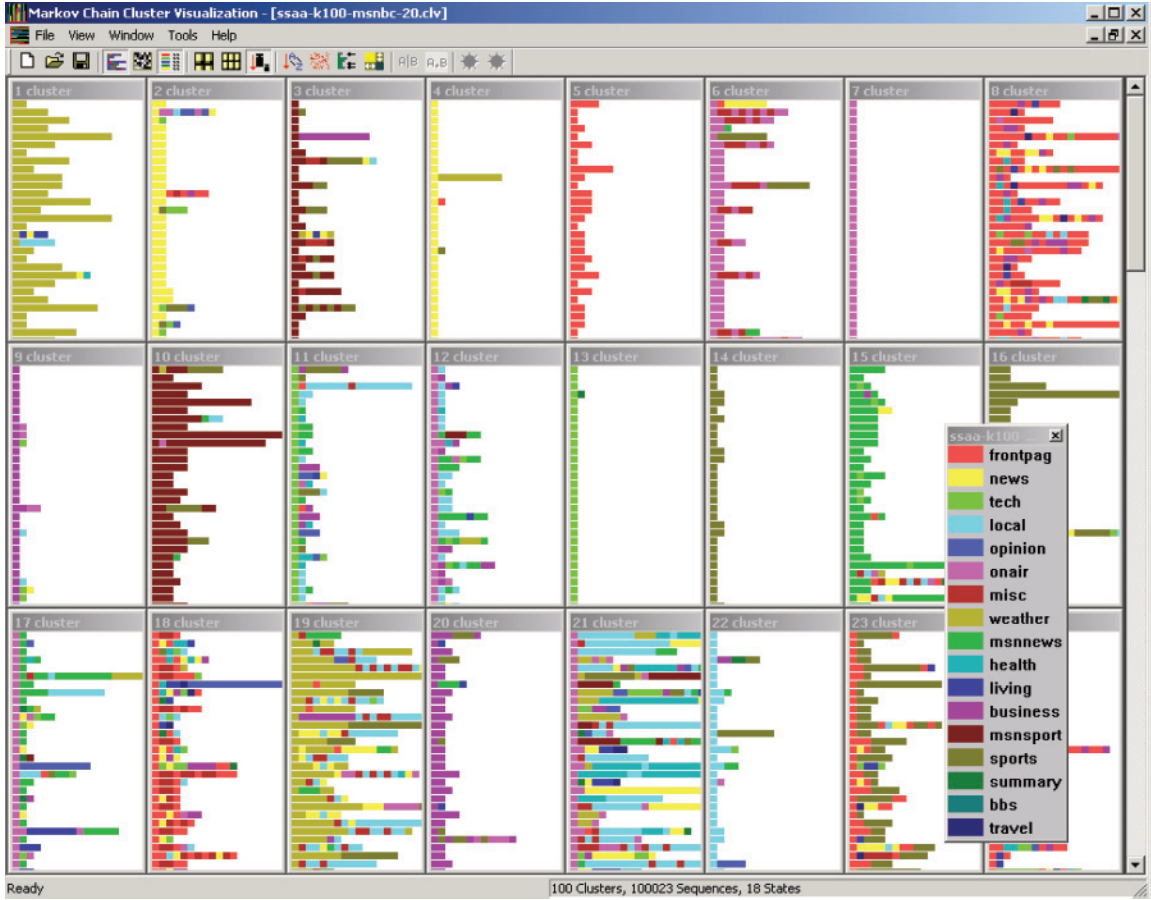


Figure 2.4: WebCanvas [CHM<sup>+</sup>00a]. Clickstreams are clustered according to their similarity. Each line in each cluster is a single user's path. Different pages are color coded.

to improve back-navigation by producing a three dimensional display of a user's browsing history. *WebCiao* [CK97] on the other hand creates 2D node-link graphs of websites in order to visualize a website's structure, track its changes over time and by that help users finding updated or new areas at a site. *WebCanvas* [CHM<sup>+</sup>00a, CHM<sup>+</sup>00b] is intended for website analysis and administration by clustering users' clickstreams by their similarity and visualizing these clusters as a list of sequences of squares that are color coded depending on the specific page requested (see also Figure 2.4). Hochheiser and Shneiderman [HS99] use a Starfield display to depict website usage in terms of a quantitative analysis like accesses per page and directory per time, accesses per host and directory or accesses per referrer and time. Other approaches like presented by Eick [Eic01] show website traffic displayed by colored 3D-bars for each page, encoding the number of pageviews as height and using a radial layout for the positioning of each page.

## 2 Related Work

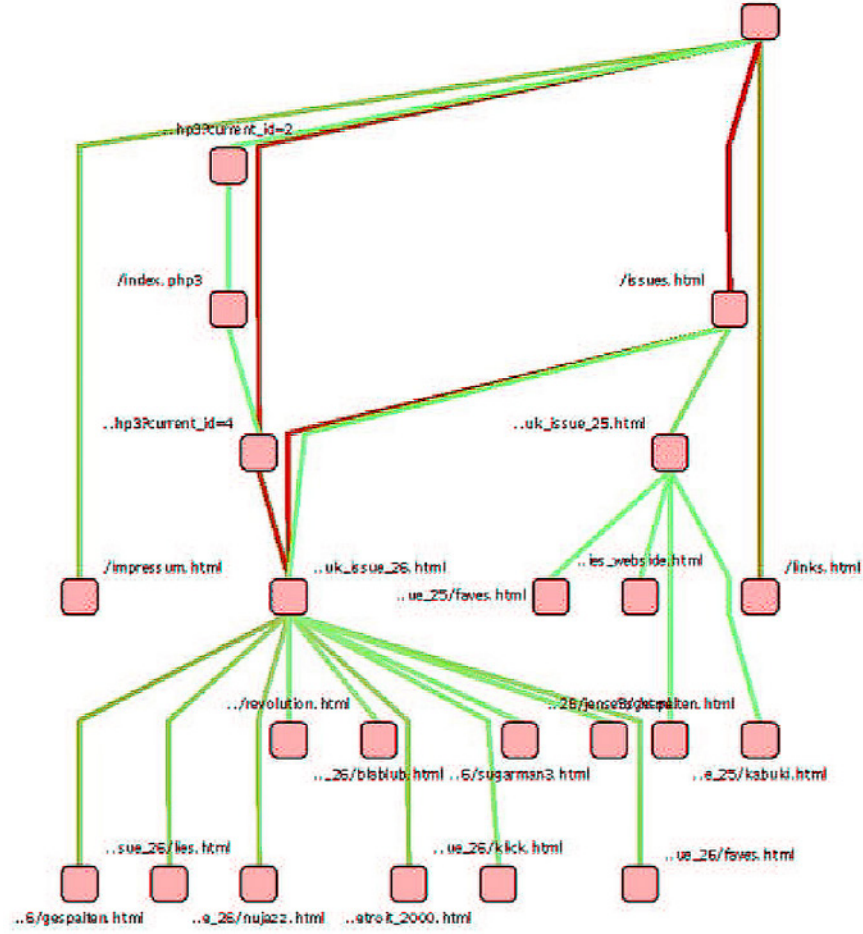


Figure 2.5: Forward oriented, popular paths [DK01b]. Users’ popular paths through a website using a hierarchical layout algorithm. Paths traversed more frequently are colored red.

Approaches related closer to what is the aim of the current work have been presented for example by Diebold and Kaufmann [DK01b] who use a node-link diagram for their *Popular Paths* through a website (see Section 2.1.3 as well as Figure 2.5). Similar to that, Waterson et al. [WHS<sup>+</sup>02] describe their *WebQuilt* visualization system which has been developed for usability testing of a site and uses an edge-weighted layout algorithm that also creates a node-link diagram with the most trafficked path at the top of the display. Just like before, nodes are pages and edges are traversed links, where the thickness of the edges – depicted as arrows to show their direction – is used to encode the number of overall traversals for an edge. Moreover the edges’ color encodes the time spent before traversing this link. Visual path analysis has also been done by Keahey and Eick [KE02].

## 2.1 Clickstream Visualization and Analysis

Besides a *Source/Target Chart* that depicts paths between two predefined nodes, the so called *Leaky Pipe Chart* allows path and flow analysis, displaying all paths between a given set of pages where in both cases nodes represent pages and edges are paths between these pages. Again, the thickness of the edges encodes the frequency a certain link has been traversed.

The most relevant approach for our work has been presented by Brainerd and Becker [BB01]. In their *ClickViz* e-commerce *Clickstream Visualization* data is organized in a series of 1<sup>st</sup> order *Markov models*. As they – among other visualizations – concentrate on overall flows between pages in an online shop, loops as well as self-references are allowed. Nodes – representing pages – are positioned hierarchically depending on their in- and outdegrees, which corresponds to the users’ flow on the website. Edges, again represented as arrows, depict the traffic between these pages, where color encodes any desired attribute (like gender in Figure 2.6). The edges’ thickness represents the aggregate number of traversals. The graph layout itself is generated by a third-party-software from Tom Sawyer Software [Sof].

## 2 Related Work

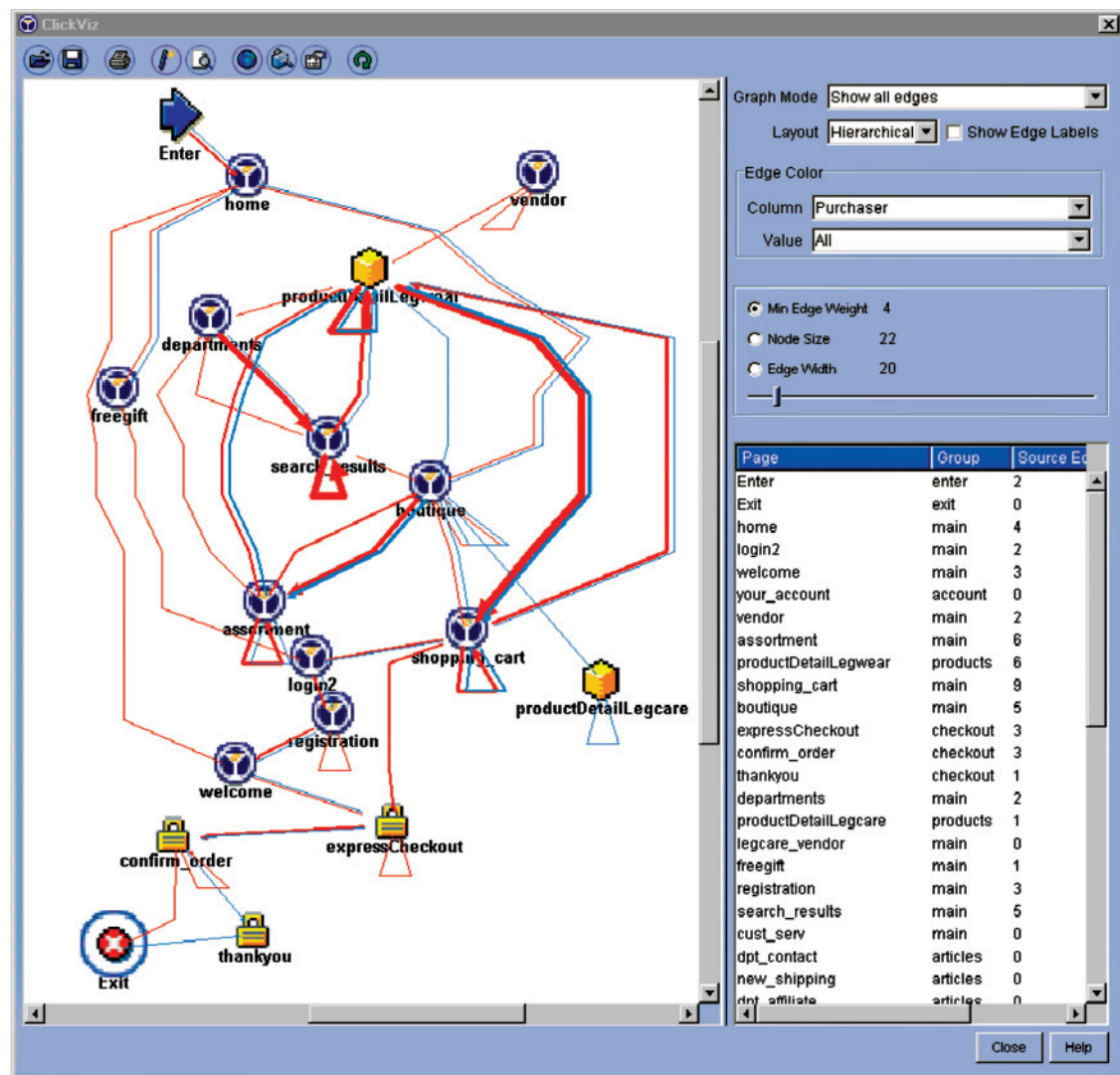


Figure 2.6: ClickViz main window [BB01]. Overall flows in the online shop. Clickstreams are separated by gender, where men are colored blue and women are colored red.

## 2.2 Parallel Coordinated Hierarchies

For the second approach presented in this thesis – the *Parallel Coordinated Hierarchies* – mainly two concepts are needed to create the linked view discussed later: a *Hierarchical List* and *Parallel Coordinates*. The following section will therefore give an overview on what these techniques are about and will try to capture the state of the art considering both approaches. First a short introduction to *Hierarchical Lists* will be given before discussing *Parallel Coordinates*.

### 2.2.1 Hierarchical Lists

According to Xiang et al. [XCAC05], *Hierarchical Lists* are the most frequently used tree structure visualization. Although not well documented in literature, most hierarchical list-views follow the convention that nodes either belong to exactly one parent node or a root, respectively. *Hierarchical Lists* are often referred to as *Tree Lists*, emphasizing the data’s hierarchical tree structure. The concept became very popular in context with displaying file-systems on early window-based operating systems, later also as an approach to organize overview maps of websites. For an example of a folder-structure of a filesystem see Figure 2.7.

One reason, why *Hierarchical Lists* are not well documented in literature – and also why they are chosen for this work – is their intuitive use. *Tree Lists* moreover provide an efficient way of organizing and searching data: while looking up data in an ordered list has to be done searching top to bottom, organizing data hierarchically by certain attributes reduces the cost of searching to a logarithmic extent, as only the desired attribute in each hierarchy has to be found. Thirdly, data used in the field of interest is – to a wide extent – organized in hierarchies like year and month. Besides different levels of aggregation (for example a year containing twelve months) the *Hierarchical List* also provides aggregated values for each hierarchy. This is also why DiTech already makes use of this kind of representation in some of its analysis tools.

## 2 Related Work

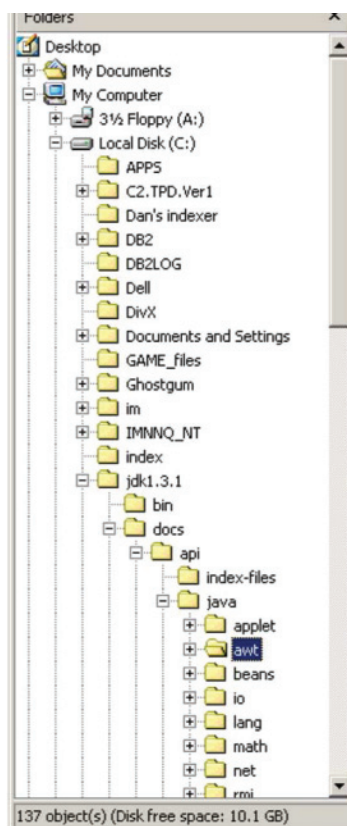
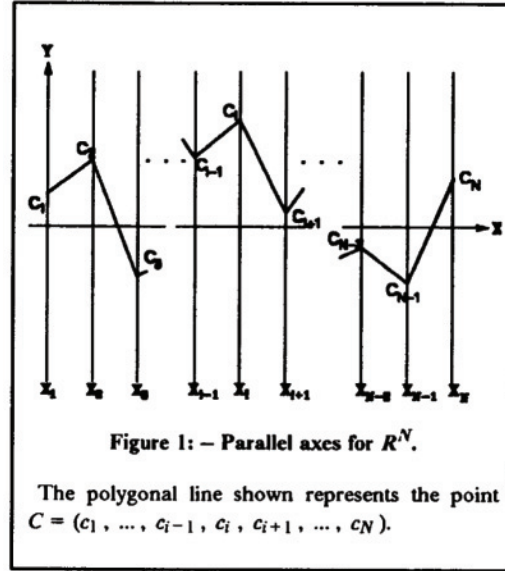


Figure 2.7: *Hierarchical List* (Tree List) showing the folder-structure of a filesystem [XCAC05].

### 2.2.2 Parallel Coordinates

*Parallel Coordinates* [ID90] have first been introduced by Alfred Inselberg and Bernard Dimsdale. They are designed to display multidimensional data in a planar representation. The main motivation behind *Parallel Coordinates* is, that data do not just have two or three dimensions which can be depicted using conventional coordinate systems but may encapsulate multiple attributes that define multiple dimensions. To display these multiple dimensions in two-dimensional planes, Inselberg and Dimsdale [ID90] presented in 1990 their approach of *Parallel Coordinates*, that maps points in  $R^N$  to polylines as follows: On a plane with xy-coordinates, starting from the y axis a variable number of copies of the y-axis is placed equidistantly and orthogonal to the x-axis, where each dimension has its own copy of the y-axis. Displaying a data item – consisting of multiple variables (which is dimensions) – in the *Parallel Coordinates* system then is just drawing a polygonal line (*polyline*) intersecting each axis at the point corresponding to the value of the

Figure 2.8: *Parallel Coordinates* [ID90].

item's attribute. Any  $N$ -dimensional point  $C$  with coordinates  $(c_1, c_2, \dots, c_N)$  therefore is represented by a polyline connecting the positions of  $c_i$  on their respective axes. For an example see Figure 2.8. Having more than just one polyline displayed in a *Parallel Coordinates* plot, relations between data may be detected: Apart from simple qualitative relations within one axis, *Parallel Coordinates* clearly show distributions and outliers in each dimension. As mentioned for example by Wegenkittl et al. [WLG97], *Parallel Coordinates* can also be used to show that multidimensional points reside on the same line in  $R^N$  if their polyline representations intersect each other at specific points between the vertical axes. This characteristic turned out to be useful in preventing collisions in airtraffic for example. Correlations between dimensions may be pointed out by concentrating on the lines' intersections with the axes: If lines cross two axes in the same order, a positive correlation can be assumed, crossing axes in opposite order shows negative correlations. Attributes are then indirectly proportional.

## Limitations

As for many visualization techniques, readability as well as efficiency of *Parallel Coordinates* plots suffer when large data-sets are displayed. Novotny [Nov04] names three major

## 2 Related Work

problems high density displays are confronted with:

- First, a loss of speed and interaction that is noticeable when displays become more populated.
- Secondly, occlusion may happen, meaning that two or more visual elements overlap, making it impossible to recognize single items.
- Thirdly, having high density displays, the viewer is confronted with aggregation in a way that objects are drawn over each other. The actual number of objects then cannot be perceived any more.

### Interaction

Basic approaches for efficient working with *Parallel Coordinates* were for example discussed by Martin and Ward [MW95], who present different brushing-methods, allowing the definition of brushes that for example support logical operations. Siirtola [Sii00] presents the so called Parallel Coordinates Explorer, that among other things mentioned below, supports aggregating polylines by certain characteristics on one axis by averaging their values on all axes. For an example see Figure 2.9, where the upper image shows conventional brushing of cars data by country of origin (the axis on the far right) through color coding. In the lower image data is reduced to one average car for each manufacturer country, giving further information on the distribution of the original data.

The cars data set is also used by Hauser et al. [HLD02] to depict their concept of *angular brushing* and *smooth brushing*. With angular brushing users may define a subset of slopes between two axes to brush the respective points, whose lines have the defined angle between the two defined axes. As shown for the cars data set for example, only a few values have negative slopes between the dimensions Horsepower and Acceleration, showing, that most cars do better with relatively less power. Smooth brushing on the other hand takes those values into consideration that definitely reside in the area defined for brushing. Moreover, it uses a variable percentage to accommodate a smooth gradient of values at the borders.



## 2.2 Parallel Coordinated Hierarchies

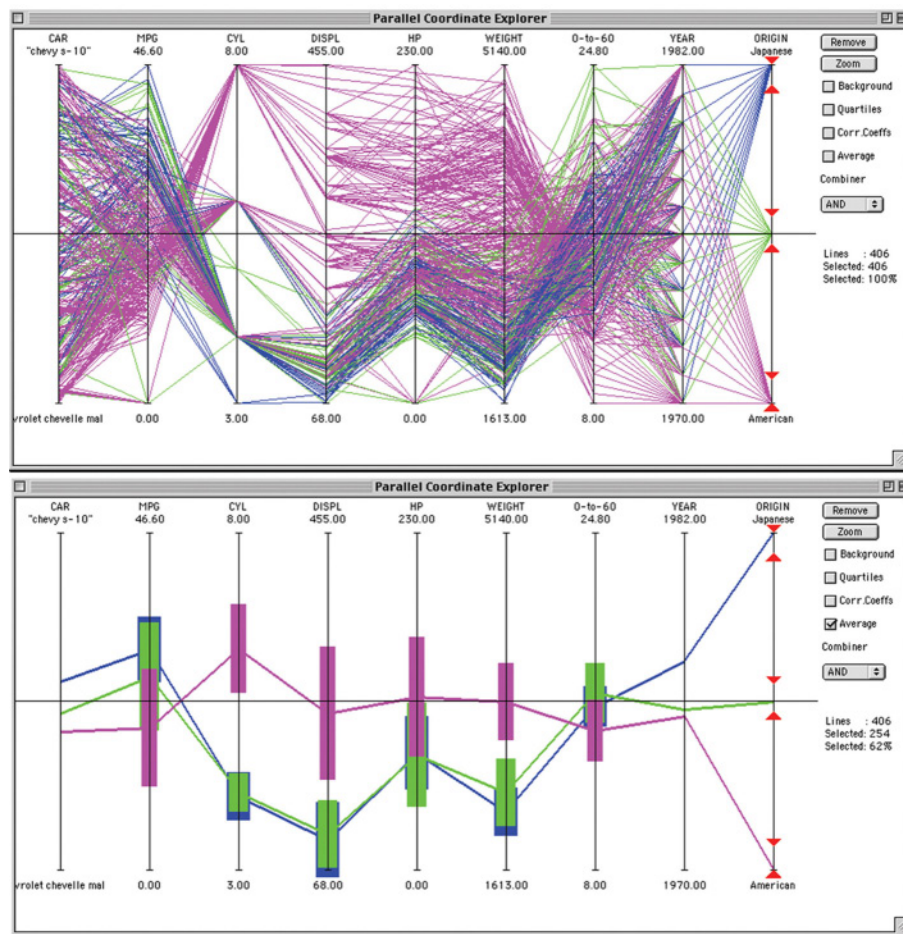


Figure 2.9: Aggregating data by averaging [Sii00].

