SellTrend: Inter-Attribute Visual Analysis of Temporal Transaction Data

Zhicheng Liu, John Stasko, *Member, IEEE*, and Timothy Sullivan

Abstract—We present a case study of our experience designing SellTrend, a visualization system for analyzing airline travel purchase requests. The relevant transaction data can be characterized as multi-variate temporal and categorical event sequences, and the chief problem addressed is how to help company analysts identify complex combinations of transaction attributes that contribute to failed purchase requests. SellTrend combines a diverse set of techniques ranging from time series visualization to faceted browsing and historical trend analysis in order to help analysts make sense of the data. We believe that the combination of views and interaction capabilities in SellTrend provides an innovative approach to this problem and to other similar types of multivariate, temporally-driven transaction data analysis. Initial feedback from company analysts confirms the utility and benefits of the system.

Index Terms—investigative analysis, transaction analysis, information visualization, multiple views, time series data, multiple attributes, categorical data.

1 Introduction

Multi-variate transactional data is ubiquitous in many domains such as science, business, logistics and engineering. Time is typically also a fundamental component of transaction data. Streaming data cases accumulate and contain interesting patterns to be revealed. Well-designed interactive visualization coupled with computational analysis can help users find such patterns and generate insights to shed light on problems related to the data. In this paper we describe our experience designing and developing an interactive visualization system, SellTrend, in collaboration with Travelport, a global travel information service provider. We characterize the involved data to be *multi-variate event sequences* with strong *categorical* and *temporal* characteristics. By combining temporal linear visualization, historical trending analysis and faceted browsing techniques, we show that our system is useful in saving time and money and helping company analysts make sense of their data. Although this is a domain-specific case study, the method can be generalized for similar datasets to identify interesting combinations and relationships between attribute values in user-defined time ranges.

The contributions of the work are:

- We introduce the design of a visualization system for temporal transaction data. Our work provides a perspective on problems where temporal analysis and faceted browsing are integrated to analyze relationships between categorical attribute values that are of interest to the users. We speculate that this design will be useful in other domains with similar data and problems.

- We present a case study of a real world challenge where information visualization and visual analytics techniques are being successfully applied on huge datasets with considerable complexity.

2 Understanding the Data and the Problem

2.1 Sales Failures in Flight Transactions

As a flight reservation coordinator, Travelport plays the role of a "middle man" between travel agents or web-based travel services and airlines. Travelport acquires flight information from various airlines and makes it available to travel agencies. When a customer contacts a travel agent and tries to book a flight, the travel agent sends a flight availability query to Travelport. This query contains information about the customer's preferences on attributes such as operating airline, departing city, arriving city, time of departure and so on. Travelport finds any possible matches and offers the results as available flight options at given prices to the agent and the customer. When the customer decides to book one of these flights, the agent sends a reservation request to Travelport, which then checks with the relevant airline. Usually the airline will confirm the booking and complete the reservation transaction successfully, but it is also possible that customers are not able to actually purchase one of the offered flights. Such a failed transaction is often called a "sell failure" by Travelport analysts (Figure 1).

Failures are undesirable because they result in a loss of revenue for Travelport and they occur for a variety of reasons. First, airlines change fare and availability information frequently to maximize their profits. Second, there are usually only limited seats available especially at reduced fares and many travel agencies may be trying to book them. A seat shown available to one agent a few minutes earlier may have been booked by another agent. Alternately, a purchasing entity such as a travel agent may deploy new software that performs availability queries incorrectly and causes problems in the system. It is also possible that agents use the Travelport system in an inefficient manner by directly booking a flight without querying its availability status first. In other cases, failures may be caused by internal system errors involving parameters such as a particular airline, a particular flight class, a particular departure and destination city, or a combination of these. In any of these cases, the earlier the problems are identified, the higher the chances are that they can be fixed and this leads to an improvement in revenue and profit for the company.

Even though failed transactions only constitute a small portion of all the reservation requests, they still pose a great challenge in terms of data scalability. The company stores failures of the latest 40 days for in-depth analysis and abnormality identification, and there are many millions of failures in the actual database. In addition to the attributes already specified in a reservation transaction, each failed transaction registers other internal attributes such as the date and time of occurrence and the travel agency sending the reservation. Each failed transaction has about 20 attributes or variables, and each of these attributes takes on a finite number of distinct values. For example, the airline attribute can have as value any of the airlines Travelport currently ser-

- *Zhicheng Liu is with the School of Interactive Computing and the GVU Center at the Georgia Institute of Technology, E-mail: zliu6@gatech.edu.*
- *John Stasko is with the School of Interactive Computing and the GVU Center at the Georgia Institute of Technology, E-mail: stasko@cc.gatech.edu.*
- *Timothy Sullivan is with Travelport Corp., E-mail: tim.sullivan@travelport.com.*

Manuscript received 31 March 2009; accepted 27 July 2009; posted online 11 October 2009; mailed on 5 October 2009.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org .

vices.