Brushing with respect to value-occlusion was introduced by Wegman and Luo[WL96, Weg99], who use saturation to depict overlapping lines: When brushing a certain region, lines usually are highlighted with a specific color, ignoring overlapping regions. In contrast to that, instead of pure coloring, the line segment’s saturation here is changed whenever a segment belongs to a polyline that is brushed. As a result, regions with overlapping, brushed lines appear higher saturated, whereas line segments that are brushed as well but do not overlap are seen less intense.

### Line Drawing Extensions

Just like mentioned in paragraph Limitations, aggregation and occlusion may prevent viewers from recognizing separate polylines in high density plots. To overcome this weakness, Novotny [Nov04] adds visual abstraction to the *Parallel Coordinates* plot by creating clusters of polylines displaying similar samples as areas that allow overviewing data at first. Detailed information on the other hand – that is not visible due to too many lines here anyway – cannot be extracted from these plots. Similar to that, Fua et al. [FWR99] present their approach of Hierarchical Parallel Coordinates that tries to reduce the amount of overplotting. Data is again organized in clusters, the resulting bands are visualized using varying translucency, where the center of each cluster is displayed with highest opacity that fades out towards the clusters’ borders. Yet another approach for clustering and visualizing data with *Parallel Coordinates* was presented by Johansson et al. [JLJC05], who use different colors and intensity levels to depict overlapping polylines within clusters. In addition to the solutions presented above, clusters’ variance (the values spread around the expected value) or skewness (the degree of asymmetry in a distribution) can be animated. Moreover different transfer functions (i.e. linear, logarithmic, exponential), responsible for coloring the clusters, may be used and defined.

Problems of occlusion and aggregation also motivated the suggestion to use curves instead of straight lines in *Parallel Coordinates* as presented by Graham and Kennedy [GK03], facilitating visual separation of single lines. For an example of this concept see Figure 2.10, where in the standard *Parallel Coordinates* plot on the left lines are merged. In contrast to that lines on the right still can be separated visually. The main disadvantage of this concept may be, that the information on intersections between axes is lost, which for example means, that one cannot reliably tell, whether points reside on the same line in  $n$ -dimensional space.



Figure 2.10: Curves in *Parallel Coordinates* [GK03].

## Functional Display Extensions

Extensions to the original *Parallel Coordinates* display were presented for example by Ong and Lee [OL96] (see also [SM00], p. 187), who suggest the use of histograms that are painted over each axis and give further information on the distribution of each dimension's values. Hauser et al. [HLD02] take this idea and extend the display by semitransparent areas surrounding the axes that emphasize the histogram representation of the values' distribution. Moreover, statistical means were implemented by Siirtola [Sii00], which allow displaying quartiles as a Box Plot or correlation coefficients. To overcome weaknesses of axes regions with highly dense intersections, Graham and Kennedy [GK03] suggest to make use of empty space on axes. They introduce a bounding box that covers an affected region and allows rescaling the respective axis by spreading the axis segment within the box, facilitating visual separation of curves. In order to remind the viewer of the different scale, the respective bounding box is framed with a border.

## Conceptual Extensions

Apart from different fields of use and interaction, the graph drawing community has also presented some approaches concerned with the conceptual extension of *Parallel Coordinates*. For example Wegenkittl et al. [WLG97] introduce *Extruded Parallel Coordinates*, where the coordinate system is moved along the third spatial axis and points originally residing on the same axis are connected. Resulting from this, a complex surface – like shown in Figure 2.11 – is created, allowing further insights into the underlying data. Another approach presented by Wegenkittl et al. [WLG97], are so called *Three-dimensional Parallel Coordinates*. Instead of having parallel vertical axes, it is always two variables that define conventional coordinate systems on planes. Combining these planes in three dimensional space and additionally connecting the values that belong to the same data item then gives the visualization shown in Figure 2.12, where planes are arranged orthogonal on the left hand side while multiple planes are combined into the display in the right.

## 2 Related Work

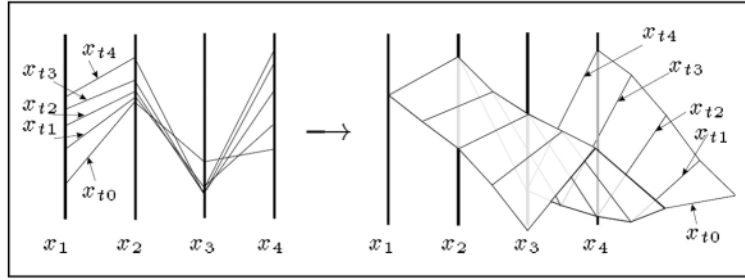


Figure 2.11: Extruded Parallel Coordinates [WLG97].

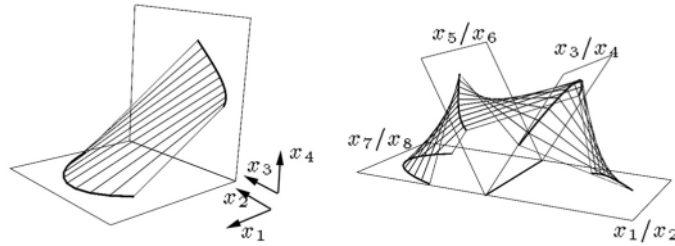


Figure 2.12: Three-Dimensional Parallel Coordinates [WLG97].

In his approach of *Higher Order Parallel Coordinates*, Theisel [The01] introduces additional axes that may be placed between two existing axes. Lines originally connecting two points on two axes in a straight fashion are deflected by the item’s value on the respective axis placed in between. Lines therefore are not straight anymore but – depending on the factor of deflection – curved. For an example of *Higher Order Parallel Coordinates* see Figure 2.13, where again the cars data set was used. Instead of straight lines (Figure 2.13.a) now curves connect points (Figure 2.13.b). As can be seen, correlations between triplets of dimensions (for example on the right “Displacement”, “Horsepower” and “Cylinders”) may now be found.

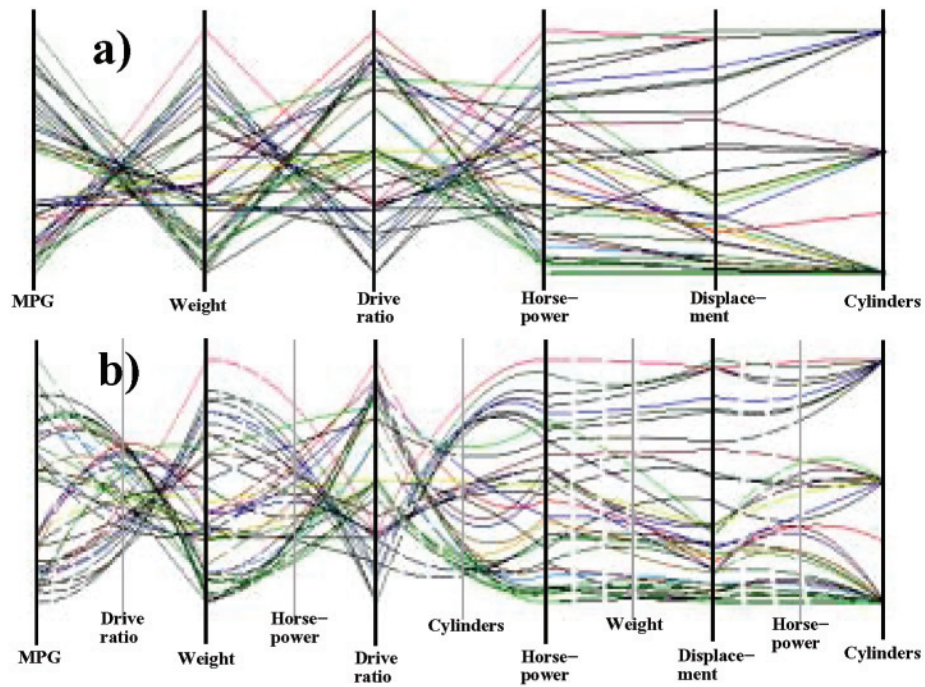


Figure 2.13: Higher Order Parallel Coordinates [The01].

## *2 Related Work*

## 3 Clickstream Visualization

After having presented the state-of-the-art considering *Clickstream Visualization* as well as *Parallel Coordinates* and *Hierarchical Lists*, we now present the main part of the thesis, that introduces the approaches realized. The first chapter will be dedicated to *Clickstream Visualization*, where data collection and preparation will be described and an overview on the developed prototype's capabilities is given. The next chapter will introduce the *Parallel Coordinated Hierarchies* approach, discussing general motivation and display setup as well as all features realized. In Chapter 5 we will finally describe interpretation and evaluation of both prototypes and its results.

### 3.1 Motivation

The first concept that was examined in this work is clickstream analysis and visualization. Although only 14% of DiTech's sales are handled without the customer ever going to any of the stores (by ordering online and having the products shipped), the vast majority of sales is initiated online. It therefore seemed interesting to investigate what pages users are visiting, what flows exist between pages, where there is heavy traffic, and whether a flow map of DiTech's online shop could reveal weaknesses in the site's structure, navigation and features. As stated in Section 2.1.4, the work of Brainerd and Becker [BB01] was used as a model for this section. As neither a commercial nor a freeware tool explicitly implementing their concept could be found, an own prototype has been developed. The following section will give an overview on concepts and solutions realized. Analogous to the related work section, data collection and preparation for analysis will be pointed out first, followed by an overview on current functional capabilities of the prototype.

## 3.2 Data Collection

The data used for the prototype is collected server-sided. Although a webserver standard access protocol exists, DiTech additionally logs user interactions in a way similar to the *Navigation Controller* presented by Winckler and Pontico [WP06]. As the main parts of the website consist of ASP-Pages, clicks on links are forwarded to the server with certain attributes. Whenever a request reaches the ASP-framework, it generates the required HTML code of the requested website and writes the transaction including the attributes click ID, eventcode, eventparameter, session ID, and a timestamp to a database table, where these attributes are defined as follows:

- **Click ID.** Each click has a unique, seven-digit integer key that is generated by a sequence triggered automatically whenever a new click is registered in the database.
- **Eventcode.** Each click registered is related to a certain class of pages, that are grouped by either their content or technology respectively. At the moment nine different groups of pages exist, that have been listed below.
- **Eventparameter.** In order to precisely know what content was requested, an *eventparameter* is logged for each access. Depending on the class of pages registered in the *eventcode* attribute, the *eventparameter* for example stores the filename of the static HTML page requested or the unique identifier of the product viewed in detail.
- **Session ID.** The *session ID* attribute (written to the database as *sid*) logs a string containing the actual session ID, the time the session was created, the user's IP address, the date the session was created and a referrer in case the user was redirected to DiTech's online shop from a site that runs for example a banner-campaign. As the character *P* is used as a separator the exemplary session ID *2783605P59P33193P154P18P9818P01P2005fromgeizhals* can be separated into the ID *27836*, the time the session was created *05:59:33* and the IP address the request comes from, *193.154.18.98*. Moreover the creation date *18.01.2005* and the referrer – *from geizhals* – is encoded in the session string. In addition to identify users in the analysis, the session ID is used to enable users to save their shopping carts for



later checkout. In order to finish purchasing, users can come back to the shop via a link stored in their browser's bookmarks and immediately continue their session. The session ID then is used to retrieve saved carts from the database.

- **Timestamp.** Finally, the *timestamp* attribute (written to the database as *zeitstempel*, the German word) saves the exact time a click is performed with an accuracy of a second as well as the current date. The format used is *dd.mm.yyyy hh:mm:ss* which for example is *18.11.2006 14:57:03*.

#### Logged Clickevents

In order to be able to generate a flow map of website usage, one of the most important attributes logged is the eventcode as presented above. At the time of realizing the prototype, the nine following different codes were used just like listed below, reflecting different groups of requests:

- **bannerclick.** A *bannerclick* is logged whenever a user clicks a banner displayed at the site. Possible eventparameters are for example `http://www.x1800x1.com/` or `openkat_pcsys.aspx`, giving further information on the target of the link.
- **bookmarked\_wk.** The code *bookmarked\_wk* is logged if a user comes back to the website via a link stored in his browser's bookmarks in order to checkout a previously saved shopping cart he saved in advance.
- **html.** Whenever a static HTML page is requested, the server logs the code *html*. Possible eventparameters are for example `/graz/angebote.htm` or `/HTML/kontakt.htm`, giving further information on the content requested.
- **openkat.** DiTech's online shop is organized in a wide range of product categories and subcategories that are accessible through a vertical, hierarchically structured menu. Whenever a user clicks one of the offered menu items, an entry is created in the database having *openkat* as eventcode and the respective category identifier as eventparameter.

### 3 Clickstream Visualization

- **orderclick.** An article-detail page always displays a linkbutton which adds this very product to the personal shopping cart. Doing so generates an *orderclick* entry in the usage log having the article identifier as eventparameter.
- **search.** As the website provides a search function, sending queries to the server is logged with the eventcode *search*, having the string to be searched for as eventparameter.
- **titelseiteclick.** A so called *titelseiteclick* is logged when a user clicks one of the links displayed at the main entry page (Titelseite in German) of the website.
- **view.** Viewing articles in detail (before for example ordering them) causes a *view* entry, having the article number as eventparameter.
- **zubehoerclick.** A so called *zubehoerclick* finally is registered whenever someone makes use of the links showing accessories (Zubehör in German) to a certain product viewed in detail.

Unfortunately, the online shop does not log checkout events. Therefore, one cannot reliably tell, whether a user ever really checks out a shopping cart. To do so, data logged would have to be compared to orders that are registered in the sales system. As this would be too resource demanding, no information concerning this matter can be withdrawn from the prototype.

#### Reliability

Logging website usage by an intermediary – like the Microsoft<sup>®</sup> .NET-Framework used here – has the advantage, that client-sided caching can be controlled. This facilitates achieving reliable logging of users' sessions by simply not allowing to access cached content but to forward every request to the intermediary. And indeed: Evaluating the behavior of the cache in Microsoft<sup>®</sup> Internet Explorer Version 6.0.2900.2180 while browsing the online shop showed, that repeated access to the same content – with some restrictions pointed out below – was handled by the server.

Although the online shop uses caching, a repeated access to an article detail view for example caused the browsing client to cache a new version of the website he received by the server. What can be concluded from this is, that each access was forwarded to and handled by the server – no matter whether the requested page had been cached before. At the same time this guarantees, that the intermediary framework has generated an entry in the log database respectively. On the other hand, using the browser’s refresh button caused the server to send a new version of the requested content as could be expected. However, contents were assigned a new session ID which definitely causes systematic problems when analyzing users’ paths through the website. Using the browsers back button in contrast does not create a new session but a cache hit, that unfortunately is not registered by the server.

For our log’s reliability this means: Whenever the browser’s back-button is used, the registered clickstream is not valid anymore as intermediary pages are missing. Using the refresh button on the other hand creates a new session – which is even more undesirable. Looking at the site’s structure, we may assume that refreshing contents does not bring any advantage – as for example the vertical menu is reduced to the highest hierarchy after refreshing – and may therefore not be used too frequently. Cache hits – occurring when using the back-button – on the other hand will have to be accepted as it seems too resource demanding to exclude those sessions from analysis, that may have been interrupted.

## 3.3 Path Reconstruction and Data Preparation

After having discussed data collection in the previous section, we will now turn to path reconstruction and data organization. First, this section will look at exemplary data before discussing preparation of data for analysis.

### 3.3.1 Path Reconstruction

As pointed out for standard webserver access log in Section 2.1.2 a user’s path can be reconstructed by sorting data by *session ID* and *timestamp*. Figure 3.1 shows an exemplary

### 3 Clickstream Visualization

ID	Eventcode	Eventparameter	sid	zeitstempel
858235	openkat	124	1000215P58P3080P108P150P11113P01P2005from	2005-01-13 15:58:53.220
858242	view	FI16SA	1000215P58P3080P108P150P11113P01P2005from	2005-01-13 15:58:58.610
858246	orderclick	FI16SA	1000215P58P3080P108P150P11113P01P2005from	2005-01-13 15:59:03.563

Figure 3.1: Exemplary user session.

user session extracted from the database by applying these sorting criteria where *sid* is the *session ID* and *zeitstempel* the *timestamp*, including the attributes as presented in Section 3.2. We can tell from these data that the user in the first place reached the online shop by typing the URL in the address bar of his webbrowser as no referrer is logged following the substring *from* in the *session ID*. Taking the sequence of events but also the time spent at the website clearly shows, that the user exactly knew, what he wanted: only 5 seconds after opening category 124 he chose the detail-view for a specific product identified by FI16SA (a 160 GiB harddisk). Only another 5 seconds later he decided to add the product to his shopping cart. Unfortunately recordings end at this point, as the checkout process is not logged.

#### 3.3.2 Data Preparation

As mentioned in Section 2.1.4 concerning the work of Brainerd and Becker [BB01], data preparation for the prototype's visualization is done building a *1<sup>st</sup>-order Markov Model* (also see Section 2.1.3, Markov Models), where the *Transition Probability Matrix (TPM)* is a  $i \times i$  matrix defined by the nine possible eventcodes plus two possible states for site entry and an end state. The *TPM* for March 2005 considering all sessions logged in this period is shown in Figure 3.2: The values of each row add to 100 percent, which is due to the fact, that being in a certain state, a transition will definitely take place. The diagonal values are colored red to emphasize self-references. The last rows's values are undefined because the end state cannot be left. Any value contained in the *TPM* (and the table in Figure 3.2 respectively) gives the probability for a transition from the state denoted by the row to the state denoted by the column.

The *TPM* therefore provides appropriate values for predicting the user's next step. For direct visualization and the goals of analysis in this thesis on the other hand the *TPM* cannot be used. As it does not cover overall traffic, denoted as global probability for the

### 3.3 Path Reconstruction and Data Preparation

	partner	Start	bannerclick	bookmark	html	openkat	orderclick	search	titelseitecl	view	zubehoercli	end	SUM
partner	0.00000	0.00000	0.15754	0.03063	8.44128	22.40067	0.00438	11.51760	9.98162	47.46630	0.00000	0.00000	100.00000
start	0.00000	0.00000	0.21983	0.05468	14.28884	58.89539	0.00984	12.75442	11.73401	2.04298	0.00000	0.00000	100.00000
bannerclick	0.00000	0.00000	1.19926	0.00000	44.74170	49.63100	0.00000	0.46125	0.09225	0.92251	0.00000	2.95203	100.00000
bookmark	0.00000	0.00000	0.41876	31.82579	5.11390	28.30621	0.00000	2.51256	0.25126	17.00168	0.00000	13.56784	100.00000
html	0.00000	0.00000	0.38811	0.10396	34.68190	22.61071	0.38811	3.24347	1.40689	10.93458	0.16980	26.07249	100.00000
openkat	0.00000	0.00000	0.05156	0.03380	1.21402	54.45754	0.01628	0.70532	0.05649	33.66054	0.00321	9.80123	100.00000
orderclick	0.00000	0.00000	0.04519	1.22001	3.08495	65.79445	0.06572	7.33651	0.35327	6.07131	0.05340	15.97519	100.00000
search	0.00000	0.00000	0.08131	0.01459	2.62916	8.54009	0.03127	45.05234	0.16680	28.87286	0.01042	14.60114	100.00000
titelseitecl	0.00000	0.00000	0.29670	0.00873	11.95951	26.24460	2.17723	2.78372	32.64541	0.82464	0.58467	22.47480	100.00000
view	0.00000	0.00000	0.08597	0.07849	3.86543	26.43101	7.32376	1.48014	0.36941	46.43872	0.51300	13.41407	100.00000
zubehoercli	0.00000	0.00000	0.10121	0.10121	3.18626	36.63967	0.05061	5.97166	1.51822	38.61336	3.34008	10.47571	100.00000
end	undef	undef	undef	undef	undef	undef	undef	undef	undef	undef	undef	undef	undef

Figure 3.2: Transition Probability Matrix.

transition between two (not necessarily different) states, we have to build a  $i \times i$  *Overall Transition Probability Matrix (OTPM)* that contains transition probabilities

$$p_{ij} = \frac{\sum_t t_{ij}}{\sum_t t}$$

Now  $p_{ij}$  is the probability for transition to state  $j$  given state  $i$ ,  $\sum_t t_{ij}$  denoting the absolute number of all transitions from state  $i$  to state  $j$  and  $\sum_t t$  denoting the absolute number of all transitions between all possible states. Corresponding to the example before, Figure 3.3 shows the *Overall Transition Probability Matrix* for all sessions logged in March 2005. In contrast to Figure 3.2, the values here are given in tenth of a percent to better detect more frequented paths. Another difference is, that the last row's values are not undefined but zero, meaning that there is an overall probability of exactly zero that any transition starts in the end-state. Moreover, it now is not the single row's sum that adds up to 100% but the sum over all fields.

	partner	Start	bannerclick	bookmark	html	openkat	orderclick	search	titelseitecl	view	zubehoercli	end
partner	0.00	0.00	0.04	0.01	1.93	5.13	0.00	2.64	2.29	10.87	0.00	0.00
start	0.00	0.00	0.20	0.05	13.09	53.97	0.01	11.69	10.75	1.87	0.00	0.00
bannerclick	0.00	0.00	0.01	0.00	0.49	0.54	0.00	0.01	0.00	0.01	0.00	0.03
bookmark	0.00	0.00	0.01	0.38	0.07	0.34	0.00	0.03	0.00	0.20	0.00	0.16
html	0.00	0.00	0.22	0.06	20.06	13.08	0.22	1.88	0.81	6.32	0.10	15.08
openkat	0.00	0.00	0.21	0.14	4.93	221.21	0.07	2.87	0.23	136.73	0.01	39.81
orderclick	0.00	0.00	0.01	0.30	0.75	16.05	0.02	1.79	0.09	1.48	0.01	3.90
search	0.00	0.00	0.04	0.01	1.26	4.10	0.02	21.65	0.08	13.88	0.01	7.02
titelseitecl	0.00	0.00	0.07	0.00	2.75	6.03	0.50	0.64	7.50	0.19	0.13	5.16
view	0.00	0.00	0.28	0.25	12.44	85.04	23.56	4.76	1.19	149.41	1.85	43.16
zubehoercli	0.00	0.00	0.00	0.00	0.06	0.73	0.00	0.12	0.03	0.76	0.07	0.21
end	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Figure 3.3: Overall Transition Probability Matrix.

**Calculation.** Calculating these figures was realized as follows: A set of timely ordered sessions is loaded. First, the referrer of the *session ID* attribute is analyzed on whether the user came to the website via a link at a partner's website or by typing the website's URL

### 3 Clickstream Visualization

in his browser's address bar. Secondly the session's first registered *eventcode* is read. The number of transitions between the identified starting state and the state denoted in the first eventcode is incremented by one. At the same time the overall number of transitions is incremented. Next, the subsequent entry's eventcode is read, incrementing the number of transitions between the current and the next state as well as the number of overall transitions by one. After that, the algorithm changes to the next state that now becomes the current state. These steps are repeated until the end of a session is reached. Finally the number of transitions from the last registered state to the (virtual) end state as well as the number of overall transitions is incremented. In the end, going through all sessions gives an array containing the absolute number of transitions performed between the states as well as the number of overall transitions. To get the *Overall Transition Probability Matrix* these numbers have to be divided by the number of overall transitions. As already mentioned in the paragraph before, the resulting *OTPM* for March 2005 can be seen in Figure 3.3.

**Threshold.** Moreover, in order to limit displayed transitions to those that were performed with a certain frequency, a threshold had to be defined in first place before the matrix was displayed like presented in the following section. In the example for March 2005 (shown in Figure 3.3) 111 of the overall possible 144 transitions were performed at least once. Displaying this *OTPM* directly, the resulting graph would be confusing and misleading, as one could not distinguish between clicks performed only once in a month and those performed a 100 times more often – which still was not very frequent. As could be found out in informal tests, a threshold of only 0.7% seemed sufficient to still cover 92.6% of all clicks in only 20 different transitions, like shown in Figure 3.4. Here, the transitions' overall probabilities in tenths of a percent are highlighted black if their value is more than 0.7%. Fields for clicks with a frequency less than 0.7% of all transitions are zero, no matter what the actual probability was. Labels for eventcodes that neither were logged as starting point nor as end point – in any transition having a probability of more than the defined threshold – are colored gray.

Having a 0.7% threshold, still 92.6% of all transitions and 6 of the original 9 eventcodes logged are included in the sample. Allowing a threshold of only 0.3% added the *book-*

### 3.3 Path Reconstruction and Data Preparation

	partner	Start	hammerclick	bookmark	html	openkat	orderclick	search	titelseiteclick	view	zuhauseclick	end	Threshold 7%	92.6%	Percentage of all transitions
partner	0	0	0	0	0	0	0	0	0	0	11	0			
start	0	0	0	0	0	13	54	0	12	11	0	0			
hammerclick	0	0	0	0	0	0	0	0	0	0	0	0			
bookmark	0	0	0	0	0	0	0	0	0	0	0	0			
html	0	0	0	0	0	20	13	0	0	0	0	0			
openkat	0	0	0	0	0	0	221	0	0	0	157	0			
orderclick	0	0	0	0	0	0	16	0	0	0	0	0			
search	0	0	0	0	0	0	0	0	23	0	14	0			
titelseiteclick	0	0	0	0	0	0	0	0	7	0	0	0			
view	0	0	0	0	0	12	85	24	0	0	149	0			
zuhauseclick	0	0	0	0	0	0	0	0	0	0	0	0			
end	0	0	0	0	0	0	0	0	0	0	0	0			

Figure 3.4: Frequent transitions.

*marked\_wk* event to the 0.7% sample, where the *OTPM* now covered 96.6% of all clicks, containing 29 different transitions. On the other hand, being more restrictive and raising the threshold to 2.0%, only 5 of the original 9 eventcodes were displayed, covering only 79.5% of all clicks in only 10 transitions. For the kind of analysis we want to perform, it therefore seemed reasonable to use a threshold of 0.7% which excludes the least frequent transitions but still covers the vast majority of flows.

**Performance Issues.** Having about 120.000 monthly visitors performing about 1 million registered clicks, data preparation as described above is heavily resource demanding. As can be seen from Figure 3.5, analyzing all data from 2005 takes more than 15 minutes on a modern desktop computer. In order to limit execution to a reasonable time, computing flows in a commercial system will therefore be restricted to a sample containing for example only half of all sessions logged. As could be shown for March 2005 the flow values just deviate up to 3% when only 50% of the sessions are used for analysis. Still, computation time is about 4 minutes. Further restriction to only 10% of all sessions logged would bring an additional (tremendous) gain of time: in a test carried out with a tenth of all data the system only took 19 seconds to execute data preparation. Data reliability on the other hand suffers: taking only 1% of all data for calculation, values deviate up to more than 28%. For the graphs produced by the prototype therefore a more extended calculation time was accepted for more reliable results.

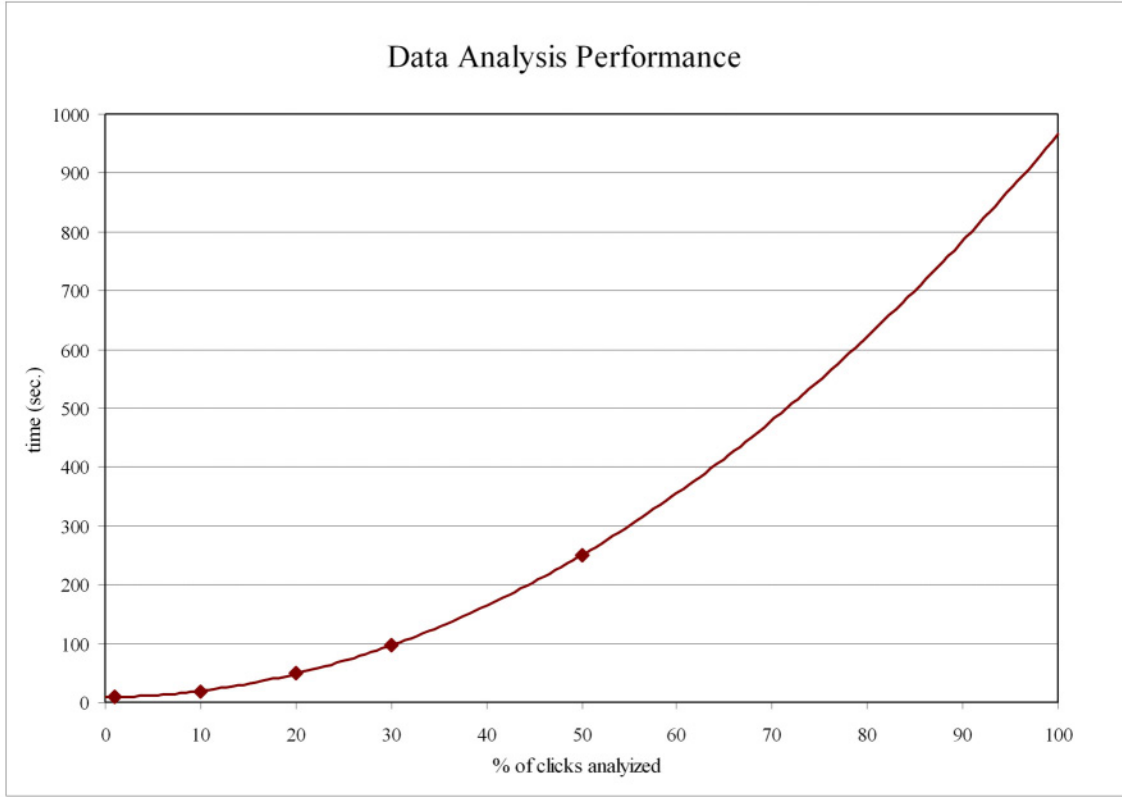


Figure 3.5: Data analysis execution time.

## 3.4 Clickstream Visualization

Having data collected and prepared as described in the sections above, the resulting *Overall Transition Probability Matrix* can now be visualized directly. As can be seen from Figures 3.6, just like in Brainderd and Becker's work [BB01], a node-link display was used to point out users' flows between pages. At the top of the display, the site's entry nodes are placed, where *URL* stands for every user coming to the online shop by typing the website's URL in his browser's address bar and *Partner* denotes all clients that are forwarded by any partner site. At the bottom of the site the (virtual) end state is placed, logged after the last (actual) click registered. In between the start nodes and the end node, nodes depict pages visited and are labeled according to the *eventcode* logged as described in Section 3.2. Each outgoing edge represents the over all probability for a transition between the two connected nodes, where frequencies are denoted by the thickness of edges. The nodes' color – in this first approach – is determined by the position in the display (for the weaknesses of this solution see the User Evaluation Section 5.1), the edges'



color on the other hand always is determined by the color of the node they leave. Self references are denoted by triangular connections and show the relative frequency, a certain transition was carried out after the same transition was performed before. For further interpretation of displays generated by the prototype, see Section 5.1, where exemplary usability problems are discussed on the basis of two different months. Moreover this section will present feedback collected in an informal talk about the generated displays with the person responsible for programming and maintaining the online shop of the hardware vendor, the data was taken from.

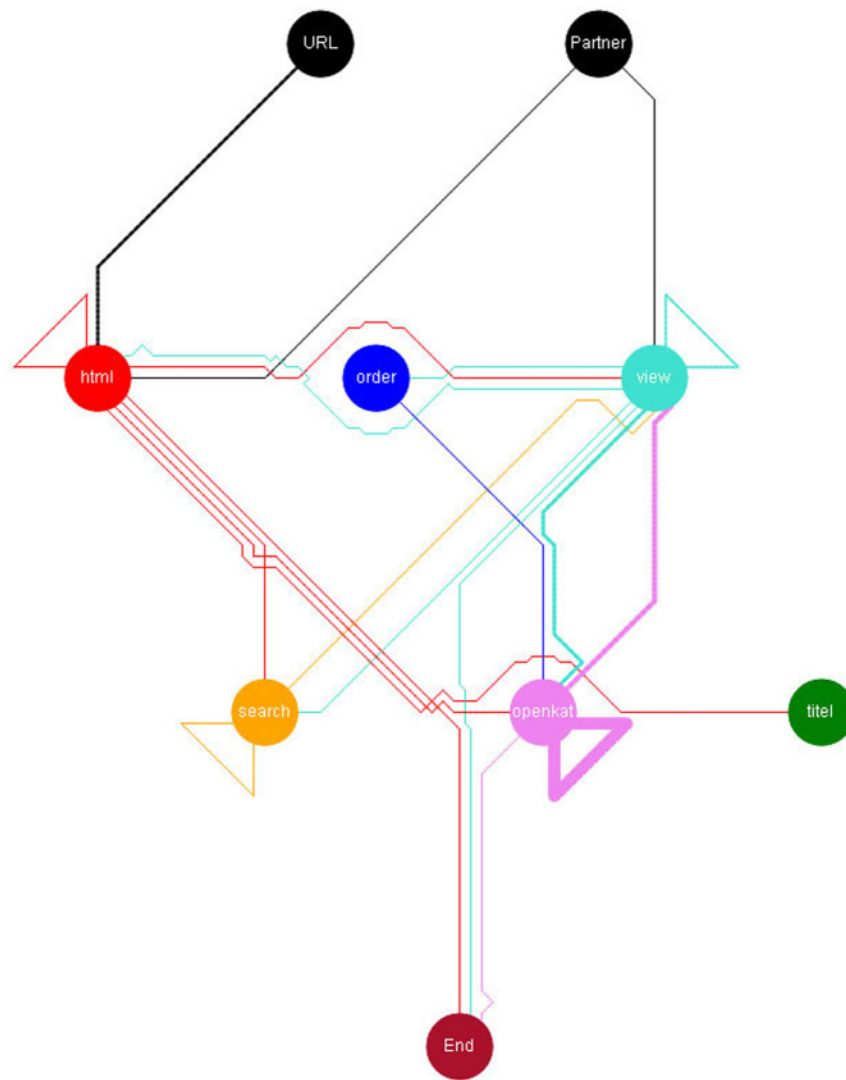


Figure 3.6: User flows in November 2005

### 3.4.1 Graph Drawing

After having presented the prototype’s visualization approach for users’ flows on DiTech’s website, the following paragraphs will now discuss basic graph drawing strategies. As the source code is not part of this thesis, we will thereby just give an overview on the approaches applied by the prototype.

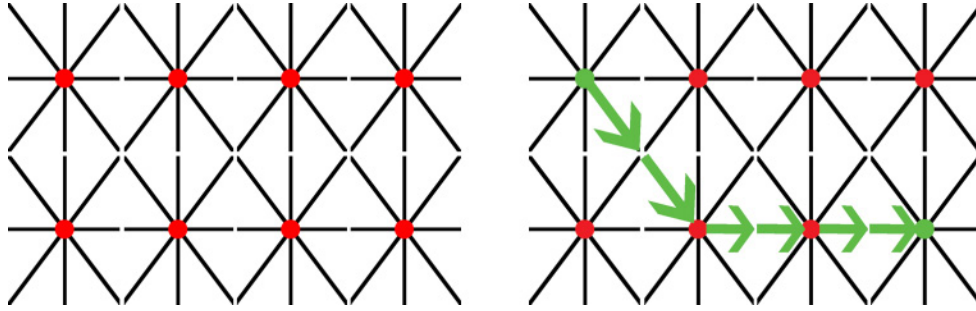


Figure 3.7: Basic structure for edge drawing on the left and an application of the greedy algorithm traversing this structure on the right.

**Node Placement.** Just like in the work of Brainerd and Becker [BB01], the nodes’ position in the prototype is determined by a hierarchical structure. Nodes with higher out-degree are placed in the upper row of the display, nodes with higher in-degree in the lower. Within each row, the more connected nodes are in the middle, the less connected ones at the side. Once placed, the nodes position – in this early version – is not optimized concerning edge-crossings.

**Edge Drawing.** Edges are painted using a greedy algorithm on a fixed underlying structure that is depicted on the left of Figure 3.7. First, the display is separated into rectangular areas, each having a central point that is connected to all four corners as well as to the middle of all four edges. These areas form a grid, allowing vertical, horizontal and diagonal line drawing with a change in direction in the central points of the areas. Each of an area’s eight lines can be marked according to whether it is used already, has been visited before, is locked for any reason or whether it can be passed. Connecting two points basically works like shown on the right of Figure 3.7: The algorithm starts at the top left point and first tries to overcome vertical and horizontal distances by diagonal moves in

the target's direction. Once the horizontal level of the target node is reached, it turns left and directly follows the horizontal lines from node to node. When the algorithm has found the target, it goes back every edge it marked as traversed and marks it as used. When the initial node is reached again, a path between the source and target has been found. In case the algorithm encounters an edge used or locked that was needed for the direct path to the target, it tries side steps: First, it goes back for another try to cross the edge. Whenever it takes a step back, the already used edge is marked as locked as it cannot be used to reach the desired target.

Together with some other conflict solving strategies, this basically is sufficient to paint and connect nodes in the presented visualizations, as it only has to deal with 9 nodes and about 20 edges. For high-density displays this algorithm may not provide appropriate solutions as it neither cares about crossing reduction nor optimized node placement.

### *3 Clickstream Visualization*

## 4 Parallel Coordinated Hierarchies

The second concept this work will present is *Parallel Coordinated Hierarchies*, a novel approach of a linked view combining the ease of use of a *Hierarchical List* and the visualization and interaction capabilities of *Parallel Coordinates*. First, this chapter will discuss the choice of using a *Hierarchical List* and *Parallel Coordinates*, next it will present data used for the prototype. After that, the approach itself will be presented, followed by introducing implemented features, before giving an interpretation of some aspects of possible analyses. The systematics and outcomes of the user evaluation, carried out with DiTech’s managing director, will be discussed hereafter in Chapter 5.

### 4.1 Choice on Hierarchical List and Parallel Coordinates

The use of a *Hierarchical List* for displaying data is due to different circumstances itemized below.

- Business data used may easily be organized hierarchically. Be it turnover or contribution ordered by year, month, location, and – depending on the desired domain of interest – by payment or shipping method, whatever data has to be analyzed may be organized in these types of “natural” hierarchies.
- A *Hierarchical List* is a space saving way to display a great amount of data that reduces the costs for searching to a logarithmic complexity.
- Any tree structure can provide data in different levels of aggregation, at each hierarchy.

## 4 *Parallel Coordinated Hierarchies*

- DiTech already uses *Hierarchical Lists* to display business data in different domains for different statistical analyses which seemed interesting with respect to offering an extended version for direct comparison to existing tools in evaluation (Section 5.2).

*Parallel Coordinates* on the other hand were chosen because of their ability to display many dimensions within a compact, two-dimensional plane.

- Business data – in our case consisting of attributes like turnover, contribution, return-on-investment and number of items sold – then can be visualized within one single display.
- Compared to scatterplots this means that it will be sufficient to use one display for each hierarchy depicted with *Parallel Coordinates*, where six two-dimensional scatterplots would be necessary, combining each dimension with all others.
- *Parallel Coordinates* enable the viewer to detect outliers, correlations and the values' distribution in each dimension, facilitating visual data analyses over several dimensions.
- Interacting with *Parallel Coordinates* is intuitive and can be applied to several dimensions within the same display.

## 4.2 Domain of Interest

As already mentioned above, it were business data that were used for the present proof of concept. In order to make the prototype comparable to tools already in use, we decided to discuss the concept with data that – concerning structure, hierarchies and dimensions – are generated for DiTech's current sales statistics. As a reasonable number of hierarchies should be available, aggregated sales data grouped by payment groups and methods were chosen.

**Data Structure.** In detail, data used for the prototype is aggregated by merging database tables related to sales from DiTech's branches in Vienna, Graz, Klagenfurt/Celovec and

from the shop in Austria’s biggest shopping mall in the south of Vienna. The desired level of detail is limited to monthly values for every location as the objective of this analysis is gaining an overview on monthly developments in the four dimensions turnover, contribution, return-on-investment and number of products sold, where the respective values are defined as follows:

- **Turnover (T)** Turnover is calculated as the sum of all products sold times each product’s sales price before taxes. The prototype will refer to it as “Umsatz”, the German word for turnover.
- **Contribution (C)** Having a product’s purchase and sales price, the contribution is the difference of sales and purchasing price. The aggregated contribution given for every dimension then is the sum of all products times their sales prices (which is the turnover) minus the amount paid to buy these products, which is the sum of all products times their purchasing price. Again, the prototype will use the German word for contribution which is “Deckungsbeitrag” (DB).
- **Return-on-investment (ROI)** The return-on-investment – referred to as “ROI” in the prototype – is the sum of what is added to the purchasing price over all products sold in percent of overall purchasing expenditures. Expressed in terms of turnover and contribution the return-on-investment is given by  $ROI = \frac{C}{T-C}$ .
- **Number of Products Sold.** The last dimension used for analyses is the number of products sold in respective hierarchies. It seems interesting to include this attribute in analyses, because every single unit causes overhead costs for example for storage or packaging. The prototype will refer to this dimension as “Stück”, the German word for unit.

Using fictitious values, a database row for our four dimensions in February 2006 in the payment group “Assignment”, payment method “Assignment within 14 days” in Vienna then looks like in Table 4.1, where database rows here have to be depicted as columns in order not to exceed the page’s capacity.

The four dimensions presented above are headed by values for year, month, payment group and method as well as location. In contrast to the dimensions, these five attributes

#### 4 Parallel Coordinated Hierarchies

<b>Year</b>	2006
<b>Month</b>	02
<b>Payment Group</b>	Assignment
<b>Payment Method</b>	Assignment within 14 days
<b>Location</b>	Vienna
<b>Turnover</b>	128,150.10
<b>Contribution</b>	12,560.23
<b>ROI</b>	10.87
<b>Units</b>	1,562

Table 4.1: A typical database record for sales per time, payment method and location.

cannot be aggregated meaningfully by summing them up but may be arranged in a hierarchical order as they define navigational paths within the dimensions. Year and month thereby form a “natural” hierarchy, as any year consists of months. Payment group and payment method on the other hand are only virtually hierarchical, as any payment method per definition belongs to exactly one payment group – that is introduced only to facilitate navigation. A change in hierarchies therefore does not make sense here, whereas changing the hierarchies year and month does – as any month is included in more than just one year. Finally, location can be seen as hierarchy as it exists just like the other hierarchies independently from any of the dimensions.