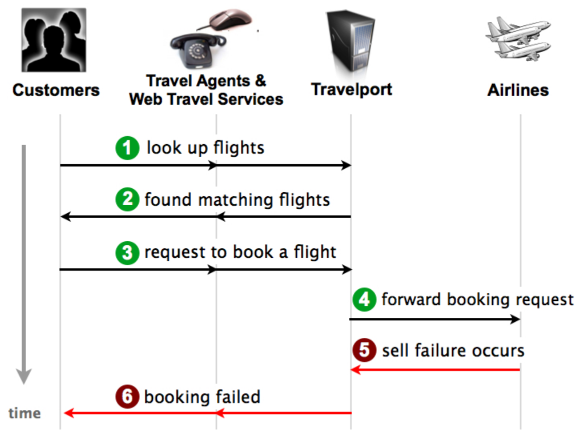


Fig. 1. Illustration of a Sell Failure Occurrence

2.2 Identifying Abnormality

It is an unrealistic goal to eliminate failed transactions completely. A low percentage of failures is normal and expected simply because of inventory volatility: flight availability information frequently changes as a result of updates from airlines or purchases made by other agents. Instead, Travelport's goal is to be aware of any abnormality that results in an unusual failure situation, identify the cause of this abnormality, and fix it. However, there are no precise threshold metrics to distinguish between "normal" and "abnormal" failure situations. Two high level statistics, number of failed transactions and failure rate (percentage of the number of failed transactions out of the total number of reservation transactions), are relevant indicators, yet simply looking at these two numbers is not sufficient. It is expected that reservation transaction volumes will be high on certain days (around Thanksgiving for example), and hence result in higher number of failed transactions. On the other hand, due to the huge number of reservation transactions, a small difference in failure rate could mean tens of thousands of failed transactions, and it may not be clear whether this discrepancy is due to internal factors that can be fixed or simple fluctuations of failure rate that are beyond the company's control.

To conduct more in-depth analysis on the distribution and structure of failures, company analysts currently use a web-based interface to issue database queries that specify parameters of failed transactions they want to examine, and they download the results as Excel spreadsheets for further analysis. These tools, however, are inadequate for the problem and the task remains difficult. First of all, without a clear idea of the up-to-the-minute situation on failed transactions, analysts often have to rely on their intuition or past experience and start probing with a few major airlines or travel agents that may contribute high percentages of failed transactions. Being associated with a high percentage of failures is not always indicative of abnormality though. Major airlines or travel agents are expected to contribute more failures simply because of their size. The abnormality can instead be associated with any combination of any number of the attribute values. For example, the problem could arise from transactions initiated by travel agent TA49 with San Francisco as the departure point, taking first class seats, and operated by Airline ABC. The analysts simply cannot generate all possible combinations and test them out one-by-one by issuing database queries. Second, abnormality often represents deviation from long term average performance and hence must be determined based on historical trending analyses, which are not well supported by current tools. Furthermore, it is necessary to retrieve and compare results across multiple levels of detail, and the need for repetitive database query entries adds to the difficulty of the problem. As a result, analysts often are only able to identify potential problems several days after the abnormality has happened. Sometimes the problem even goes undiscovered and remains hidden in the large accumulation of transaction data.

3 User-Centered Design

We adopted a user-centered design methodology and conducted formative evaluation with the technology specialists from Travelport throughout the design process. We continuously gathered feedback to modify and improve the design. In the meantime, our understanding of the data set and the problem domain deepened significantly as we moved from designing and implementing prototypes to deploying and incorporating the system into real work settings. Figure 2 shows the high level stages and timeline of this iterative process. We started with meetings and discussions to gain an initial understanding of the domain and the problems that data analysts were facing. After a period of brainstorming for data representation and interaction ideas, we developed a high fidelity prototype that visualizes sample data from spreadsheets. The sample data set used in prototyping was failed transactions associated with only four airlines in one week, but when we moved towards deployment it became clear that the spreadsheet-based data representation could not scale well to support smooth analyses at multiple levels of detail. We decided that the only way to handle the scalability issue was to directly query the failure database and dynamically create visualizations in accordance with user interaction. As we will discuss later, querying the database on the fly imposed constraints on the system that had not surfaced in earlier design iterations. We revised the system accordingly to balance the trade-off between a degree of high interactivity and the need for readily available visual information to guide user exploration.