Creating a tree structure out of a table containing values like the presented above, starts with sorting values in the desired hierarchical order. After having done so, the table can be processed top to bottom, creating new nodes whenever a new identifier is found in the respective hierarchy. The values of the three dimensions turnover, contribution and units thereby have to be summed up and assigned to each node as the sum of its respective children’s values. In contrast the ROI dimension has to be calculated for every node as it is a ratio.

**Data Supplied.** In general, the prototype on the one hand supports test data that were generated randomly for three years (2004, 2005 and 2006), each only containing four months. Five payment groups – each holding five payment methods – organize data from three different locations for all periods. On the other hand, with regard to the user test carried out with the prototype (see Section 5.2), also real data from the sales database



was pre-aggregated and used in the prototype. Thereby real data only contain values for January to September 2006 as former years were stored in an archive database that was not accessible. On the other hand the months October to December were not available yet at the moment data was retrieved. Both, test and real data were pre-aggregated to enhance the prototype's performance. Retrieving data live from all locations and aggregating them to the desired table takes already about two minutes only for 2006 and was left aside therefore. Moreover, to proof the concept, real data covering nine months seemed sufficient. Finally, the test data contained 899 records, where real data only consisted of 180 records.

## 4.3 General Approach

After having presented the data, its hierarchies, dimensions and aggregation as well as its volume, we will now turn to the *Parallel Coordinated Hierarchies* approach and by that to the implemented prototype. The main idea behind this approach is the connection of a *Hierarchical List* and *Parallel Coordinates*.

### 4.3.1 Systematics of Creation

As we have seen in Table 4.1, each database row consist of nine fields, where five are used as hierarchies (year, month, payment group, payment method and location) and four as dimensions (turnover, contribution, return-on-investment and number of goods sold). The first approach – and the one already realized by DiTech – is to organize data hierarchically in a *Hierarchical List* (see Figures 4.1 and 4.2). Thereby, the dimensions' data as stored in database records form the lowest hierarchy, data in hierarchies above are aggregated: Having the default sequence of hierarchies as described above, the *Hierarchical List* will display one node for each year in the highest level of the hierarchy, giving aggregated values for all records that are stored for this year. Next, month nodes are created and associated to the respective year node, encapsulating all values that have the same month in the corresponding year. This procedure is repeated for all hierarchy attributes (payment group and payment method) in the database until the the last one – in the default sequence

## 4 Parallel Coordinated Hierarchies

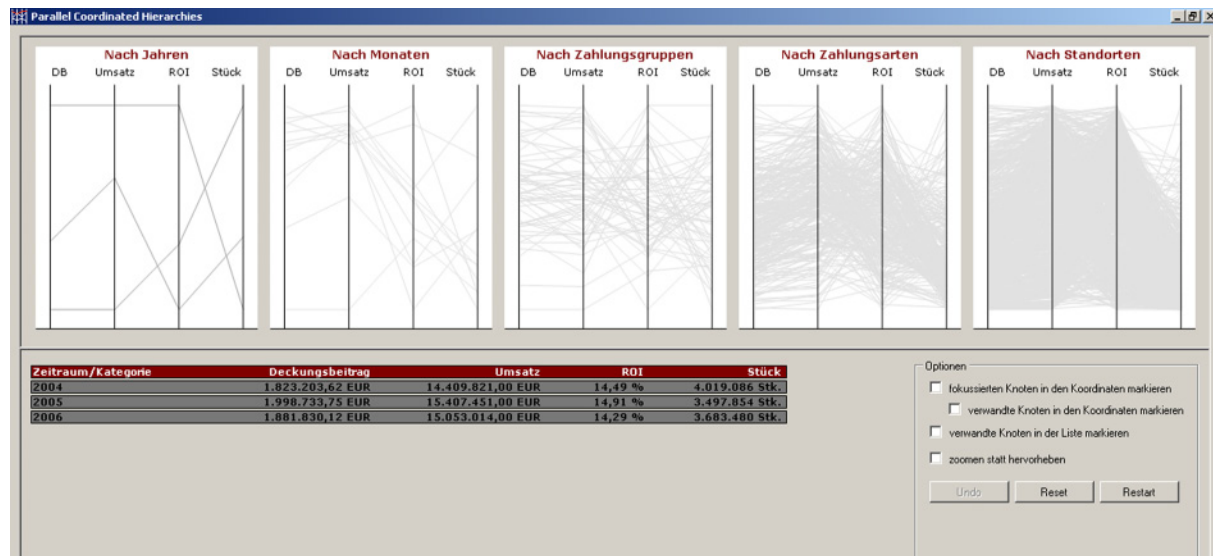
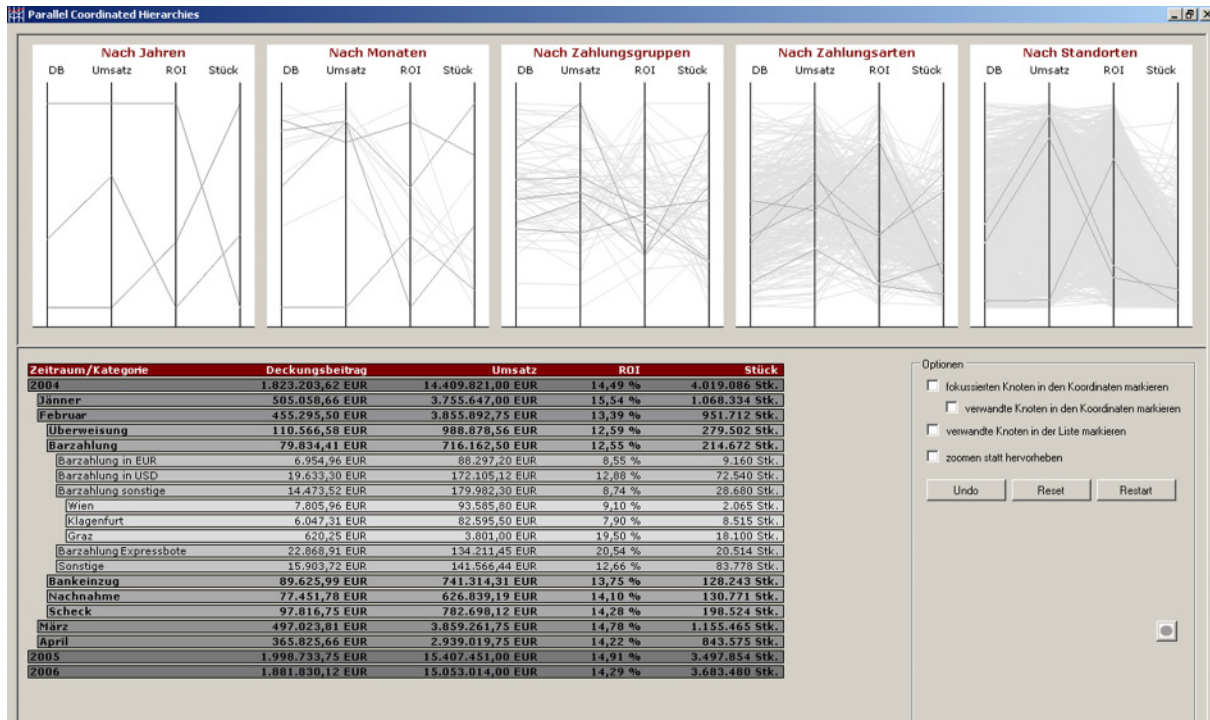


Figure 4.1: *Parallel Coordinated Hierarchies* – default view.

the location – is processed. Instead of aggregated values, the lowest hierarchy always represents values as originally recorded in the database, regardless of the sequence of hierarchies.

According to the levels of hierarchy in the *Hierarchical List* five *Parallel Coordinates* plots (PCPs) are created (see Figures 4.1 and 4.2), each consisting of four axes (one for each dimension). Every node (every multidimensional point) that has been added to the list at a certain level of hierarchy is also displayed as a polyline in the respective PCP. Having in mind the default sequence of hierarchies as described above, the PCP for year for example will therefore display one polyline for each year, where the values of the multidimensional points – just like in the list – are defined as the sum of all records' values stored for the respective year (except for return-on-investment that is calculated as a ratio). Each polyline in the second display – the one for month in the default view – represents the values aggregated by months and belonging to a certain year. Thereby we do not distinguish between months that belong to different years but display all values of this dimension within the same PCP. To clarify which year a certain month node belongs to and which months belong to a certain year, respectively, several means of interaction will be introduced later in this chapter. Again, these systematics can be applied to the remaining three PCPs: The one for payment groups displays all those nodes as polylines, that are in the third level of hierarchy in the list. All nodes contained in the list's fourth

Figure 4.2: *Parallel Coordinated Hierarchies* with expanded hierarchies.

level of hierarchy are displayed as polylines in the fourth PCP – the payment methods – in the default view. Finally, all nodes at the last level of hierarchy are used to create the polylines displayed in the fifth PCP, again regardless of the particular parent node it is associated with.

### 4.3.2 Exemplary Displays

The linked view consists of a *Hierarchical List* and one *Parallel Coordinates* plot (PCP) for each hierarchy. As can be seen from Figure 4.1, test data containing three years are depicted in the *Hierarchical List* at the bottom of the display and in the first PCP at the top left respectively. As all three year nodes are contained in the highest level of hierarchy and therefore are visible, they are also highlighted in the respective PCP as they are displayed in a more saturated black hue. Polylines in the four displays to the right are dimmed out, as the respective nodes in the *Hierarchical List* are not visible for the moment. Each of the five PCPs is linked to exactly one hierarchy and displays a cross section through all nodes in the respective level.

Figure 4.2 on the other hand illustrates the linking concept between the PCPs and the *Hierarchical List*: Here the nodes “2004”, “February” and the payment group “Barzahlung” (paying cash) have been expanded. Within this combination of year/month/group the payment method “Barzahlung sonstige” (paying cash by other means) is expanded, now displaying nodes for three locations in which this method (within this specific group within this specific month within this specific year) was applied. Just like in the list, the PCPs display the expanded nodes in the respective dimensions – highlighted with a more saturated hue. The linking between the list and the *Parallel Coordinate* plots in this context can be summarized as follows: Whatever node is visible in the list is also highlighted in the respective PCP – and the other way round, as will be shown in the following section.

### 4.4 Interaction and Features

Linking goes beyond what has been presented before. Mostly any kind of interaction in both the *Hierarchical List* and the PCPs is synchronized. In general, this means that whatever manipulation is made in one representation has the same – or equivalent – effects in the other one. This section will not only present interaction features realized by the prototype but will especially take linking into consideration when discussing:

- General features
- Methods of highlighting
- Features concerned with brushing and zooming
- Changing hierarchies and axes

Each feature discussed below moreover is assigned firstly an identifier that will be referred to in the evaluation in Section 5.2 and secondly a representation – either PCP, the *Hierarchical List* or the linked view as a whole that acts as the initiator of the described feature. For reasons of confidentiality, we will only show test data while discussing interaction. Real data will only be used for evaluation (Section 5.2).

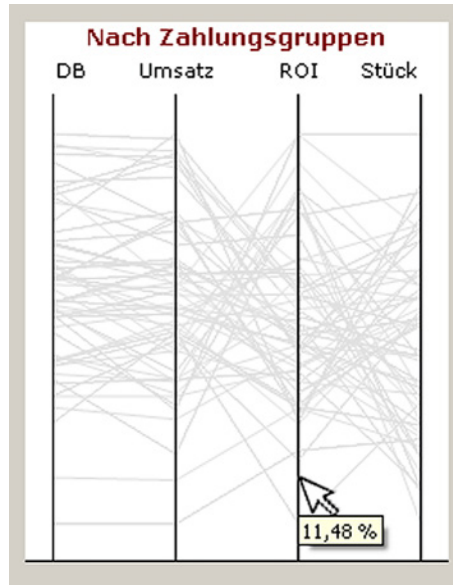


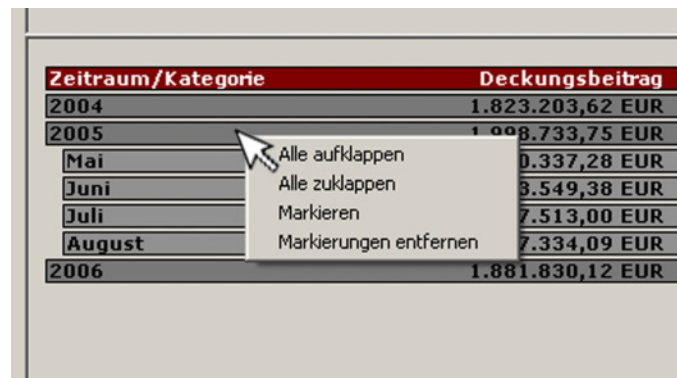
Figure 4.3: (G1) Information on axis' scale via text tooltip.

#### 4.4.1 Section G: General Features

The first features discussed are general ways of interaction and definitions that are not necessarily restricted to the *Parallel Coordinated Hierarchies* approach but could be found in either, *Hierarchical Lists* or *Parallel Coordinates* plots.

**G1: Parallel Coordinates – Scale Information.** As can be seen in Figure 4.1 and Figure 4.2, PCPs do not have axes labels. Due to the fact, that it would overload the displays, the axes' scales are only available by a text tooltip, superimposed whenever the user's mouse pointer is over a point of an axis (for an example see Figure 4.3).

**G2: Parallel Coordinates – Scaling.** Scaling is done with respect to the highest and lowest value displayed, where each axis' scale is calculated depending on whether only one or more polylines are contained in the PCP. Given the axis' maximum and minimum value, the maximum difference between values on the respective axis is calculated. The upper bound of the axis then is the maximum value plus an additional 10% of the maximum difference calculated in advance. The same is done for the minimum: The lower bound of the axis is the minimum value minus an additional 10% of the maximum difference.



Zeitraum/Kategorie	Deckungsbeitrag
2004	1.823.203,62 EUR
2005	1.908.733,75 EUR
Mai	0.337,28 EUR
Juni	3.549,38 EUR
Juli	7.513,00 EUR
August	7.334,09 EUR
2006	1.881.830,12 EUR

Figure 4.4: (G4) Right click menu for *Hierarchical List*.

In case only one polyline has to be displayed – i.e., minimum and maximum values are equal – the upper and lower bound are defined to be 5% higher, respectively lower, than the value.

**G3: Hierarchical List – Expanding and Collapsing Nodes.** The list’s nodes may – according to standard list features – be expanded by a double click. At the same time all affected hierarchies’ PCPs are updated concerning highlighting of expanded nodes. Double clicking an already expanded node collapses it – however only if no descendant node has been brushed. If a node is expanded again that has already been expanded (at least once) before, also his children’s expansion state is restored. According to the link between the *Hierarchical List* and the PCPs, all expand and collapse operations are synchronized with the PCPs’ displays.

**G4: Hierarchical List – Expanding and Collapsing All Nodes.** In order to facilitate browsing the *Hierarchical List*, a pop-up menu (like shown in Figure 4.4) superimposes the respective node on a right click, containing the option “Alle aufklappen” (expand all) and – in case any lower level of hierarchy for the focussed node is displayed – the option “Alle zuklappen” (collapse all). If the expand-all option is chosen, all nodes on lower levels having the focussed node as ancestor are expanded. Choosing the collapse-all option collapses the respective node, even in case any child nodes are brushed. The respective PCPs are updated concerning expanded nodes and nodes formerly brushed. Collapsing all nodes moreover discards all nodes’ expanding states.

**G5: Parallel Coordinated Hierarchies – Undo.** *Parallel Coordinated Hierarchies* support the standard feature undo, where going back to former system states is not restricted to a certain number of actions. Any permanent change in the system’s state – like for example starting the application, expanding nodes, brushing polylines or zooming axes – is logged and may be restored.


**G6: Parallel Coordinated Hierarchies – Reset.** The “Reset” button restarts the system with exactly the sequence of hierarchies and axes, that is displayed currently. Any operation like expanding nodes, brushing polylines or zooming axes carried out before is discarded – but may be restored by using the “Undo” feature.

**G7: Parallel Coordinated Hierarchies – Restart.** The “Restart” button on the other hand not only resets the systems state to an intermediary starting point (G5) but restarts the prototype all over. At the moment, there is no “Undo” support (G4) for this operation, as the history of all former system states is discarded as well.

**G8: Parallel Coordinated Hierarchies – Record.** The prototype supports recording events initiated by the user (square button at the lower right in Figure 4.2). This feature will be used during the user evaluation to also allow quantitative analyses of the user’s interaction with the application.

### 4.4.2 Section H: Highlighting

The first application specific set of features discussed is concerned with highlighting nodes and/or polylines. The distinction between highlighting and brushing (see the following Section 4.4.3) is based on the actions’ permanence. While highlighting includes all changes in the nodes’ and polylines’ color that are non permanent – for example caused by the mouse pointer being over a certain node – brushing includes all user-initiated, permanent selections of certain polylines or nodes respectively.



Zeitraum/Kategorie	Deckungsbe
2004	1.823.203,62
2005	1.998.733,75
2006	1.881.830,12

Figure 4.5: (H1) Highlight node on “mouse pointer over” in the *Hierarchical List*.

**H1: Hierarchical List – Highlight Focused Node.** A feature that cannot be turned off at the moment is highlighting a focussed node when the mouse pointer is over it (see Figure 4.5). In order to give the user a better sense of navigation, nodes are colored light-red if the region defined by their borders is entered. In contrast to most of the following functions, this feature used alone does not show the link to the PCPs because highlighting is restricted to the node.

**H2: Hierarchical List – Highlight Related Nodes.** All nodes related to the focussed node may be highlighted, where the node’s ancestors are colored dark-red, and the node’s children are colored in a very light-red. Just like with highlighting the focussed node (H1), this feature used alone is restricted to the *Hierarchical List* and has no consequences on any of the PCPs. In contrast to highlighting before, this option has to be activated by selecting the respective checkbox on the lower right, saying “verwandte Knoten in der Liste markieren” (“mark related nodes in the list”). For an example on the outcome of this feature see Figure 4.6, where related nodes in higher levels of the hierarchies are colored dark-red, whereas the lower level hierarchies are colored lighter. In case lower level hierarchies are not visible, only the focussed node and his ancestors are colored.

**H3: Parallel Coordinated Hierarchies – Highlight Open Nodes.** The link between the list and the PCPs becomes visible at first in highlighting polylines equivalent to expanded nodes. As mentioned before and shown for example in Figure 4.2, the PCPs’ polylines corresponding to expanded nodes in the list are colored in a more saturated hue.



Zeitraum/Kategorie	Deckungsbe
2004	1.823.203,62
2005	1.998.733,75
Mai	490.337,28
Juni	483.549,38
Überweisung	80.493,80
Barzahlung	125.057,73
Bankeinzug	117.787,46
Nachnahme	93.315,66
Scheck	66.894,73
Juli	507.513,00
August	517.334,09
2006	1.881.830,12

Figure 4.6: (H2) Highlight related node on “mouse pointer over” in *Hierarchical List*.

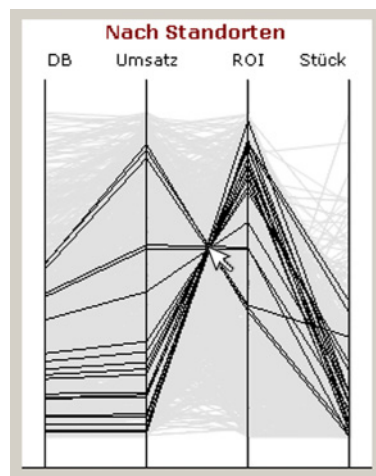


Figure 4.7: (H4) Highlight focussed polylines in *Parallel Coordinates*.

**H4: Parallel Coordinates – Highlight Focussed Polylines.** Another basic feature is highlighting polylines in all PCPs that are below the mouse pointer’s position as shown in Figure 4.7 for the level of locations. Especially in a high density display this facilitates tracing polylines crossing a specific point of the display. Further operations based on this selection are for example brushing like presented in Section 4.4.3 (B1) – where interacting with coordinates is linked to the list. The feature presented here only affects the focussed PCP.

## 4 Parallel Coordinated Hierarchies

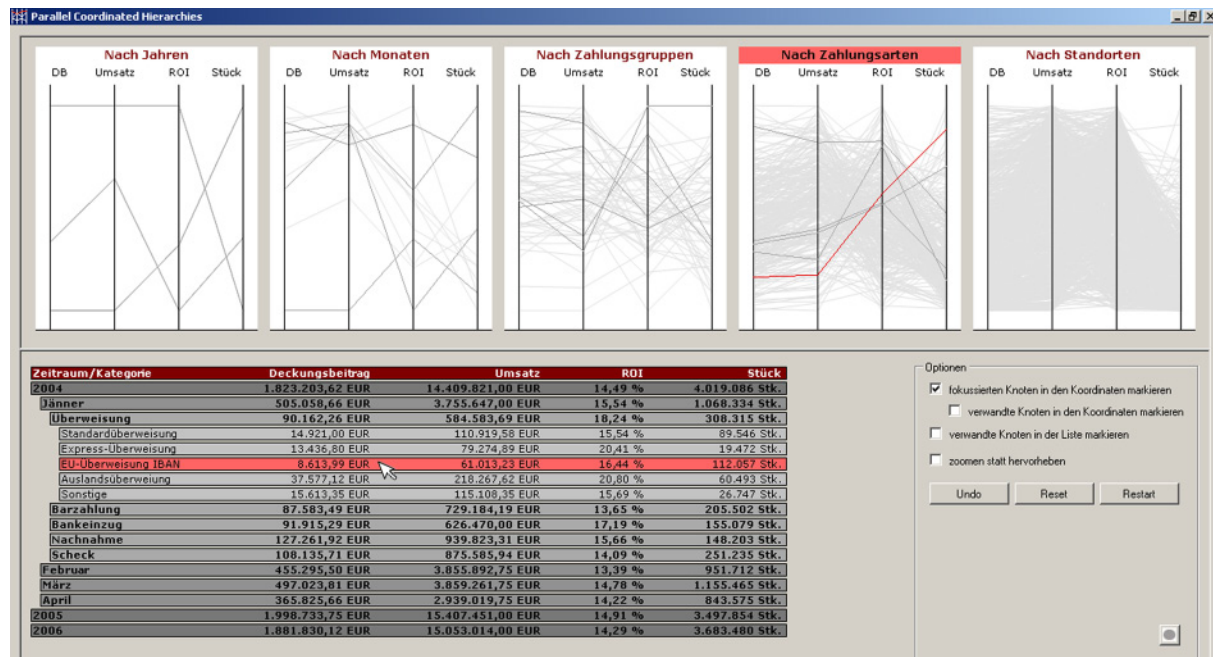


Figure 4.8: (H5) Highlight focussed node and the corresponding polyline.

### H5: Parallel Coordinated Hierarchies – Highlight Focussed Nodes and Polylines.

As in (H3), highlighting focussed nodes and corresponding polylines emphasizes the link between the two representations. As mentioned for highlighting related nodes (H2), this feature can be activated by selecting the respective checkbox. Apart from coloring the polyline red, the headline of the affected PCP is colored red, too. As can be seen from Figure 4.8, the focussed node and the corresponding polyline in the fourth PCP change their background color to light-red. The checkbox saying “fokussierten Knoten in den Koordinaten markieren” (“mark focussed node in coordinates”) on the right has been selected to activate this feature.

### H6: Parallel Coordinated Hierarchies – Highlight Related (Nodes and) Polylines.

The last feature concerned with non-permanently marking is highlighting related points in the PCPs. As presented for highlighting related nodes (H2), highlighting related polylines can be activated by selecting the respective checkbox. Whenever a node is entered by the mouse pointer the corresponding polyline is marked in red (H5) as well as the polylines corresponding to its ancestor and child nodes. Coloring follows the same scheme as in (H2): Children are colored in a lighter red, parents in a darker red. As can be seen from

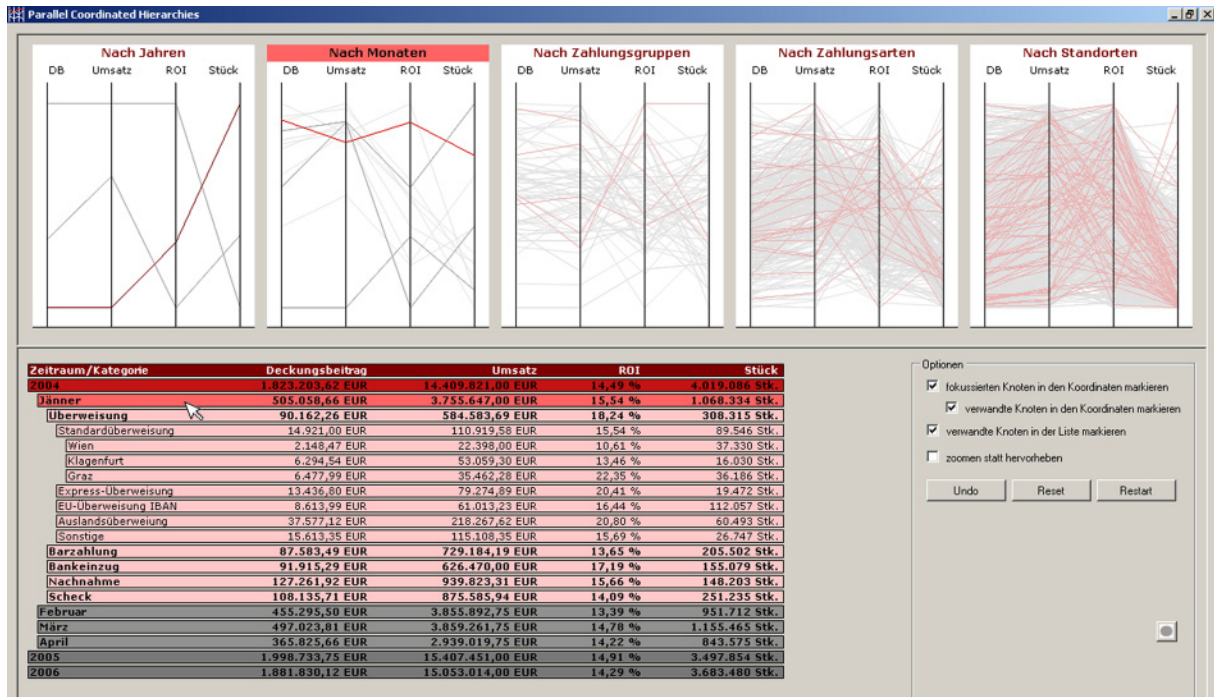


Figure 4.9: (H6) Highlight focussed and related node and polylines.

Figure 4.9, selecting this option (“verwandte Knoten in den Koordinaten markieren” – “mark related nodes in coordinates”) may only be done in combination with (H5), highlighting the focussed node in coordinates. Moreover, Figure 4.9 shows the combined use of highlighting related nodes (H2) as well as coordinates (H6), coloring the affected nodes and corresponding polylines as described above. As stated for feature (H2), child nodes will only be colored, if they are visible. Any successive node whose parent is not expanded at the moment of highlighting will not be expanded for this non-permanent operation.

### 4.4.3 Section B: Brushing and Zooming

Highlighting includes all non-permanent changes in the nodes’ and polylines’ colors, for example due to the mouse-pointer position. In contrast brushing includes all interactions, that are initiated by the user and have permanent changes as a consequence. The following section will give an overview on the realized brushing techniques as well as on zooming, a special case within brushing operations.

**B1: Parallel Coordinates – Brush Focussed Polyline.** As seen in (H4), if the mouse pointer resides on a polyline, this polyline will be highlighted in black. In order to permanently brush this line or a set of lines, a right click can be performed that permanently colors the affected polylines in dark-green. At the same time – considering linking between the two representations – the list is modified as follows: If the node corresponding to the brushed polyline is visible, its background color will change to dark-green. In case it is not, the respective node (including all affected ancestor nodes) will be displayed before the corresponding node is colored in dark-green. An example is given in Figure 4.10, where the mouse pointer resides on two polylines that are brushed by a right click. Immediately the list is adapted, (if necessary) expanding the affected ancestors and coloring the corresponding nodes. Although this operation is permanent considering the selection of polylines and nodes so, that operations like changing axes (C2) do not influence the selection, nodes and polylines will be deselected by any subsequent brushing operation (B1, B2) except (B3). For possible future extensions on brushing – allowing for example additional selection – see Chapter 6, Conclusions.

**B2: Parallel Coordinates – Brush Polyline and Nodes by Axes Segment(s).** Complementary to brushing in feature (B1), users may define axes segments where polylines – and their corresponding nodes – intersecting the axes in the defined segment may be brushed. First, the user specifies a segment on any of the axes within a PCP using the mouse as shown in the left display of Figure 4.11. This region does not necessarily have to be restricted to only one axis, but may also cover several axes whose respective polylines – intersecting the axes within the defined rectangle – will be selected for brushing. The righthand display of Figure 4.11 shows the resulting PCP with polylines colored in dark-green. Corresponding to the brushed polylines, also the list is adapted as presented before (B1).

**B3: Hierarchical List – Brush/Unbrush Focussed Node.** Another way to brush single polylines is using the popup menu displayed on a right click on any of the nodes (see also Figure 4.4. Choosing the option “Markieren” (highlighting) brushes the focussed node. In contrast to brushing presented in (B1) and (B2), brushing a single node here adds the

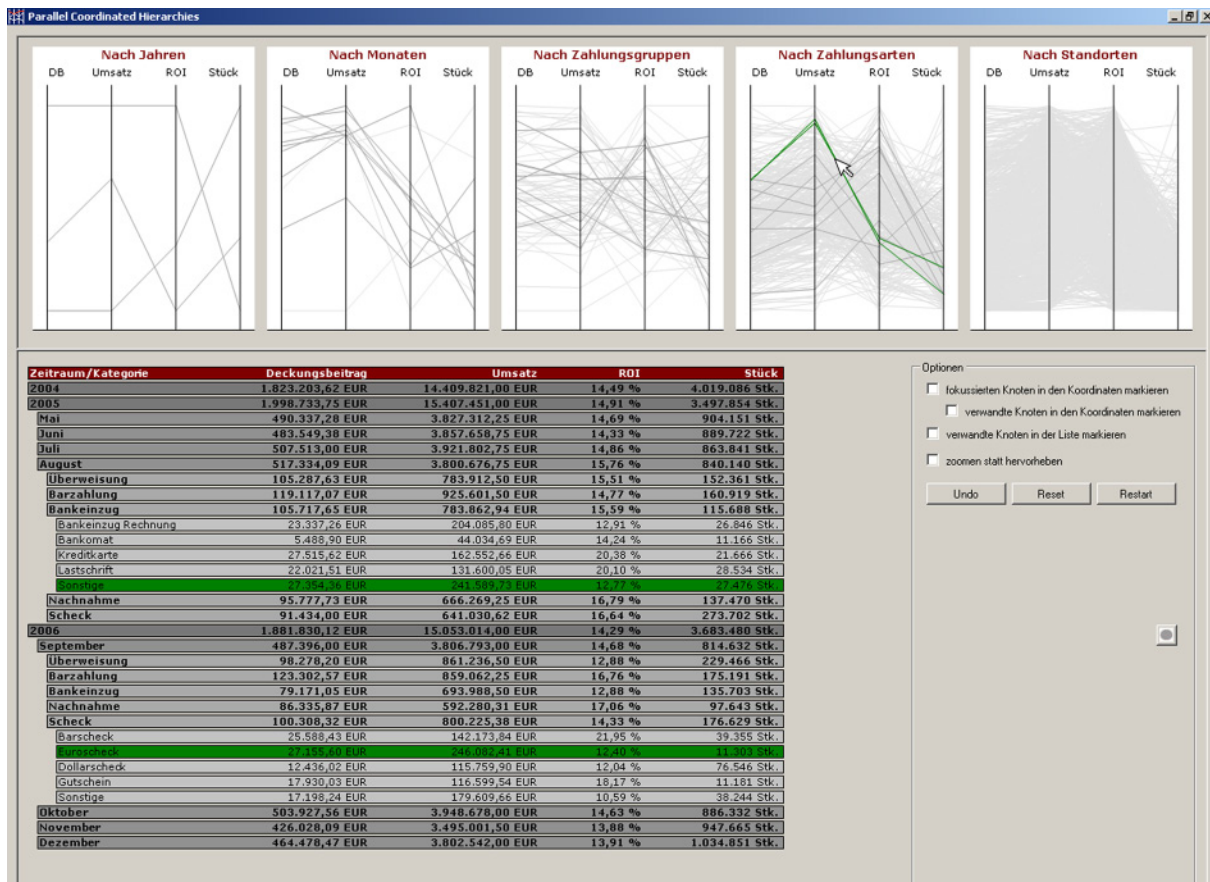


Figure 4.10: (B1) Brushing polylines and nodes according to the mouse-pointer position.

selected polyline to the set of already brushed nodes and polylines, no matter which PCP or hierarchy is affected. Deselecting nodes and polylines either works by simply double clicking nodes or by using the menu's option "Markierungen entfernen" (deselect) (also see Figure 4.4), where – in contrast to brushing a node via menu – the focussed node and all its ancestors are excluded from brushing. As mentioned above, unbrushing may also be done by using the collapse-all feature (G4), where nodes are collapsed and their brushing state is discarded. Another feature that (not only) may be used to unbrush polylines and nodes is the "Reset" button (G6), that resets the system considering brushing and zooming as well as the nodes' expansion states.

**B4: Parallel Coordinates – Zoom Axes Region(s).** A special case of brushing is zooming axes segments. As for brushing in (B2), zooming works with selecting certain axes segments. In contrast to brushing, this feature is designed to be cumulative, meaning,

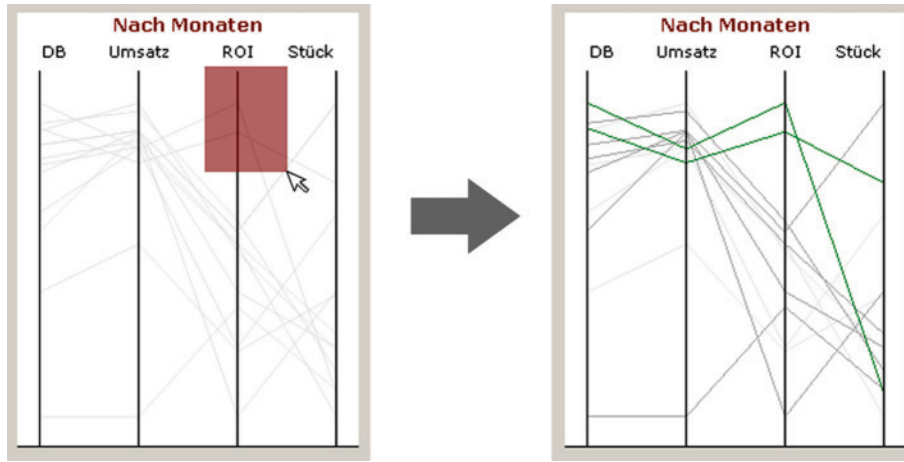


Figure 4.11: (B2) Brushing axes segments by mouse-pointer selection.

that once a zoom was performed, zooming again does not undo the first operation but applies the operation to the remaining values. Zooming itself works as follows: First, the respective checkbox has to be selected to change from conventional brushing to zooming. Secondly, a region covering one or more axis in one PCP is defined using the mouse. After confirming the selection, all PCPs are updated, where in the focussed PCP only those polylines now still are contained, that intersected either of the axes in the area defined before. All other polylines are excluded from display – in the PCP as well as in the *Hierarchical List*. Next, axes are rescaled with respect to the new maximum and minimum values. Afterwards, all other PCPs – and the list respectively – are updated considering the polylines in the focussed PCP. Only those values, that either are ancestors or descendants of polylines still contained in the focussed PCP are displayed, the rest is excluded. Axes are rescaled with respect to the values still contained in the display. For an example see Figure 4.12, where in the upper figure an axis segment is defined by first drawing a rectangle over a chosen axis. The lower figure shows the result with PCPs only containing polylines that either intersected the chosen axis in the defined segment of the focussed PCP or – for the other PCPs – are ancestor or descendant points of any of the polylines still contained in the focussed PCP. As a result, not only the PCP display changes, but so does the list, that now – considering linkage between the two representations – is restricted to either nodes corresponding to the zoomed polylines or their relatives still contained in the other PCPs.



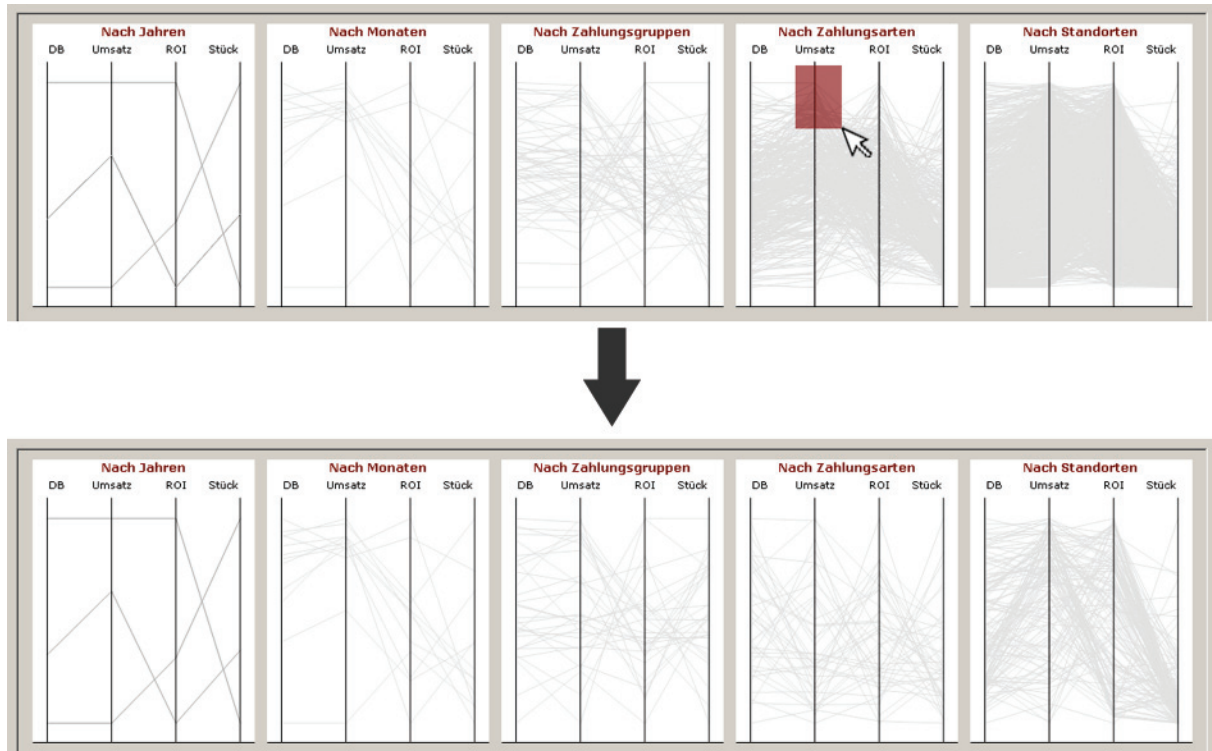


Figure 4.12: (B4) Zooming axes segments.

#### 4.4.4 Section C: Changing Hierarchies and Axes

The last set of features is concerned with changing hierarchies and axes. In the evaluation described in Section 5.2, these two features turned out to add great value to the *Parallel Coordinated Hierarchies*. Just like the features presented so far, both operations are reversible and therefore may be reversed using the prototype's Undo feature (G5).

**C1: Parallel Coordinates – Change Hierarchies.** Changing hierarchies was realized as a drag-and-drop operation. The user may pick one hierarchy as a PCP and either drag it on or besides another PCP. If the dragging ends on another PCP, the respective hierarchies – and with them the PCPs – are exchanged. In case the user drops the picked PCP besides another one, the dragged hierarchy is inserted at the level it is dropped.

**C1a: Parallel Coordinates – Change Hierarchies by Drop On.** Figure 4.13 gives an example on changing hierarchies by dropping a PCP (the one for year) on another one (in

#### 4 Parallel Coordinated Hierarchies

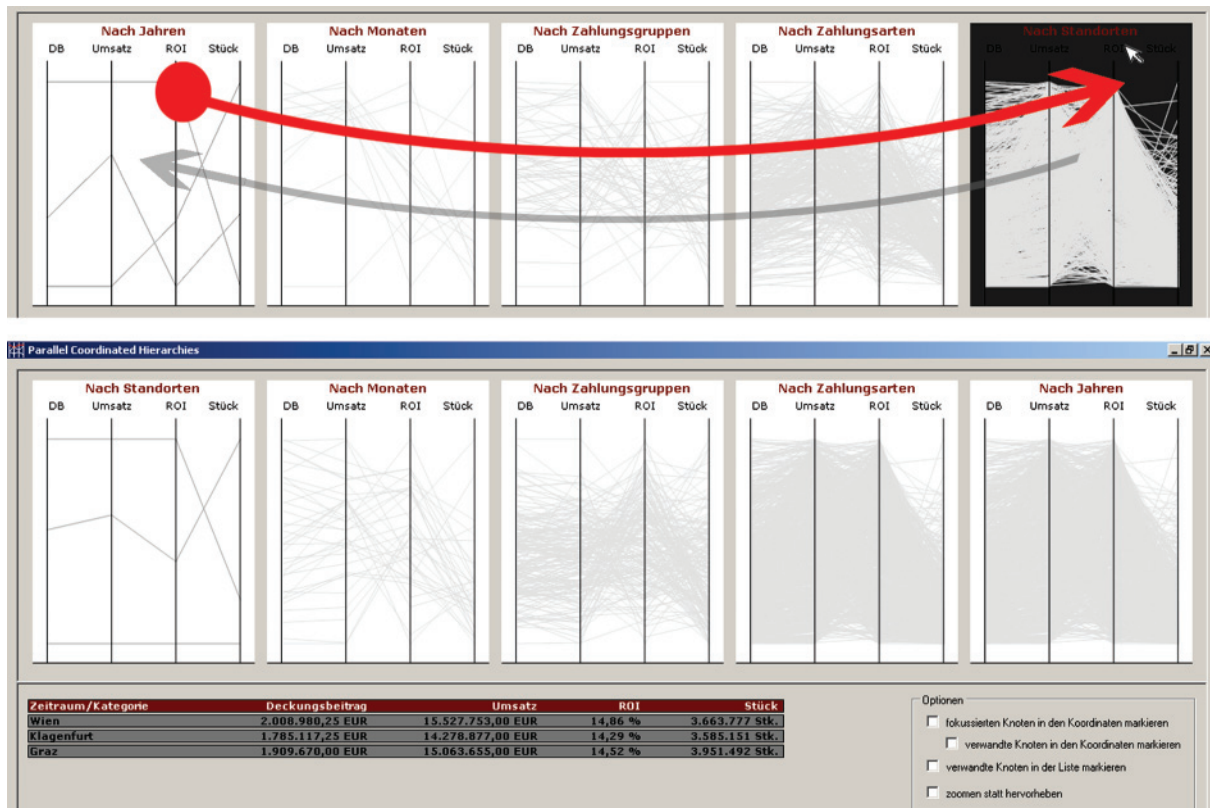


Figure 4.13: (C1a) Exchanging hierarchies by dropping one PCP on another one.

this case location) – here depicted by the red arrow to better illustrate the process. First, the user picks up the PCP for year by clicking on the label “Nach Jahren” and holding the left mouse button pressed while secondly moving to the PCP for location. Being there, the prototype signals its readiness to change hierarchies by coloring the target PCP’s background black. If the mouse button is released, the exchange of hierarchies, whose result is shown in the lower figure, takes place. Now location is the first hierarchy to be displayed, year on the other hand is the last one, as can be seen from the order of PCPs. Consequently the exchange not only takes PCPs into consideration but also the *Hierarchical List*, as shown in the lower display of Figure 4.13, where the three locations are displayed as first level nodes.