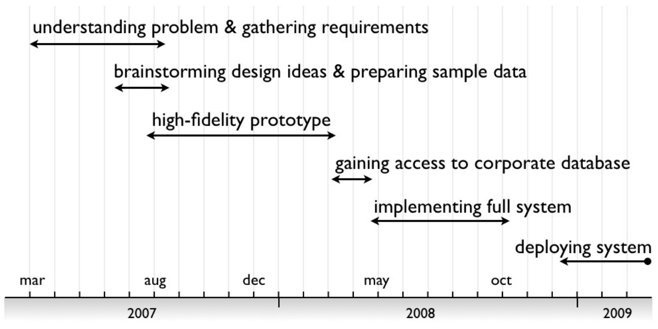


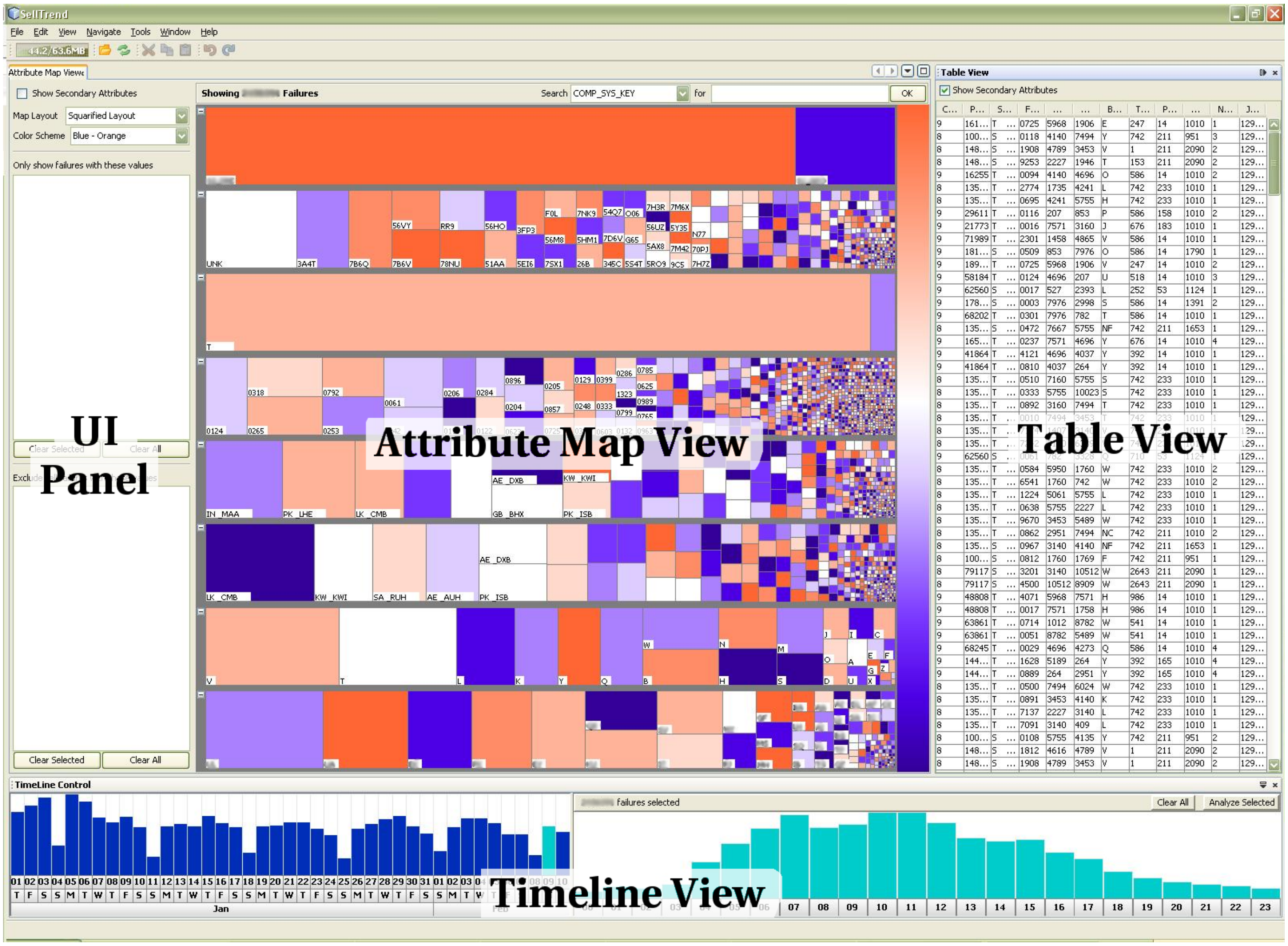
Fig. 2. The iterative design timeline

4 Supporting Visual Analysis Activities : Design Requirements

Based on our understanding of the data and the problem to be addressed, we identified the following inter-related tasks that the interactive visualization system needs to support.

First, it is important that the analysts have a *high-level awareness* of how reservation transactions are being handled by Travelport at any given point in time. For example, are many failed transactions occurring today? Is the failure rate at this hour higher compared to yesterday or just two hours ago? Which attribute values are contributing most to the failures? Answering these questions should require minimal efforts that are more glance-and-tell in nature, and a dashboard-like interface seems appropriate. The system thus needs to provide real-time visual summaries of failed transactions in meaningful temporal granularities, such as a day or an hour, and display such information side-by-side to support easy comparison and extrema identification.

Moreover, given any selected time intervals and the failed transactions that fall into those windows, an overview of the various attribute values is needed to provide visually salient information for general awareness. The analysts should be able to identify, for each attribute, the top values that account for the majorities of the failed transactions. For example, the analysts may want to look at the airline attribute in the past 3 hours, and determine quickly which airlines, if any, have



☐ Show Secondary Attributes

Map Layout: Squarified Layout

Color Scheme: Blue - Orange

Only show failures with these values

Exclude

Showing [redacted] **Failures**   Search COMP_SYS_KEY for

Table View

☑ Show Secondary Attributes

C…	P…	S…	F…	…	…	B…	T…	P…	…	N…	J…
9	161…	T …	0725	5968	1906	E	247	14	1010	1	129…
8	100…	S …	0118	4140	7494	Y	742	211	951	3	129…
8	148…	S …	1908	4789	3453	V	1	211	2090	2	129…
8	148…	S …	9253	2227	1946	T	153	211	2090	2	129…
9	16255	T …	0094	4140	4696	O	586	14	1010	2	129…
8	135…	T …	2774	1735	4241	L	742	233	1010	1	129…
8	135…	T …	0695	4241	5755	H	742	233	1010	1	129…
9	29611	T …	0116	207	853	P	586	158	1010	2	129…
9	21773	T …	0016	7571	3160	J	676	183	1010	1	129…
9	71989	T …	2301	1458	4865	V	586	14	1010	1	129…
9	181…	S …	0509	853	7976	O	586	14	1790	1	129…
9	189…	T …	0725	5968	1906	V	247	14	1010	2	129…
9	58184	T …	0124	4696	207	U	518	14	1010	3	129…
9	62560	S …	0017	527	2393	L	252	53	1124	1	129…
9	178…	S …	0003	7976	2998	S	586	14	1391	2	129…
9	68202	T …	0301	7976	782	T	586	14	1010	1	129…
8	135…	S …	0472	7667	5755	NF	742	211	1653	1	129…
9	165…	T …	0237	7571	4696	Y	676	14	1010	4	129…
9	41864	T …	4121	4696	4037	Y	392	14	1010	1	129…
9	41864	T …	0810	4037	264	Y	392	14	1010	1	129…
8	135…	T …	0510	7160	5755	S	742	233	1010	1	129…
8	135…	T …	0333	5755	10023	S	742	233	1010	1	129…
8	135…	T …	0892	3160	7494	T	742	233	1010	1	129…
8	135…	T …	0010	7494	3453	T	742	233	1010	1	129…
8	135…	T …	[illegible]	1407	3140	V	[illegible]	[illegible]	1010	1	129…
8	135…	T …	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	129…
9	62560	S …	0061	782	3328	Q	710	53	1124	1	129…
8	135…	T …	0584	5950	1760	W	742	233	1010	2	129…
8	135…	T …	6541	1760	742	W	742	233	1010	2	129…
8	135…	T …	1224	5061	5755	L	742	233	1010	1	129…
8	135…	T …	0638	5755	2227	L	742	233	1010	1	129…
8	135…	T …	9670	3453	5489	W	742	233	1010	1	129…
8	135…	T …	0862	2951	7494	NC	742	211	1010	2	129…
8	135…	S …	0967	3140	4140	NF	742	211	1653	1	129…
8	100…	S …	0812	1760	1769	F	742	211	951	1	129…
8	79117	S …	3201	3140	10512	W	2643	211	2090	1	129…
8	79117	S …	4500	10512	8909	W	2643	211	2090	1	129…
9	48808	T …	4071	5968	7571	H	986	14	1010	1	129…
9	48808	T …	0017	7571	1758	H	986	14	1010	1	129…
9	63861	T …	0714	1012	8782	W	541	14	1010	1	129…
9	63861	T …	0051	8782	5489	W	541	14	1010	1	129…
9	68245	T …	0029	4696	4273	Q	586	14	1010	4	129…
9	144…	T …	1628	5189	264	Y	392	165	1010	4	129…
9	144…	T …	0889	264	2951	Y	392	165	1010	4	129…
8	135…	T …	0500	7494	6024	W	742	233	1010	1	129…
8	135…	T …	0891	3453	4140	K	742	233	1010	1	129…
8	135…	T …	7137	2227	3140	L	742	233	1010	1	129…
8	135…	T …	7091	3140	409	L	742	233	1010	1	129…
8	100…	S …	0108	5755	4135	Y	742	211	951	2	129…
8	148…	S …	1812	4616	4789	V	1	211	2090	2	129…
8	148…	S …	1908	4789	3453	V	1	211	2090	2	129…

[redacted] failures selected

Fig. 3. The SellTrend System Interface. Specific items and values have been blurred in the image when they disclose proprietary company information.

been key contributors to the failed transactions. In addition, the analysts may want to know if these airlines have been contributing more failures than usual. Such information should be readily available and easily identified.