**C1b: Parallel Coordinates – Change Hierarchies by Drop Besides.** The second approach to change hierarchies is shown in Figure 4.14, where hierarchies are changed by dropping one PCP besides another one. Like mentioned for (C1a), any PCP is first picked



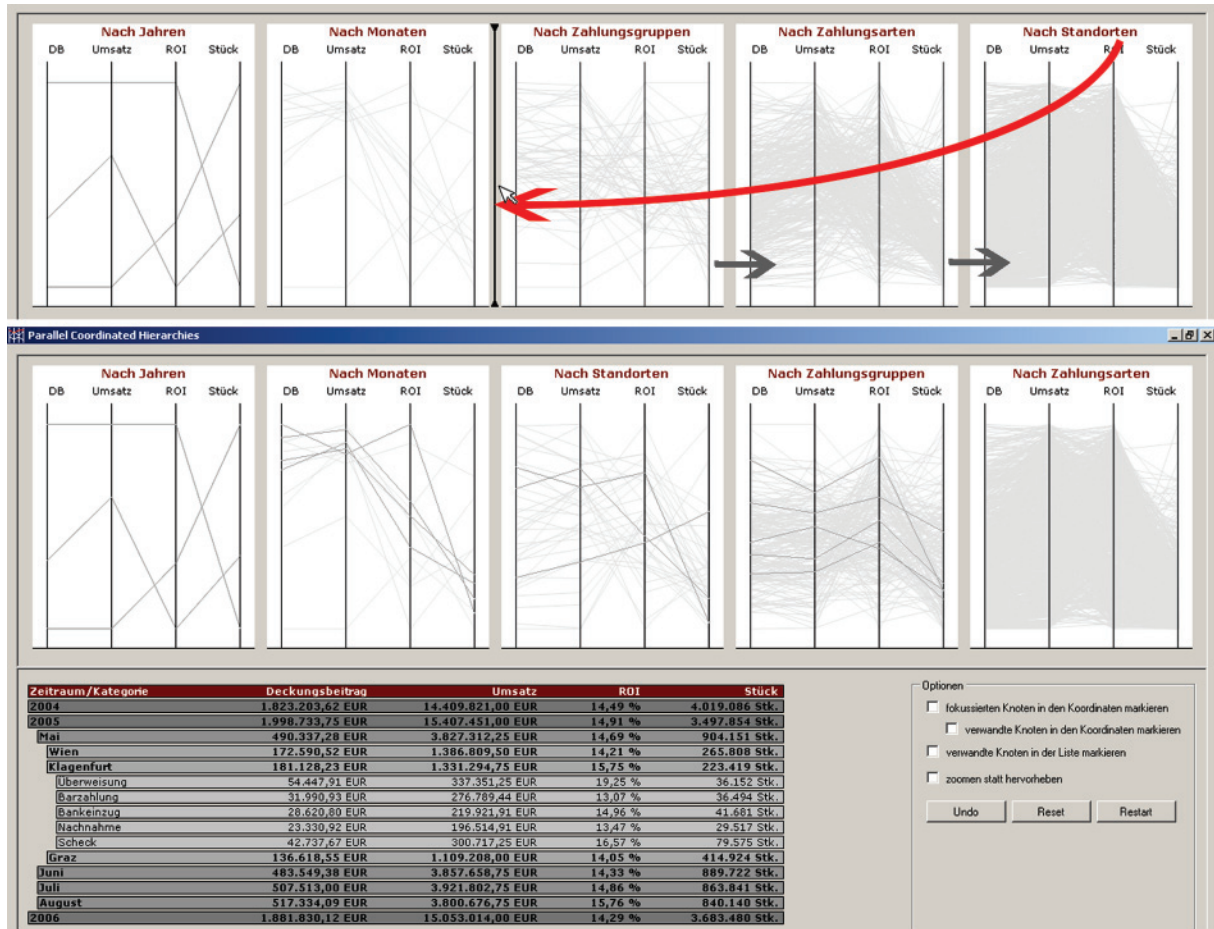


Figure 4.14: (C1b) Changing hierarchies by dropping one PCP besides another one.

and moved, where the target now is not another PCP but the area between two PCPs (or the left and right side of the display, respectively). Again, the prototype signals its readiness to receive the dragged PCP, in this case by showing a black line – in the example depicted in the upper display of Figure 4.14 between the second and the third PCP. Once dropped, the hierarchy dragged is inserted at the respective position. PCPs at lower hierarchies are moved to the right – indicated by the gray arrows – which is equivalent to rearrange corresponding hierarchies at lower levels. As a result, the lower display in Figure 4.14 shows the new order of PCPs and the rearrangement in the list: Data is now aggregated by year and month as before, followed by the location now in third place. Finally, data is grouped by payment group and payment method in the last two levels of the hierarchy.

## 4 Parallel Coordinated Hierarchies

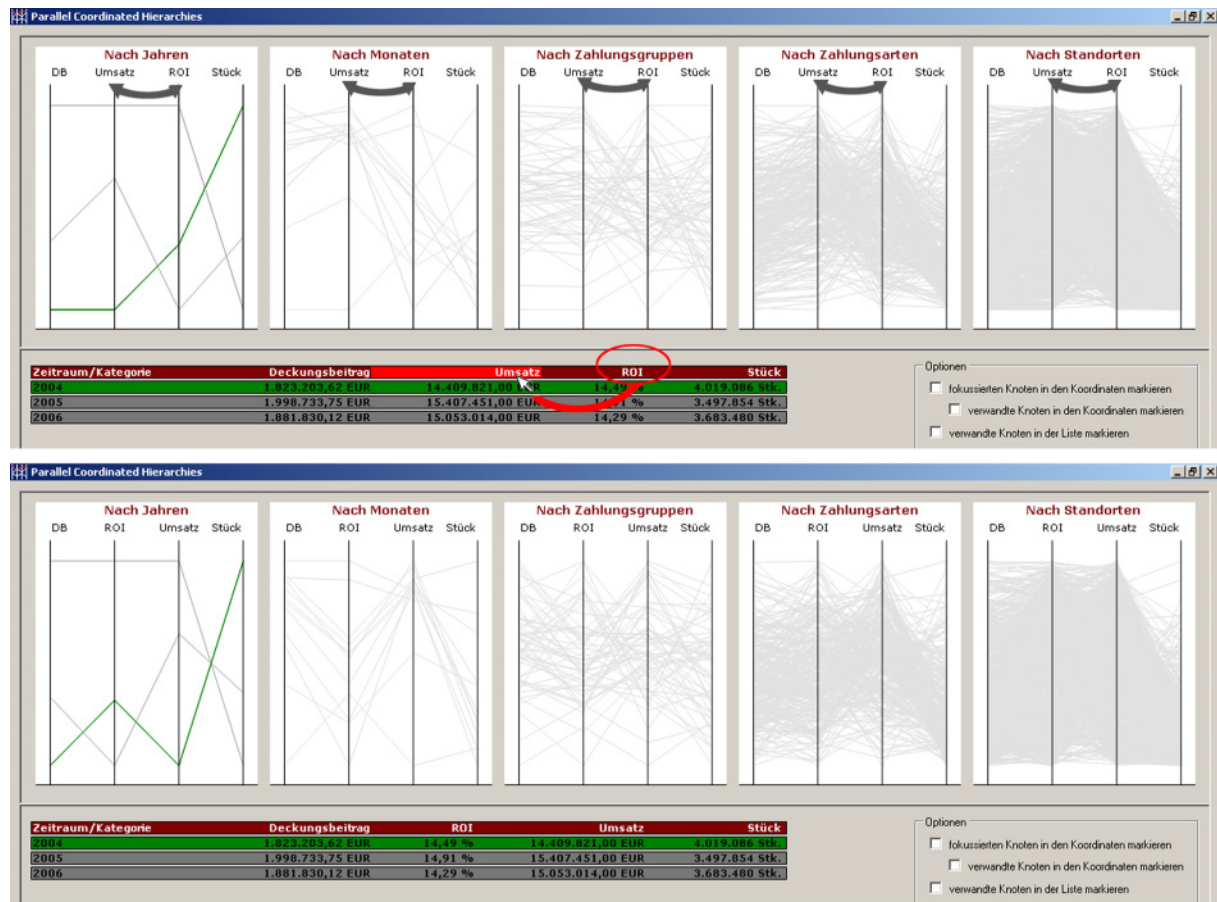


Figure 4.15: (C2) Changing axes by dropping one label on another one.

**C2: Hierarchical List – Change Axes.** The last feature implemented is changing axes. Although it is also realized as a drag-and-drop operation, changing axes is – in contrast to changing hierarchies (C1) – only implemented as a drop-on feature. To change axes, the user simply drags one of the dimension labels from the list on another one, causing axes to exchange. For an example see Figure 4.15: Here the user drags the label “ROI” on the one for turnover (“Umsatz”) in order to find out about correlations between contribution (“Deckungsbeitrag”) and return-on-investment (“ROI”) for example. The label for turnover is highlighted in red in the upper display of Figure 4.15 to signal the user that he may drop the dragged label here. The red arrow symbolizes the user’s drag operation, the gray ones in the PCPs depict the change of axes in all other representations. In the lower display in Figure 4.15, the change has already been carried out, now showing the “ROI” dimension between “Deckungsbeitrag” and “Umsatz” – as well in the list as in all PCPs. Also illustrated in Figure 4.15: The point “2004” has been brushed before starting

the change of axes (upper display) and – after the exchange – still is brushed in the lower display. A change of axes therefore does not influence brushed values, as it just rearranges dimensions without newly aggregating them.

## 4.5 Implementation

Before describing the evaluation of the prototype and its outcomes in the following chapter, the next paragraphs will give an overview on the implementation of the prototype. As the source code is not part of this thesis we will just describe the development environment and the main characteristics of the *Hierarchical List* and the *Parallel Coordinates* controls.

**Software Environment.** The prototype itself was developed using Microsoft® Visual Studio .NET 2005. Microsoft®'s SQL Server 2000™ is used as database for the test dataset, whereas the real data were processed on the free version of Microsoft®'s SQL Server 2005™, the SQL Server Express. The code was written in C#, one of the object oriented programming languages supported by the .NET framework.

**Hierarchical List.** Although the .NET framework provides a *Hierarchical List* with standard components, its interaction possibilities were insufficient for what was needed for the prototype. Therefore a proprietary list was developed, that basically is organized in nodes which again have a list attribute for the administration of their children. Each of the lists – whether the main list itself or any node's list – can have an arbitrary number of nodes that determine the list's behavior. Nodes provide certain flags – for example for being visible, expanded, brushed, excluded, focussed and some more – that allow an efficient administration of the list. Painting the *Hierarchical List* on the screen – and therefore painting nodes – is also handled by the nodes, that each first paint themselves. Afterwards – if they are marked as expanded – they go through their list of children, in turn initiating their painting by passing them the current vertical position. After a child has finished painting, it returns the updated vertical position and the primary node can ini-

tiate painting the next child. As every node is organized as a control – that for example automatically recognizes mouse events and therefore causes some overhead in administration – the list unfortunately has limits concerning drawing performance. If for example a great number of polylines is brushed, updating the list may take some time.

**Parallel Coordinates Plots.** Just like the list, the *Parallel Coordinates* plot (PCP) is a proprietary development as no freely available implementation of *Parallel Coordinates* could be found for the .NET framework. In contrast to the nodes, polylines here are not realized as controls for performance reasons, but were implemented using Microsoft®’s graph drawing library GDI+. Here, all (mouse events – like mouse over a certain position for example – had to be handled “manually” by determining where the mouse was in the display, what polylines for example intersect with the mouse pointer position or where on an axis the mouse pointer resided at the moment. Interacting with PCPs therefore is more efficient than with the *Hierarchical List* so, that more elements can be administrated efficiently – due to the fact, that only those events had to be handled (centrally), that really were needed for the prototype’s features.

## 5 Evaluation

After having introduced both *Clickstream Visualization* and *Parallel Coordinated Hierarchies*, we will now turn to the concepts' evaluation. DiTech's managing director and the main responsible person for programming and maintaining DiTech's online shop were asked to comment on the most interesting domains of analysis and visualization. First, we will give an interpretation of possible usability problems by presenting and discussing two displays generated with the *Clickstream Visualization* prototype and used in a short user evaluation carried out with the main responsible person for programming and maintaining the online shop. The second part of this chapter is dedicated to the systematics and outcomes of the user evaluation carried out with the *Parallel Coordinated Hierarchies* prototype and DiTech's managing director.

### 5.1 Interpretation and Evaluation of Clickstream Visualization

The first proof of concept discussed in more detail is the *Clickstream Visualization*. As we have described in Section 3, the displays generated by the prototype are designed to give further insights in users' flows on DiTech's websites. So far, only the graph itself and the meaning of its nodes and edges have been discussed, leaving aside conclusions and interpretations of these flows. This section will therefore first look at some possible usability problems that could be revealed in the user-feedback session before presenting necessary extensions to make the prototype usable for everyday business.

### 5.1.1 Interpretation

Displaying flows through a website with a node-link diagram like presented in Figures 5.1 and 5.2 can be used to uncover weaknesses in the website's structure and features. What these weaknesses are and how to point them out having the prototype's displays will be discussed in the following paragraphs. Moreover, we will interpret the website usage apart from usability problems.

**Most Traveled Path.** As can easily be seen from Figures 5.1 and 5.2, both in March and July 2005 the most traveled path was the click within categories – denoted by the self reference at the *openkat* node in the middle of the lower row. Having about 3.000 products categorized by type, compatibility or manufacturer for example, the hierarchical structure of categories has to be clicked several times before getting displayed products that meet the user's requirements. As the majority of users browses the online shop in order to find certain products, and as main and all sub categories are logged with identical *eventcode* attributes, it is not surprising, that browsing categories makes up most page traffic. For example in March 2005 22.1% of all transitions were clicks on a category after a click on a category has been made directly before.

**Different Website Usage.** The second aspect that seems noticeable is the systematically different website usage in March and July. While users split up in four groups of pages after entering the website in March (*search*, *html*, *openkat*, *titel*) via the *URL* node, all users in July seem to request a static *html* page at first. On the other hand, coming to the online shop via a partner link (*Partner* node) in March, users concentrate on viewing product details, while in July – additionally to that – they request a static *html* page.

The first fact – splitting up in March while requesting only one specific group of pages in July – can be due to certain different circumstances: As was found out for all months following July but December, user flows from *URL* were concentrated on *html*, where one might assume that logging was changed in this period: while a user's request for the main entry page might not have been logged until March and after the end of November, it may have been done in between. If, on the other hand, the content in the main frame of

## 5.1 Interpretation and Evaluation of Clickstream Visualization

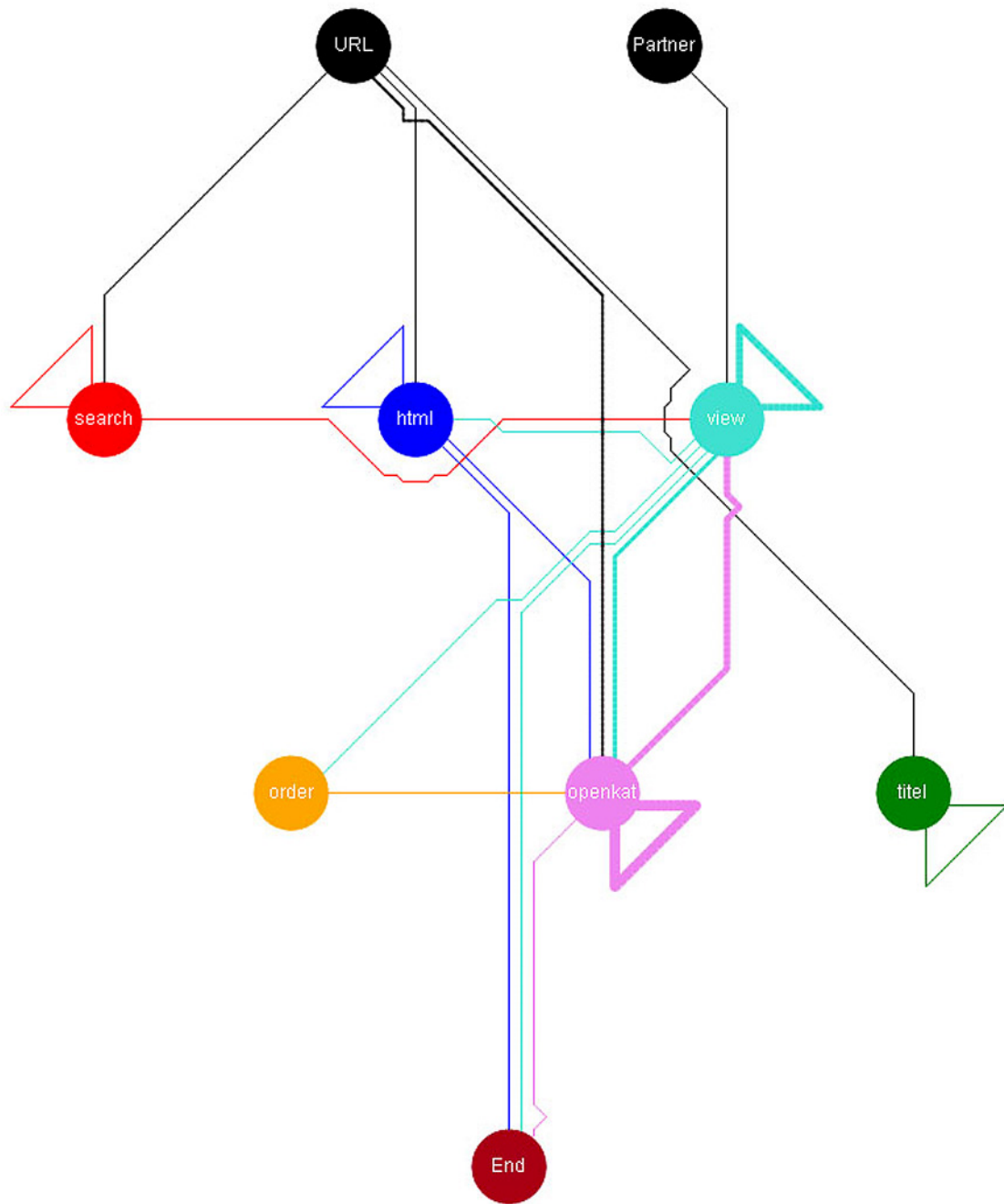


Figure 5.1: User flows in March 2005.

the site was static HTML from July to November, its access may have been logged as if the user actively clicked a link.

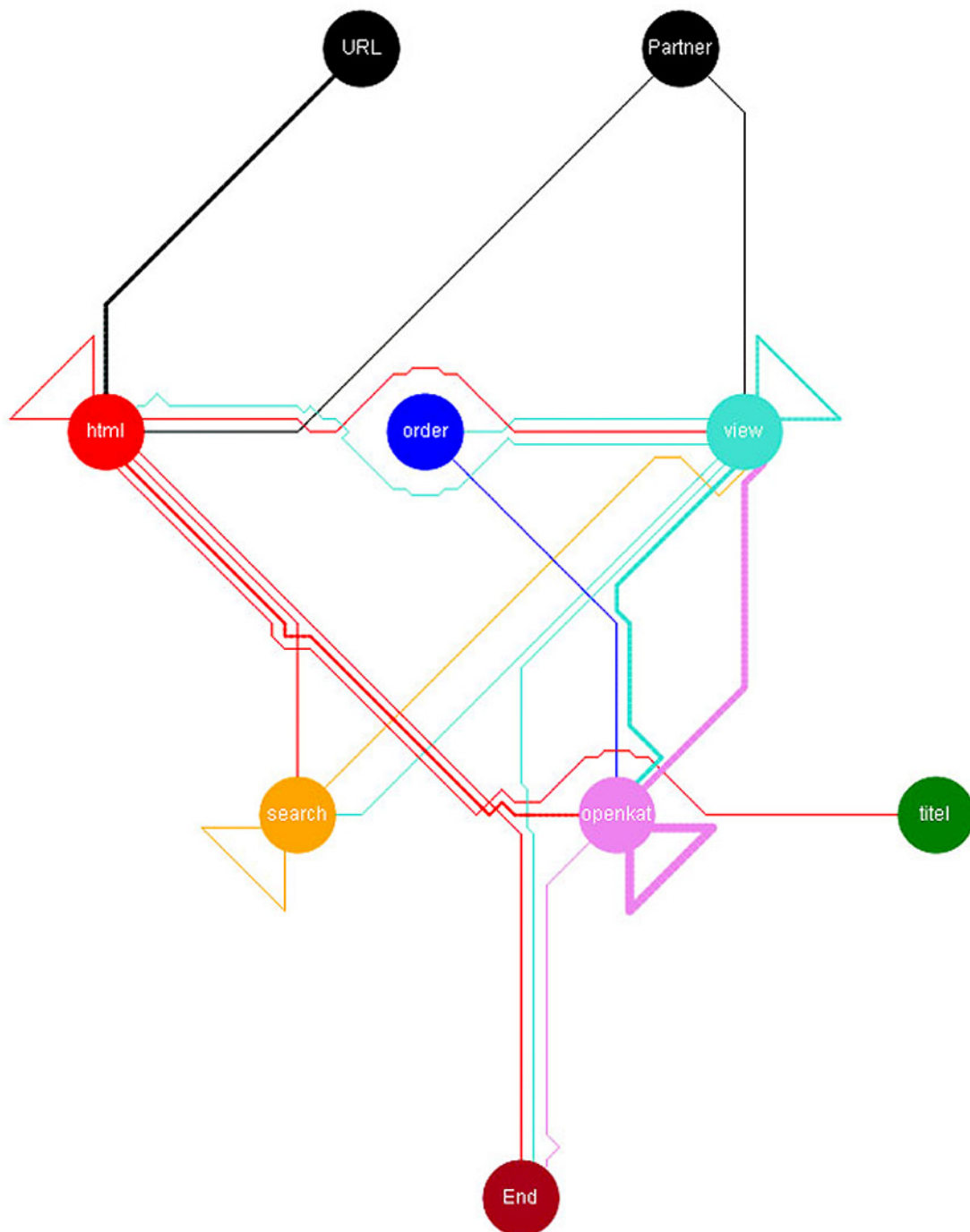


Figure 5.2: User flows in July 2005.