Once an interesting finding is made, a more *in-depth interactive analysis* is needed. Suppose there has been an unusually high number of failed transactions today, and the analysts would like to know what constitutes this problem. Often a combination of different values can be identified as the major cause of failure spikes. Since there are nearly 20 attributes and each attribute can take hundreds to thousands of categorical values, it is non-trivial to identify the combination of values that actually lead to the abnormality in failure volume. Interactive visual exploration by drilling down and focusing on a small number of failed transactions can play an important role in addressing these issues. Experienced analysts should be able to identify the problem with the help of meaningful visual cues and intuitive operations that focus on or exclude any failed transactions. With these filtering techniques, the analysts can isolate and concentrate on factors that truly matter.

5 Designing SellTrend: Principles and Specifics

SellTrend's user interface (UI) provides multiple views for exploring the data (Figure 3). At the bottom, the Timeline View shows bar charts of failed transactions by day and hour. An analyst examines and clicks to select a time period to begin investigation. All the failures from the selected period then are displayed in the Table View which provides a spreadsheet-like representation of individual failures. The Attribute Map View displays the same failures in a different manner. Each of the rows corresponds to a transaction attribute such as airline, purchasing agent, departure date, fare class, and failure code. Within an attribute's row, each small rectangle represents a different categorical value of that attribute, with the size of the rectangle corresponding to the number of failures with that value. Thus, a particular failure transaction will be represented within each of the attribute rows, where it will be aggregated with all other failures containing that value during the selected time period. The largest rectangles to the left signify the attribute values with the most failures during the selected period. Hovering over any of these rectangles will bring up a tooltip that shows the attribute value it represents, and the number of failed transactions associated with that value. Analysts can narrow down to failed transactions based on a specific set of attribute values of interest during the visual exploration process. SellTrend provides two basic filtering operations: focus and exclusion. A focus operation shows only failed transactions with a particular value for an attribute. An exclusion operation discards failed transactions with the specified value instead.

In this section, we describe more details of SellTrend and discuss our rationale in making these design decisions. To make our discussion coherent and organized, we group design choices by general design principles listed below.

5.1 Event-based Coordination between the Views

Our earlier analysis in Section 4 showed that it is beneficial to focus on different aspects of the data and to correlate and interweave them in a meaningful visual exploration. A popular solution that has been used in a number of systems [7, 24, 28] is to show these different aspects of the data in coordinated views.

Two major events trigger coordinated changes in SellTrend views: change of the analysis time window and change of filters. Users select specific time windows in the Timeline View and the other two views show different visual information about failures in the chosen periods accordingly. Filters can be added or removed during any stage in the visual analysis, and changes in filter cause all three views to update so that only failed transactions with attribute values as specified by the filters are shown.

5.2 Simple Visual Representation Consistent with Users' Mental Model

We characterize the failed transactions being examined here as temporal multi-variate event sequences. In a failed transaction, there are three time-related attributes: date of failure occurrence, time of failure occurrence and date of flight in the reservation request. Based on our analysis of data characteristics and user tasks, we treat these temporal attributes differently. Date and time of failure occurrence are two special dimensions to be visualized separately in the Timeline View, while date of flight is akin to other transaction attributes such as departing city and flight class.

In the Timeline View, we identify two levels of temporal granularity: daily and hourly. This is derived from the conceptual model that the analysts usually hold. The analysts are interested in the accumulated failed transactions within an hour or within a day. Hence the view provides flexible aggregation at different levels of granularity. We use a simple bar chart to represent the distribution of failed transactions across hours and days. Our initial design only addressed a single granularity level: the bar chart represented the hourly number of failed transactions in the entire time frame implicit in the data set, and users could zoom and focus on a selected period of time using a range slider. However, the analysts had no direct visual cues on the number of failed transactions aggregated by days and comparison across days was not intuitive. As a result, we split the view into two panels, the day panel and the hour panel. The analysts can select one bar in the day panel to see the hourly distributions in the hour panel. If multiple bars are selected, the hourly distributions are displayed as stacked histograms [10] in the hour panel.

Currently the Timeline View only shows the absolute number of failed transactions in temporal units. As we have discussed earlier, another useful metric is failure rate. Due to the lack of needed data, however, we are not able to obtain and visualize this information. To calculate failure rate requires information on all reservation requests. Unfortunately Travelport only stores detailed information on failed transactions, and to record successful reservation transactions means even greater requirement on company's data storage capabilities. It will nevertheless be very useful if we have access to and visualize this information.

Based on the specifications in Section 4, two important pieces of information associated with each attribute value must be shown explicitly. First, how does each attribute value contribute to the failed transactions in a given time window? Second, how does each attribute value perform in comparison to its historical average? The need to show these two pieces of information implies that we should use two distinct forms of visual cues, and color and shape are two perceptually salient choices. In the Attribute Map View, attributes are displayed in rows. Thus, one row corresponds to airline carrier, one to fare class, one to departure city and so on. Within each row, a rectangle represents each value that attribute takes on. In the airline row for instance, the different rectangles correspond to the different airlines. The size of the rectangle is directly proportional to the number of failed transactions having that value for that attribute out of the total number of failures in the selected time period.

Given a wide range of visual representation techniques available for multi-variate categorical data, we decided that a modified version of the Treemap [14] would be the most appropriate among the alternatives, even though the Treemap technique was originally developed for hierarchical data instead of categorical data. First it is important to separate the attributes visually so that they do not overlap. It is also desirable that there is enough room to depict value specific information within each attribute. Hence a space-filling approach seemed appropriate. Second, it is usual for an attribute of the failed transaction to be associated with a value set consisting of hundreds to tens of thousands of elements. As a result, approaches such as Parallel Sets [4] are not satisfactory because values with low frequencies tend to be represented as thin-slices that are not visually perceptible. This problem also happened when initially we used a simple slice-and-dice Treemap algorithm to display all the values within an attribute. Values associated with small ratios thus were in the form of thin slices, and the analysts pointed out this was not desirable as they would also want to see values that account for small ratios of failures but have done worse compared to their performance in historical levels. We thus used the squarified Treemap layout [6] as the default and added the strip layout [2] as well, but we have kept the slide-and-dice layout as an option in case there are situations where it is preferred. One limitation of using the Treemap technique is that when the area representing a value becomes too small, it is difficult to label that value. However this is a problem applicable to other alternatives as well. We show the label and additional detailed information about each value using a tooltip in the conventional way.

The color of the rectangle indicates the performance of that value in the selected time period compared to its historical value over time. Figure 3 shows a blue-orange scheme, where blue shows that the attribute value accounts for less failures on average in the chosen period than it does historically, and orange indicates worse performance than usual. Given an attribute v_i with a value set S_i which can contain elements $\{s_1, s_2, ... s_m\}$, the numerical parameter α to be mapped to a color for each value s_j in S_i at any instant of the visual exploration is calculated as

$$\alpha = \frac{N_f(\{[t,t']\}, v_i = s_j)}{N_f(\{[t,t']\})} / \frac{N_f([t_{min}, t_{max}], v_i = s_j)}{N_f([t_{min}, t_{max}])} \tag{1}$$

with parameters defined as follows:

- $\{[t,t']\}$: a set of time intervals defined by the analysts as the visual analysis time frame
- $N_f([t_{min}, t_{max}])$: total number of failed transactions in the entire time frame implicitly defined in the data set, with any user-defined filters applied
- $N_f(\{[t,t']\})$: total number of failed transactions in a set of time intervals, with any user-defined filters applied
- $N_f(\{[t,t']\}, v_i = s_j)$: total number of failed transactions in a set of time intervals with attribute v_i having the value s_j, with any user-defined filters applied

This number is then normalized and mapped to a predefined color scheme using a logarithmic function. We have chosen diverging color schemes [5] such as the blue-orange scheme because the data to be shown is double-ended and has a crucial midpoint in the data range where the performance of an attribute value has remained stable compared to its historical average. Given an array of colors ordered along the scheme with size n, and the color at index $n/2$ representing the meaningful midpoint in the data, the index of color i to be chosen for α is calculated as:

$$i = \begin{cases} n/2 * (1 + log(\alpha)/log(\alpha_{max})), & \alpha > 1; \\ n/2, & \alpha = 1; \\ n/2 * (1 - log(\alpha)/log(\alpha_{min})), & \alpha < 1; \end{cases} \tag{2}$$

Company analysts pointed out a problem with this color coding approach. We used a historical comparison analysis to decide the colors assigned to each value. However, the analysts were concerned that if a value has been contributing to high failure rates in the past and if it is improving in the current time frame, it will be colored as performing better compared to the historical records. However, it is possible that while the performance has improved, it is still above the normal rate that is deemed acceptable. Hence inappropriate coloring can mislead interpretation and hinder the abnormality identification. To address this concern, we provide two mechanisms of coloring with the first being the historical trend analysis. In the second, we allow the analysts to define a threshold of acceptable failure rates for selected attribute values, and the coloring then will be based on this threshold. The analysts have the option to switch between these two coloring mechanisms so that they can have a better understanding of the data.

Not all attributes are of equal importance: the analysts are more interested in airline ID than number of passengers, for example. In particular, nine attributes are of primary importance to the analysts. Values associated with these attributes are most likely to cause failure spikes. The other attributes are considered secondary and less crucial. On start-up, SellTrend shows the primary attributes only, the remaining attributes can be added via a simple check box in the UI panel.