## 5.1 Interpretation and Evaluation of Clickstream Visualization

The second change in user behavior – reaching the online shop via forwarding from a *partner* website and only viewing product details in March while also requesting static html content in July – can be explained more easily: Partner websites were running different campaigns. While in March they concentrated on certain products, in July (as well as September, November and December) obviously additional campaigns were conducted to promote for example job opportunities, subscription to a newsletter or taking part in a competition that was presented on a static html site.

**Paradoxical Flows in Node “titel”.** It might appear paradoxical, that both in March and in July users for example click products or news featured at the entry page – denoted by flows to the *titel* node – but do not leave this node anymore. As can be seen from the table in Figure 3.3, actually they do not stay in this state but leave it in different directions that, each taken alone, do not make up more than 0.7% of the flows and therefore are excluded from displaying.

On the other hand, the self reference of the title node in March 2005 (Figure 5.1) seems to be irrational, as the frame that was showing the links logged as *titelseiteclick*, displays the desired content once a link has been clicked. Here the visualization uncovers weaknesses in navigational structure: Instead of providing a link that allows the user to go back to the contents displayed before (in this case the starting page), users need their browser’s back button.

Looking at the flows in March 2005, moreover a significant number of users requested content provided by a link on the starting page right after they have already used this category of links before (depicted by the self reference at the *titel* node). Concluding from this, one may assume that it seems reasonable to provide these links not only at the main entry page but still when users clicked one of these links.

**Insufficient Navigation Capabilities in Detailview.** Quite similar to the problem described above, the website’s navigational capabilities seem to be insufficient concerning product-detail displays. As stated for a repeated click on a link displayed at the main page, directly subsequent article-detail views (logged with the eventcode *view*) do not

## 5 Evaluation

ID	Eventcode	Eventparameter	sid	zeitstempel
2095019	search	shuttel	1382409P33P0462P40P152P6019P02P2005from	19.02.2005 09:34:55
2095023	search	shutel	1382409P33P0462P40P152P6019P02P2005from	19.02.2005 09:35:00
2095040	search	+SHUTTLE	1382409P33P0462P40P152P6019P02P2005from	19.02.2005 09:35:17

Figure 5.3: Search for Shuttle.

reflect the actual course of interaction. Showing details for a certain product means replacing the overview display on products. Choosing a different product to be displayed in detail then means to go back to the overview and click on a certain article from the list. Subsequent detail views – as implied by the log and the visualization – therefore actually are not subsequent at all, as the user has to click his browser’s back button to see the overview again before selecting a different product. Again, it seems reasonable to provide a link that brings back the user to the product-overview page.

**Insufficient Search.** Both in March (red upper left node in Figure 5.1) and July (orange lower left node in Figure 5.2) as well as in all other months, the *search* node has a self reference, pointing out that users frequently performed successive searches. Taking into consideration that the search feature does neither support searching the results of former findings nor is it designed to be tolerant considering typing errors, this feature does not seem to be sufficient. An unsuccessful search is depicted in Figure 5.3, where a user obviously tried to find products related to “Shuttle”. As he did not know how to spell the hardware manufacturer, he searched for “shuttel” – what can be seen from the first row of the *eventparameter* column. Instead of at least showing possible alternatives to the queried string “shuttel” the system returned an error message. Considering his mistake, the user adjusted the string – and now searched for “shutel” (the second row in the *Eventparameter* column), which again turns out to be incorrect spelling. Only after 17 more seconds (the difference in the timestamp for the last and the last but one row) he uses the manufacturer dropdown list (denoted by the “+” prior to the query in the last row’s *eventparameter*), provided right beneath the search field and chooses “SHUTTLE” to be searched for.

### 5.1.2 Functional Evaluation

Knowing about the website usage and potential usability problems turned out in an informal talk with the programmer responsible for developing and maintaining the online shop, finally the *Clickstream* prototype as well as the graphs generated were evaluated considering their use in everyday business. Resulting from this part of the feedback session, a list of requirements was developed, that will have to be considered when evolving the present concept, as for the moment the following features are still missing in the prototype:

- **Threshold.** At the moment the threshold used for data preparation is “hardcoded”. Although reasonable results could be achieved using a lower bound of 0.7%, the threshold for the transition frequencies will have to be scalable by the user.
- **Frequency Range.** Unlike provided in the prototype, it turned out to be interesting not only to display the most frequent transitions, but to be able to specify a range for the transition frequencies in order to also view the least frequent transitions, for example.
- **Node Decomposition.** As already mentioned above, for every class of links a common eventcode and an eventparameter is logged, saving additional information about the very link that was clicked for example. It therefore seems interesting to be able to perform further decomposition of nodes and edges by specific parameters.
- **Source Decomposition.** Similar to the node decomposition a function has to be provided that allows the split up of the whole graph by different sources the users came from (typed-in *URL*, from *partner*).
- **Labeling of Edges.** As it was hard to distinguish between edges that are – for example – drawn with a width of 1 point and those that are drawn with 1.2 points, edges should either be labeled or a tooltip should be supplied, that shows the value represented by the focussed edge.
- **Color of Nodes.** Another problem directly concerning drawing was the change in nodes’ color, that at the moment of evaluation were determined by their position in

## 5 Evaluation

the display. As the feedback showed, each nodes' color should be fixed in order to allow faster lookup of changes over time for each node, independent of its position in different displays.

- **Position of Nodes.** Related to the problem mentioned above, the system could provide an option that fixes nodes' positions over different periods. Doing so, users could develop a “mental map” of the relations between nodes over time.
- **Improvement of Graph Drawing.** Last, but definitely not least, the graph drawing itself still has a lot of potential that will have to be exploited in future work.

## 5.2 Evaluation Parallel Coordinated Hierarchies

In order to evaluate the *Parallel Coordinated Hierarchies* prototype a user test was performed. To give an overview on what has been done and what the outcomes were, this section is organized as follows: First, the test setup, the methods and the three scenarios used in the test will be presented, followed by an introduction of the conventional tools used by the test user so far. Next, outcomes and consequences of the evaluation will be discussed before finally turning to conclusions. Suggestions made by the user with respect to improving the prototype can be found in Chapter 6.

### 5.2.1 Test Setup and Methods

Basically, the evaluation had three main goals: First, it should show, which of the features realized in the prototype are helpful in successfully reaching the scenario goals and which of them are not needed at all. Secondly, going through these scenarios helped the user to get familiar with the prototype which was necessary for qualified feedback surveyed with a questionnaire subsequent to the test. Thirdly, the test was designed to show the differences between the prototype's features and that of already existing tools. Considering these objectives, the scenarios were designed to make the user work with the most important features at first. Secondly they should show the user the prototype's capabilities with respect to the link between the figures in the *Hierarchical List* and their graphical representation in the *Parallel Coordinates*. Only in third place the scenarios' relevancy for daily work was considered.

The three scenario tasks had to be fulfilled using existing tools before answering the questions using the prototype. They were organized in ascending degree of difficulty in order to take advantage of the user's learning rate. During all tests the *Thinking Aloud Method* [Nie00, Wik] was used, where the user is asked to make comments on whatever he is seeing, doing, trying and thinking while interacting with the interface. After reading the task to the user and starting the scenario, the conversation was restricted to information about the scenario itself. As the prototype uses DiTech's sales data, the test had to be limited to only one user – the chief executive officer (CEO) of DiTech – who is familiar with

## 5 Evaluation

the data used and already uses analysis tools that could be compared to the prototype. In order to provide the environment for both the prototype and existing tools, the evaluation was carried out in the CEO's office where one screen displayed the user's conventional tools and a second one was used for the prototype. The whole test was filmed with a video camera to provide an audio and a video documentation of the evaluation.

### 5.2.2 Procedure

The procedure for the evaluation was as follows: First, the user was presented the prototype and its features using test data. After that, a 5 minutes free testing period allowed the user to get familiar with the prototype. In this phase real data was used already. Next the scenario test was conducted as described above. Finally the user's experience and suggestions were surveyed with a questionnaire.

### 5.2.3 Scenarios

Each of the three scenarios consisted of a task, defined by a scenario goal and a certain amount of time in which this goal should have been reached. The time that was provided to fulfil each task was approximately four times the duration it took the test engineer to reach the scenario goal while describing what he was doing. In order not to restrict the user in his actions, the description did not contain any hint on how to fulfill this task.

**Scenario 1 – Dimensions' Correlation.** Scenario 1 was designed to make the user get a sense for the meaning of polygonal lines in *Parallel Coordinates* and the relations between polylines and axes. Therefore correlations between the values on the four axes had to be pointed out within two minutes. In order to make sure, that the user really understands what the task is about, an exemplary answer (“Whenever the value for return-on-investment is high, so is the value for turnover and the other way round”) was given.

**Scenario 2 – Extreme Values.** Scenario 2 tried to make the user familiar with changing hierarchies and visually exploring data using *Parallel Coordinates*. The task defined was to find extreme values in payment groups and payment methods, whereas there was no restriction on the number of groups and methods that had to be found. The time limit for this task was three minutes.

**Scenario 3 – Find Similar Quadruples.** As the last scenario concentrated on the combination of visual exploration and the directed use of features, the task was distinctly more complex than the two before. Here the user had four minutes to find quadruples, that were as similar as possible in all four dimensions to the quadruple of the “transfer within 2 weeks” in January 2006 in Vienna.

### 5.2.4 Current Analysis Tools

Analyzing business data so far is mainly done using three different kinds of applications:

- First, there is the so called “Onliner” tool, that allows predefined aggregations on turnover, contribution and return-on-investment per time period as well as detailed views on single products or orders for example.
- The second tool used is an intranet solution for predefined analyses concerned with creating *Hierarchical Lists* on aggregated sales per time period and payment method – just like the one used in the prototype – or on shipping method. Other analyses done with this tool are detailed article views, concerned with the development of purchase and sales prices, the number of units in stock or the average consumption of products per day and week respectively.
- Thirdly, DiTech’s CEO uses Microsoft<sup>®</sup> Access<sup>™</sup> for predefined reports on purchases and sales as well as to query the main database.

Neither of these solutions offer visualization (apart from charts displaying price development) nor do they allow interaction in terms of reaggregating data in a comfortable way.

## 5 Evaluation

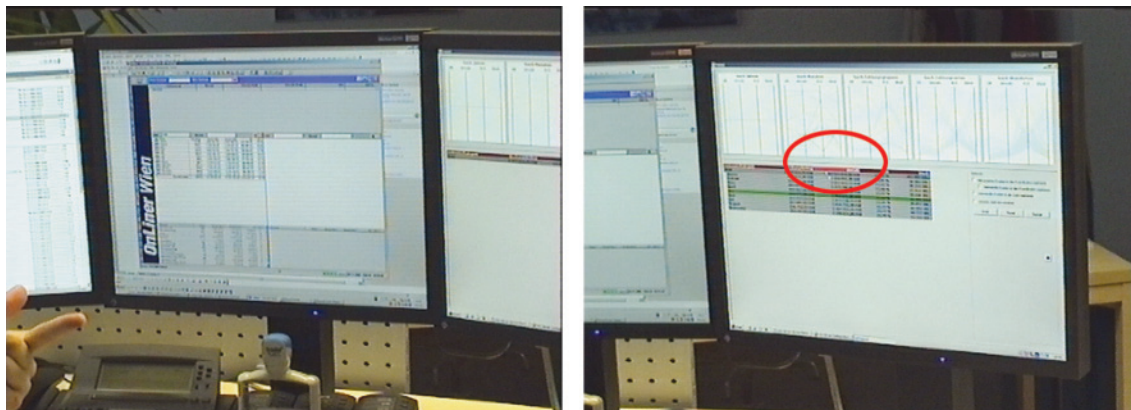


Figure 5.4: Answering questions in scenario 1. Answering questions not only takes existing tools but the user's hand to point out dependencies with current tools (left). On the right, the user performs a change of dimensions' position (feature C2) to visually display dependencies in underlying data with the prototype.

Regarding the user's knowledge of data used in the evaluation, we may assume that he knows "his" business data very well. As it turned out especially in scenario 1 and 2, he could easily have answered the questions without using any supporting analysis tools. With respect to displaying data in a *Hierarchical List* he also may be seen as an expert, as this kind of representation already was part of his analysis tools before introducing the prototype. *Parallel Coordinates* displays on the other hand were new to him.

### 5.2.5 Execution and Outcomes

After having defined methods, procedures, scenarios and describing the tools the prototype will be compared to, we will now turn to the execution of the evaluation. First, outcomes of the scenario tests will be presented before discussing general consequences and results of the questionnaire. Any of the prototype's features mentioned in the following paragraphs will be referred to with its description and its identifier (see Section 4.4).

Fulfilling the task in **Scenario 1** did not cause any problems at all as the test user was able to answer the correlation question even without using his existing tools. The prototype here turned out to be helpful in terms of displaying exactly what the test user knew in advance: The higher the number of products sold, the higher the turnover and the higher the contribution margin. Moreover the visualization clearly confirmed, that



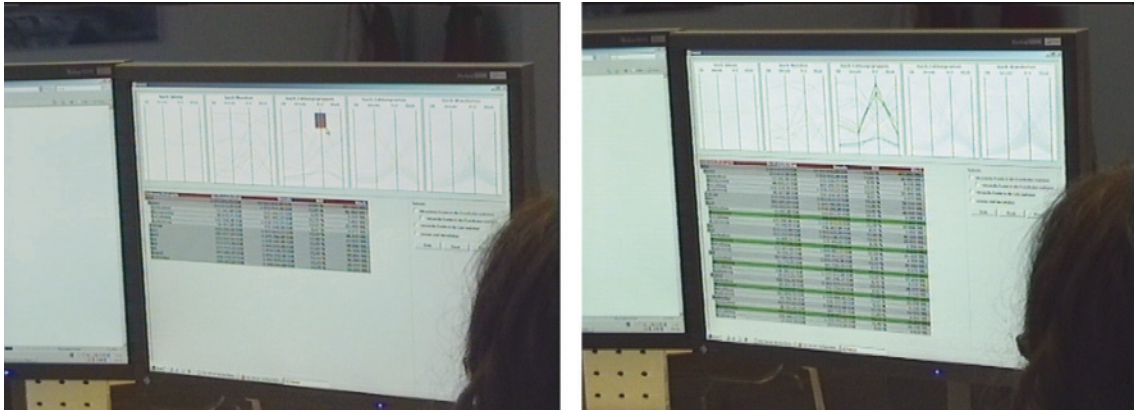


Figure 5.5: User’s initial difficulties in scenario 2. Looking for the best performing payment group the user brushes an upper axis region (feature B2) without changing hierarchies (feature C1) first (left). The result is unsatisfying (right): instead of the highest aggregate return-on-investment the list shows the best performers at a monthly aggregation level.

the return-on-investment rather depends on season and location than on the other three dimensions. Quite intuitively the user did what was needed to really see dependencies: he grouped the three correlated axes next to each other by permuting the labels on top of the *Hierarchical List* (feature C2, see Figure 5.4). Although existing tools produced the same result as the prototype, the *Parallel Coordinated Hierarchies* approach facilitated interpreting: While it took about four minutes to describe correlations and seasonal variation in the return-on-investment dimension and confirm these assumptions by going through the monthly values, it was only about two minutes – the time scheduled for this task – when using the prototype.

In **Scenario 2** – just like before – the user exactly knew what payment groups and methods were the best performing. Using his own tools he could not find the overall highest return-on-investment in a certain group and method within three minutes as these figures were only available grouped by month. The same was true for answering this question using the prototype at first: As the default hierarchical order is year, month, payment group, payment method and location, he could not immediately find a way to point out the best performing group and method. Starting with the default view, he tried to brush the best performing payment groups and payment methods (feature B2) without changing hierarchies (feature C1). As a consequence, it were not aggregated values for group or method he identified but groups (and methods respectively) that only did best in

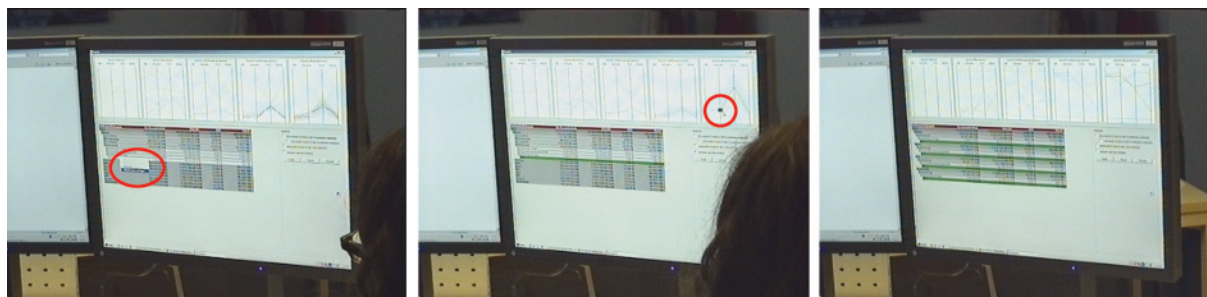


Figure 5.6: The right way through scenario 3. On the left the user marks the initial list node in *Parallel Coordinates* (feature B3). In the middle he zooms regions around the initial data on different axes (feature B4). On the right the result of highlighting similar nodes is displayed.

certain months (see Figure 5.5). Realizing this problem, he found something was missing: “There is a level missing in between the year hierarchy and the month hierarchy. It should be possible to skip levels to explicitly tell how a payment group performed in a year.” Only after breaking the rules and reminding the test user that it is possible to display hierarchies in any desired level in principle, he recalled how to change hierarchies (C1) and was able to answer the questions immediately. Instead of brushing the corresponding polylines in *Parallel Coordinates* (B1 or B2) he identified the best performing group and method by looking them up in the list. The user’s summary of the scenario reflected this circumstance: “The change of hierarchies is a very good thing.”

Finally, in **Scenario 3** the most complex task had to be fulfilled where the user was not able to answer the question about similar quadruples given one specific quadruple with his existing tools. In contrast to that, he acted goal oriented when using the prototype: At the beginning, he marked the initial node in the hierarchical list to be highlighted in *Parallel Coordinates* (feature B3). After this, he tried to highlight similar values by following the initial value’s polyline (H4). As soon as a subset of polylines was highlighted that approximately met the requirements, he marked the respective nodes in the list by a right click in *Parallel Coordinates* (B1). As the resulting sample values were similar just with respect to the axis, the right click was performed on, he wanted to switch to zooming into data around the values of the initial node (B4). After brushing nodes at first (B2), using the *Undo* button (G5) and a short period of recalling how to change from brushing to zooming, he selected the checkbox that enables zooming and finally selected an area surrounding the initial value at the return-on-investment axis and the turnover

axis (B4). Finally, a right click at the initial polyline (B1) highlighted the desired similar nodes in the list. Although the user limited the zoom to just two of the four axes he got a reasonable result – containing the two nodes that were expected to be in the sample plus a third node that slightly deviated in contribution margin and number of products sold.

### 5.2.6 Survey

The survey carried out subsequent to the scenario tests finally showed, that – above all – it was the combination of both elements, the *Hierarchical List* and the *Parallel Coordinates* plots (PCPs), that helped most in answering the questions. Although fulfilling tasks in the scenario test did not convey the impression, the user stated that he found displaying data in PCPs not that clearly comprehensible. Having the directed use of features as presented above in mind, we still may conclude from this, that *Parallel Coordinates* alone do not seem to be sufficient for this kind of analysis but added value to the (familiar) concept of the *Hierarchical List* in terms of providing means of visual interaction with the list.