The Table View is a standard spreadsheet that shows individual failed transactions in rows and attributes in the columns. Secondary attributes are also hidden on start-up and they can be turned on by clicking the check box in the toolbar. Analysts can re-arrange the ordering of columns by dragging and dropping column headers to the desired location.

The UI panel provides a set of controls that enables analysts to customize the Attribute Map View according to their own preferences. The analysts can choose from the three Treemap layout techniques described earlier (the strip, the squarified and the slice-and-dice Treemap). In addition to the blue-orange color scheme, a red-green scheme is available.

5.3 Direct Manipulation Facilitates Cognitive Coupling

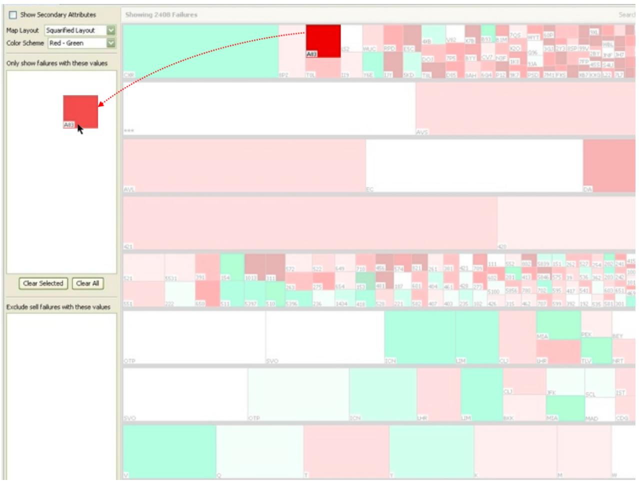
We recognize the power of direct manipulation interfaces in narrowing semantic and articulatory distances [12]. In many cases, users can actively construct and manipulate visual representations to couple with their own mental models and representations. Direct manipulation thus offloads processing of internally represented information to processing of perceptual information, and physical manipulation of visual representations itself constitutes part of the analysis [18]. In SellTrend, each rectangle representing an attribute value in the Attribute Map View is an interactive object that responds to user action such as clicking and dragging. Users perform focus or exclusion operations by clicking on a rectangle and dragging and dropping it to the focus or exclude regions in the UI panel (Figure 4). The Attribute Map View also includes a search function so that the analysts can specify the attribute value of focus. Searching for a value is equivalent to focusing on that value, and searching for multiple values within the same attribute is also supported.

The drill-down process is often exploratory in nature, hence the analysts would want to be able to return to any previous state that they have encountered and perhaps start a new analysis thread down a different path from that state. SellTrend supports returning to previous states by allowing analysts to choose specific filters to remove or to simply clear all filters to return to the overview.

When an analyst focuses on an attribute value, it appears in the focus region and also fills up the entire row in the Attribute Map View. This is a redundant encoding of information and wastes screen real estate. We provide a small control button at the beginning of each row in the Attribute Map View that allows the analysts to collapse or expand each attribute row (Figure 5). When an attribute is collapsed, only the attribute name is drawn. In expanded attribute rows, the rectangles are labeled with the represented attribute values, thus users will know which attribute the row corresponds to. In addition, the attribute name is shown in the tooltip when users interact with individual rectangles.

5.4 High Interactivity and Prompt Feedback

In a dynamic visual exploration process, timely feedback on user action is important. We achieved high interactivity in the prototype version of SellTrend by storing all the failed transactions and their attributes in memory. This clearly could not scale when we deployed the system and connected it to the company database. Instead the system renders the visualization based on dynamic database queries translated from user-initiated interaction with the interface. We have kept the data-fetching time below 15 seconds in most circumstances and inform the analysts of the progress with a modal progress bar.



□ Show Secondary Attributes

Map Layout: Squarified Layout

Color Scheme: Red - Green

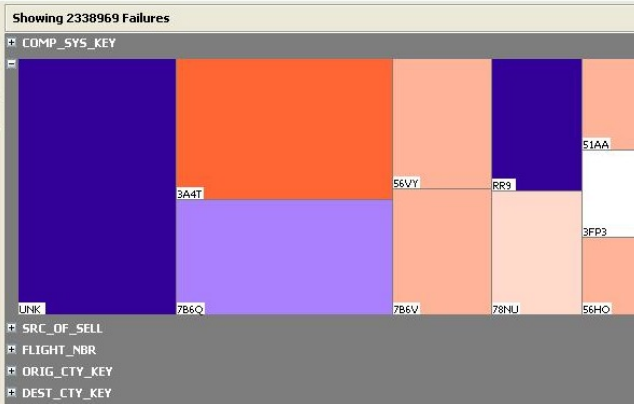
Only show failures with these values

Clear Selected | Clear All

Exclude sell failures with these values

Showing 2408 Failures

Fig. 4. Focusing on an attribute value by dragging and dropping it to the focus region



Showing 2338969 Failures

Fig. 5. Attributes COMP_SYS_KEY, SRC_OF_SELL, FLIGHT_NBR, ORIG_CTY_KEY and DEST_CTY_KEY are collapsed

Performing historical trend computation as described in Equation 1 turned out to be a problem however. It takes much longer to run database queries that do not narrow down to a specific date range (e.g., as in $N_f([t_{min}, t_{max}])$). The latency is unacceptable in an interactive exploration.