Looking at single features that facilitated finishing tasks successfully, the test user named

- the change of hierarchies (feature C1),
- zooming certain segments on axes (feature B4)
- and brushing values in *Parallel Coordinates* by marking nodes in the list (feature B3)

as most helpful in respect to the scenarios. The only features never used were those concerned with highlighting related nodes in the list (H2) and in the coordinates (H6) respectively. Considering the practical relevance of the scenarios, it was scenario 3 that seemed the least interesting for running everyday business. The questions in scenario 1 and 2 on the other hand have already been raised before. Although DiTech’s CEO did not gain new insights in sales data, he declared that he would replace certain existing tools with the prototype. This might – just like mentioned above – also be due to the fact, that current

analysis tools aggregate and organize data the way the *Parallel Coordinated Hierarchies* do – with the restriction, that they only supply limited functionality of interaction. Finally, desirable additional features that are currently not included in the prototype are sorting the list by any attribute, an export feature that allows generating spreadsheets and a fullscreen display for the *Parallel Coordinates* (for details see Chapter 6)

### 5.2.7 Test Summary and Conclusions

To sum up, the evaluation showed, that

- having a test user who has underlying data, their meanings and relations in mind turned out to be an advantage, as the prototype’s concept was easier to understand for him after all.
- conducting the test with someone who knows underlying business data very well, the prototype was rather used to show answers to the scenarios’ questions than to answer the questions using the prototype.
- the user’s problems that occurred during the test may mainly go back to the limited time for instructions.
- the scenarios designed for the test could be answered more efficiently using the prototype.
- as the survey confirmed, it was the combination of visually exploring data and interacting with the *Hierarchical List* that were found helpful.

## 6 Conclusions and Future Work

The last chapter of this work is concerned with first summing up what has been done, before discussing conclusions that may be drawn from the approaches presented. Finally we will give an overview on future work that seems important when further elaborating the two concepts.

### 6.1 Summary

Chapter 1 showed, that information visualization provides means of analyses that enable visual exploration of data. As analyzing, engineering, implementing, documenting and testing especially in this field is resource demanding, we drafted five different aims of analysis at first but limited the presented work's focus to two of them: To *Clickstream Visualization* on the one hand and analyses and visualization of sales data on the other.

Chapter 2 presented existing approaches considering data collection, data preparation and *Clickstream Visualization* at first. For the novel approach of *Parallel Coordinated Hierarchies* – the linked view consisting of a *Hierarchical List* and *Parallel Coordinates* displays for each dimension – a brief introduction to the common concept of *Hierarchical Lists* was given before the state-of-the-art considering drawing and interacting with *Parallel Coordinates* was documented.

The first concept was presented in Chapter 3, where our concept on *Clickstream Visualization* was introduced. Thereby, visualizing clickstreams was based on an existing approach presented by Brainerd and Becker [BB01]. The main aim of this chapter was to proof the concept with data collected in DiTech's online shop.

## 6 Conclusions and Future Work

Chapter 4 was dedicated to the introduction of the *Parallel Coordinated Hierarchies* approach. First, the systematics and meaning of both the list and the coordinates display were discussed before the general approach was presented. After that we gave an overview on all features including general features, features concerned with transient highlighting and permanent brushing nodes and polylines as well as changing of axes and hierarchies.

In Chapter 5 finally, both concepts were evaluated. First, we discussed the meaning of the displays generated by the *Clickstream Visualization* prototype and possible usability problems that were revealed. As also could be shown in the short user evaluation, carried out through an informal talk, the prototype's current capabilities already allow insights in users' behavior although still a lot of work will have to be done, to extend features of interaction and drawing. The end of this chapter was dedicated to the documentation of the user evaluation of the "Parallel Coordinated Hierarchies" carried out with DiTech's managing director. As could be seen here, this novel approach adds value to analyzing business data with a conventional *Hierarchical List* in terms of enabling the user to visually interact with the data.

## 6.2 Conclusions

Having introduced and evaluated *Clickstream Visualization* and *Parallel Coordinated Hierarchies* we now can draw conclusions from both approaches. First, we again will look at the graphical representation of user flows in the online shop before reflecting on the concept of sales data analysis.

### 6.2.1 Clickstream Analysis and Visualization

As we have seen when collecting feedback from the responsible person for programming and maintaining the online shop, the concept of *Clickstream Visualization* used in this thesis was confirmed to be a promising approach. Just presenting the graphs, the prototype generated seemed sufficient to have the programmer reflect about the website's structure and changes in navigational behavior over time. Although the presented work

tried to proof the concept presented by Brainerd and Becker [BB01], discussed in the related work chapter 2, the level of implementation stayed behind what has been done in their work. This is due to limitations of this work that had its focus on two concepts. On the other hand, it was interesting to see that the test user demanded many of the features realized in their approach for future elaboration of the prototype.

Looking at the (simple) displays used for evaluation, we still were able to show main characteristics of user flows on the pages provided by the online shop. Weaknesses in the search feature could be identified in terms of users' repeated searching when not exactly knowing how to spell keywords. What could be confirmed was the assumption, that most users browse the website to gather information on certain products and product groups, as flows in the hierarchical menu and the detailed view of single articles made up nearly 60% of all traffic in March 2005 for example.

As more detailed analyses seem to be helpful in terms of gaining deeper insights in the website's potential of improvement, the approach of *Clickstream Visualization* will have to be elaborated as proposed in Section 6.3 in order to provide a set of features that allow interaction.

### 6.2.2 Parallel Coordinated Hierarchies

In the evaluation (Section 5.2) the more elaborated approach of *Parallel Coordinated Hierarchies* turned out to add value to conventional tools in use so far. As DiTech already performs sales analyses in the form of *Hierarchical Lists* the test user was able to use the prototype after a short introduction and found it helpful to visually interact with data represented by the list.

Concluding from this, we may state that the linked view fulfilled expectations: *Parallel Coordinates* are able to display multidimensional data in a compact, two dimensional plane, which was needed to depict five levels of hierarchy and four dimensions. The hierarchical structure moreover allowed a (pre-)aggregated view of the data. Changing hierarchies via permuting the order of *Parallel Coordinates* plots (PCPs) supported re-aggregation of data and by that different levels of abstraction which can be done intuitively

## 6 Conclusions and Future Work

with the prototype. Visually exploring and searching data as well as the synchronous displays considering the list and the PCPs help gaining a mental model of the sales data. Up to that point these data were clustered in long lists that – although structured – did not allow overviewing data on different levels of aggregation at all.

Though, as will be discussed in more detail in Section 6.3 (Future Work), there are some enhancements that were requested. After all, the approach of *Parallel Coordinated Hierarchies* as presented here – including some extensions – is a promising, practice-oriented application of *Parallel Coordinates*.

### 6.3 Future Work

The very last section of this thesis gives an outlook on future work. Both *Clickstream Visualization* and *Parallel Coordinated Hierarchies* need to be elaborated in order to satisfy everyday business' needs. Moreover there are – from a scientific viewpoint – fields of interest that should be covered in more detail. The following paragraphs are structured as follows: First, possible extensions and fields of further elaboration of the *Clickstream Visualization* will be discussed before developing perspectives for future work on the *Parallel Coordinated Hierarchies*.

#### 6.3.1 Clickstream Analysis and Visualization

For clickstream analysis and visualization – as for most graphical representations – providing means of interaction appears to be crucial. As this dimension has been left aside in the prototype realized, the proposals on future work will concentrate on possible ways of interaction. As mentioned in the short user evaluation section in Chapter 5.1.2, additional features to be realized were collected in this feedback session on the one hand. On the other hand, techniques presented in Related Work (Section 2.1) act as a model for the proposals discussed below.



**Graph Drawing.** Maybe the most important thing that has to be improved is graph drawing. As we have seen in Section 3.4 at the moment the prototype only generates simple graphs, not optimizing edge crossing and only marginally optimizing node placement. In order to be able to display more than the suggested six nodes (apart from starting and end states) more efficient algorithms will have to be implemented.

Moreover, labeling edges appears to be crucial for better understanding the graph. Either this could be done by simply writing overall relative transition frequencies to the respective edges or for example by adding histograms to the nodes that give further information on the number of users traveling a specific path.

Another feature with respect to graph drawing that will have to be considered is coloring of nodes. As could be seen from the user feedback, coloring nodes according to their position in the display does not seem to be supportive for the development of a mental model. In contrast to the approach realized now, the nodes' colors will therefore have to be fixed for different images. In addition to that, the users should be able to fix the nodes' positions for several images.

**Variable Threshold.** After having optimized graph drawing, the application should allow displaying a variable number of nodes. A feature that may make use of it is varying the threshold, that at the moment is hardcoded at a level that seemed reasonable for analysis as well as visualization.

**Variable Analysis Period.** Analyses at the moment are limited to (hardcoded) monthly values. As users may want to analyze periods different than that, the users should be able to vary the time the analysis is made for.

**Path Decomposition.** As it turned out in the user evaluation and as implemented by Brainerd and Becker [BB01], decomposition of users by different starting points or structural characteristics like gender, residence, and income could add value to analyzing the users' behavior.

**Further User Studies.** Another topic to be mentioned for future work on *Clickstream Visualization* is the demand for further user studies, once the prototype has been evolved.

### 6.3.2 Parallel Coordinated Hierarchies

Finally, the last paragraphs take a look at future work on *Parallel Coordinated Hierarchies*. As for the analysis of clickstreams and their visualization the section will subsume requests collected in the user evaluation as well as those deduced from approaches presented in Related Work in Section 2.2.

**Display Extensions.** As presented in Related Work in Section 2.2 for *Parallel Coordinates* the *Parallel Coordinated Hierarchies* should provide further (statistical) information on the distributions on each of the axes – in terms of displaying quartiles [Sii00] as well as optional histograms that superimpose axes [OL96, HLD02]. In order to better overview data in the crowded displays in lower level hierarchies, some kind of clustering – like the visual abstraction presented by Novotny [Nov04] and the approach of *Hierarchical Parallel Coordinates* [FWR99] – might add further value to the approach.

**Functional Extensions.** Desirable additional features that are currently not included in the prototype were collected in the user evaluation. The first feature that seemed interesting here was the ability to sort the list by any given attribute. Secondly an export feature that allows generating spreadsheets from data displayed or brushed was demanded by the test user. It will be crucial to offer features considering hierarchies. For example it was conceivable to supply an export feature for every single *Parallel Coordinates* plot (PCP), adding data to spreadsheets that all have the same level of aggregation.

Moreover, a fullscreen display for the *Parallel Coordinates* was considered to be helpful, where each of the PCPs may be displayed in a separate window. Interacting with coordinates then could be facilitated, allowing even more accurate brushing of single values. The last functional enhancement demanded by the test user was the extension to more domains of analysis like aggregated values by time and shipment method for example.

**Brushing Extensions.** As presented by Martin and Ward [MW95], users should be able to combine single brushing operations by the logical operators “and”, “or” and “xor”. At the moment, these operations (apart from “xor”) are implicitly supplied by zooming values (feature B4 in Section 4.4), as this operation is designed to work cumulatively. Brushing moreover could be extended by different colors for different, subsequent brushing operations, that may additionally be facilitated by providing delimiters on the axes like shown by Siirtola [Sii00] with small, variable arrows on the brushed axis.

Related to this, brushing operations initiated by node clicks will have to be investigated more intensely. For example, unbrushing nodes at the moment may still be misleading: either it can be done by a double click on the node – which unbrushes one single node – or by choosing the respective option from the popup menu – which unbrushes the focussed node and all of his successors.

**Further User Studies in Different Domains.** Finally, further user studies seem necessary. In contrast to *Clickstream Visualization* it seems very interesting for *Parallel Coordinated Hierarchies* to proof the concept in other fields that provide a considerably high amount of data. For example, it was challenging to examine the current work with flow data from a purification plant, where reporting now has to be structured either by single measures of certain parameters over time or by displays showing certain plant aspects for a given moment. *Parallel Coordinated Hierarchies* here could combine views on plant facilities and their parameters in one single display.

## 6 *Conclusions and Future Work*

# Acknowledgments

I am most grateful to Tom Theußl and to my supervising professor Master Eduard Gröller for their incitements, motivation and suggestions, for worthwhile discussions and for answering any of my questions throughout the diploma thesis.

I would also like to thank Damian Izdebski for the ideas he contributed to this work and the time he spent evaluating the project as well as Rainer Gruber for his feedback and suggestions to the project.

Finally, I want to thank my late informatics teacher Manfred Meister who first engaged my interest in this field and my family for their support throughout my study. To them I dedicate this work.

## *Acknowledgments*

# Bibliography

- [AGJ<sup>+</sup>00] Jesper Andersen, Anders Giversen, Allan H. Jensen, Rune S. Larsen, Torben Bach Pedersen, and Janne Skyt. Analyzing Clickstreams Using Subsessions. Technical Report 00-5001, Dept. of CS, Aalborg University, <http://www.cs.aau.dk/~tbp/Teaching/DAT5E01/andersenfinal.pdf>, 2000.
- [BB01] Jeffrey Brainerd and Barry Becker. Case Study: E-Commerce Clickstream Visualization. In *IEEE Symposium on Information Visualization 2001*, Proceedings, pages 151–155, 2001.
- [CBC95] Carlos R. Cunha, Azer Bestavros, and Mark E. Corvella. Characteristics of WWW Client-based Traces. In *Technical Report TR-95-010, Computer Science Department*. Boston University, 1995.
- [CHM<sup>+</sup>00a] Igor V. Cadez, David Heckerman, Christopher Meek, Padhraic Smyth, and Steven White. Model-Based Clustering and Visualization of Navigation Patterns on a Web Site. Technical Report MSR-TR-00-18, Microsoft Research, Redmond, WA, USA. <http://www.datalab.uci.edu/papers/webcanvas.pdf>, 2000.
- [CHM<sup>+</sup>00b] Igor V. Cadez, David Heckerman, Christopher Meek, Padhraic Smyth, and Steven White. Visualization of Navigation Patterns on a Web Site Using Model-Based Clustering. In *Knowledge Discovery and Data Mining*, pages 280–284, 2000.
- [CK97] Y. Chen and E. Koutsofios. WebCiao: A Website Visualization and Tracking System. In *Proceedings of WebNet 97 Conference*, Toronto, Canada, 1997.

## Bibliography

- [Dav99] Brian D. Davison. Web Traffic Logs: An Imperfect Resource for Evaluation. In *Proceedings of the Ninth Annual Conference of the Internet Society (INET '99)*, San Jose, 1999.
- [Die01] Boris M. Diebold. Usage-based Visualization of Web Localities. Diploma Thesis at the Eberhard Karls University Tübingen, Division for Parallel Computing, 2001.
- [DK01a] Mukund Deshpande and George Karypis. Selective Markov Models for Predicting Web-Page Accesses. In *Proceedings SIAM Int. Conference on Data Mining (SDM'2001)*, 2001.
- [DK01b] Boris M. Diebold and Michael Kaufmann. Usage Based Visualization of Web Localities. In Peter Eades and Tim Pattison, editors, *Australian Symposium on Information Visualization, Conferences in Research and Practice in Information Technology, Vol. 9*, pages 159 – 164, Sydney, Australia, 2001.
- [Eic01] Stephen G. Eick. Visualizing online activity. *Commun. ACM*, 44(8):45–50, 2001.
- [FS98] Emmanuel Frecon and Gareth Smith. WebPath - A Three-Dimensional Web History. In *IEEE Symposium on Information Visualization 1998*, Proceedings, pages 3–10, 1998.
- [FSMB99] Adil Faisal, Cyrus Shahabi, Margaret McLaughlin, and Frederick Betz. In-site: Introduction to a Generic Paradigm for Interpreting User-Web Space Interaction. In Cyrus Shahabi, editor, *ACM CIKM'99 2nd Workshop on Web Information and Data Management (WIDM'99), Kansas City, Missouri, USA, November 5-6, 1999*, pages 53–58. ACM, 1999.
- [FWR99] Ying-Huey Fua, Matthew O. Ward, and Elke A. Rundensteiner. Hierarchical Parallel Coordinates for Exploration of Large Datasets. In *IEEE Visualization 1999*, Proceedings, pages 43–50, 1999.
- [GK03] Martin Graham and Jessie Kennedy. Using Curves to Enhance Parallel Coordinate Visualisations. In *IEEE Symposium on Information Visualization 2003*, Proceedings, pages 10–16, 2003.



- [GSB<sup>+</sup>94] M. Guzdial, P. Santos, A. Badre, S. Hudson, and M. Gray. Analyzing and Visualizing Log Files: A Computational Science of Usability. In *Georgia Institute of Technology, GVU Technical Report; GIT-GVU-94-08*. 1994.
- [HLD02] Helwig Hauser, Florian Ledermann, and Helmut Doleisch. Angular Brushing of Extended Parallel Coordinates. In *IEEE Symposium on Information Visualization 2002*, Proceedings, pages 127–130, 2002.
- [HS99] Harry Hochheiser and Ben Shneiderman. Understanding Patterns of User Visits to Web Sites: Interactive Starfield Visualizations of WWW Log Data. In *Proceedings of the ASIS Annual Meeting*, pages 331–344, American Society for Information Science, Washington, D.C., USA, 1999.
- [ID90] Alfred Inselberg and Bernard Dimsdale. Parallel Coordinates: A Tool for Visualizing Multidimensional Geometry. In *IEEE Visualization 1990*, Proceedings, pages 361–378, San Francisco, 1990.
- [JLJC05] Jimmy Johansson, Patric Ljung, Mikael Jern, and Matthew Cooper. Revealing Structure within Clustered Parallel Coordinates Displays. In *IEEE Symposium on Information Visualization 2005*, Proceedings, pages 125–132, 2005.
- [KE02] T. Alan Keahey and Stephen G. Eick. Visual Path Analysis. In *IEEE Symposium on Information Visualization 2002*, Proceedings, page 165, 2002.
- [MW95] Allen R. Martin and Matthew O. Ward. High Dimensional Brushing for Interactive Exploration of Muttivariate Data. In *IEEE Visualization 1995*, Proceedings, pages 271–278, 1995.
- [Nie00] Jacob Nielsen. *Usability Engineering*. Morgan Kaufmann, San Diego, CA, USA, 2000.
- [Nov04] M. Novotny. Visually Effective Information Visualization of Large Data. In *8th Central European Seminar on Computer Graphics (CESCG 2004)*, Proceedings, 2004.

## Bibliography

- [OL96] Hwee-Leng Ong and Hing-Yan Lee. Software Report: WinViz – A Visual Data Analysis Tool. In *Computers & Graphics*, 20(1), pages 83–84, 1996.
- [Pro] Apache HTTP Server Project. Log Files. <http://httpd.apache.org/docs/1.3/logs.html>. cited 2006-11-17.
- [SFKF00] Cyrus Shahabi, Adil Faisal, Farnoush Banaei Kashani, and Javed Faruque. INSITE: A Tool for Real-Time Knowledge Discovery from Users Web Navigation. In *Proceedings of the 26th International Conference on Very Large Databases*, pages 635–638, Cairo, Egypt, 2000.
- [Sii00] Harri Siirtola. Direct Manipulation of Parallel Coordinates. In *IEEE Symposium on Information Visualization 2000*, Proceedings, pages 373–378, 2000.
- [SM00] Heidrun Schumann and Wolfgang Müller. *Visualisierung - Grundlagen und allgemeine Methoden*. Springer, Berlin, Heidelberg, 2000.
- [Sof] Tom Sawyer Software. <http://www.tomsawyer.com>. Cited 2006-11-20.
- [The01] Holger Theisel. CAGD and Scientific Visualization. Habilitation thesis. University of Rostock, Computer Science Department, 2001.
- [Uni] Uppsala Universitet. Access Log Analyzers. <http://www.uu.se/Software/Analyzers/Access-analyzers.html>. cited 2006-11-17.
- [W3C] W3C. RFC2616, Section 10, Status Code Definitions. <http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>. Cited 2006-11-10.
- [Weg99] Edward J. Wegman. Data Mining and Visualization: Some Strategies. In *Bulletin of the International Statistical Institute Nr. 52*, pages 223–226. 1999.
- [WHS<sup>+</sup>02] Sarah J. Waterson, Jason I. Hong, Tim Sohn, James A. Landay, Jeffrey Heer, and Tara Matthews. What Did They Do? Understanding Clickstreams with the WebQuilt Visualization System. In *Proceedings of Advanced Visual Interfaces 2002*, Trento, Italy, 2002.
- [Wik] Wikipedia the Free Encyclopedia. Thinking Aloud Protocol. [http://en.wikipedia.org/wiki/Think\\_aloud\\_protocol](http://en.wikipedia.org/wiki/Think_aloud_protocol). Cited 2006-11-22.

- [WL96] Edward J. Wegman and Qiang Luo. High Dimensional Clustering Using Parallel Coordinates and the Grand Tour. Technical Report 124, George Mason University Fairfax, VA, Center for Computational Statistics, Fairfax, Virginia 22030, U.S.A., 1996.
- [WLG97] Rainer Wegenkittl, Helwig Löffelmann, and Eduard Gröller. Visualizing the Behavior of Higher Dimensional Dynamical Systems. In *IEEE Visualization 1997*, Proceedings, pages 119–126, 1997.
- [WP06] Marco Winckler and Florence Pontico. A Model-Driven Architecture for Logging Navigation. In *Workshop on Logging Traces of Web Activity: Workshop on the Mechanics of Data Collection, Co-located with 15th International World Wide Web Conference (WWW2006)*, Edinburgh, Scotland, 2006.
- [WYB98] Kun-Lung Wu, Philip S. Yu, and Allen Ballman. SpeedTracer: A Web Usage Mining and Analysis Tool. *IBM Systems Journal*, 37(1):89–105, 1998.
- [XCAC05] Yang Xiang, Michael Chau, Homa Atabakhsh, and Hsinchun Chen. Visualizing Criminal Relationships: Comparison of a Hyperbolic Tree and a Hierarchical List. In *Decision Support Systems (DSS)*, 41(1), pages 69–83. 2005.