An alternative to dynamic querying is to precompute these parameters before-hand, but this causes a different problem. Given that there are about 20 attributes with associated value sets that range from a few hundred to tens of thousands of elements in size, the required precomputation that considers all possible combinations of filters easily becomes computationally intractable. For instance, consider two primary attributes, airline ID and agency ID. Assuming that there are only 20 airlines and 50 agencies, the total number of possible filter combinations is already huge:

$$\sum_{m=1}^{20} \binom{20}{m} \sum_{n=1}^{50} \binom{50}{n} = 1.2 \times 10^{21}$$

We believe that this problem cannot be solved completely. Instead,

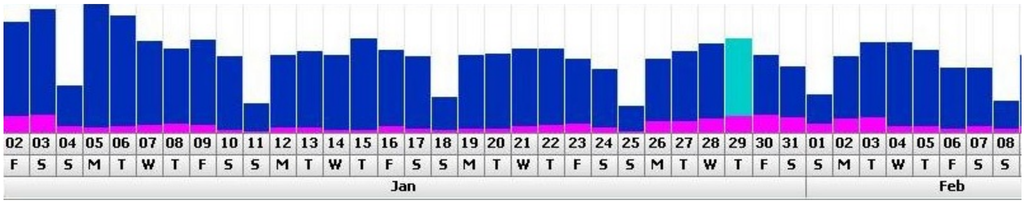


Fig. 6. Failed transactions by day with filters applied, represented as a magenta bar chart and overlaid on top of the Timeline View

we have developed two methods of doing trending analysis. First, whenever a new filter is added, SellTrend retrieves information about the number of failed transactions by days or by hours with the updated filters, visualizes this information in magenta and overlays the bar chart representation on top of the Timeline View (Figure 6). The bar chart representation provides more granular information on the performance of user selected filters and may enable easier identification of problematic filter combinations. Second, although precomputing combinations of all attribute values is not possible, we can at least compute color parameters for each attribute value when the overview is shown. When the analysts start drilling down, visual representations shown in Figure 6 can provide cues that guide further exploration. In addition, in actual work practices the analysts may never need to examine so many filter combinations, and it may still be possible to pre-identify a small subset of interesting attribute values and filter combinations. We are working with the analysts to see if this approach is viable.

Due to the high volumes of failed transactions being visualized, it is also problematic to display all the failures being examined at an instant in the Table View. Instead, in coordination with user interactions in the Attribute Map or Timeline View, only the top 500 failed transactions are retrieved from the database and displayed. More can be fetched if the analysts desire more details.

6 Implementation

SellTrend is built on top of the NetBeans Rich Client Platform, a modular plug-in based architecture. The visualizations use the Java Swing library. We use the Treemap layout algorithm library by Bederson and Wattenberg [3] in implementing the Attribute Map View. The library we used to process Excel files in the prototyping phase is the Java Excel API developed by Andy Khan [16].

7 Use-Case Scenario

In this section we provide a scenario using a small example data file for illustration purposes. This scenario is also illustrated in the accompanying video.

The sample data contains failed transactions associated with four airlines over a period of seven days. We can immediately identify in the Timeline View that a failure spike occurred on Monday, August 27th (Figure 7(a)). After selecting failed transactions over all 24 hours in that day, we focus on the Attribute Map View. We can see that, for example, travel agent DYS has contributed the most failed transactions in comparison to the other agents, but the region representing it is a light shade of blue, indicating that it has contributed less failures in the selected time window than normal. Instead, we see that airline A80 is contributing a large volume of failed transactions, worse than its historical average. Hence, it may worth investigating further (Figure 7(b)).

We focus on airline A80 so only its failed transactions are shown. Examining the other attributes, we note more than one to explore further, for example, class Z and R, flight number 4360 and error code 78 (Figure 7(c)). From our previous experience, we decide to drill down to flight number 4360 first, because flight number is typically a key attribute. It then becomes clear that airline A80 and flight number 4360 might be part of the cause of the failure spike on August 27th. Class Z and R together, and error code 78 correspond to 100% of failed transactions associated with A80 and Flight Number 4360 (Figure 7(d)).

This discovery is also corroborated by the overlaid information in the Timeline View, where we can see that August 27th stands out with a visible surge in failed transactions associated with these two attribute values (Figure 7(e)). The hour panel shows that the surge was happening around 3-4pm and 9pm on that day. The Attribute Map View further indicates that travel agent Z7F might be the right stakeholder to contact for resolution. With just a few clicks we have identified an interesting attribute value combination that likely constitutes an abnormality.

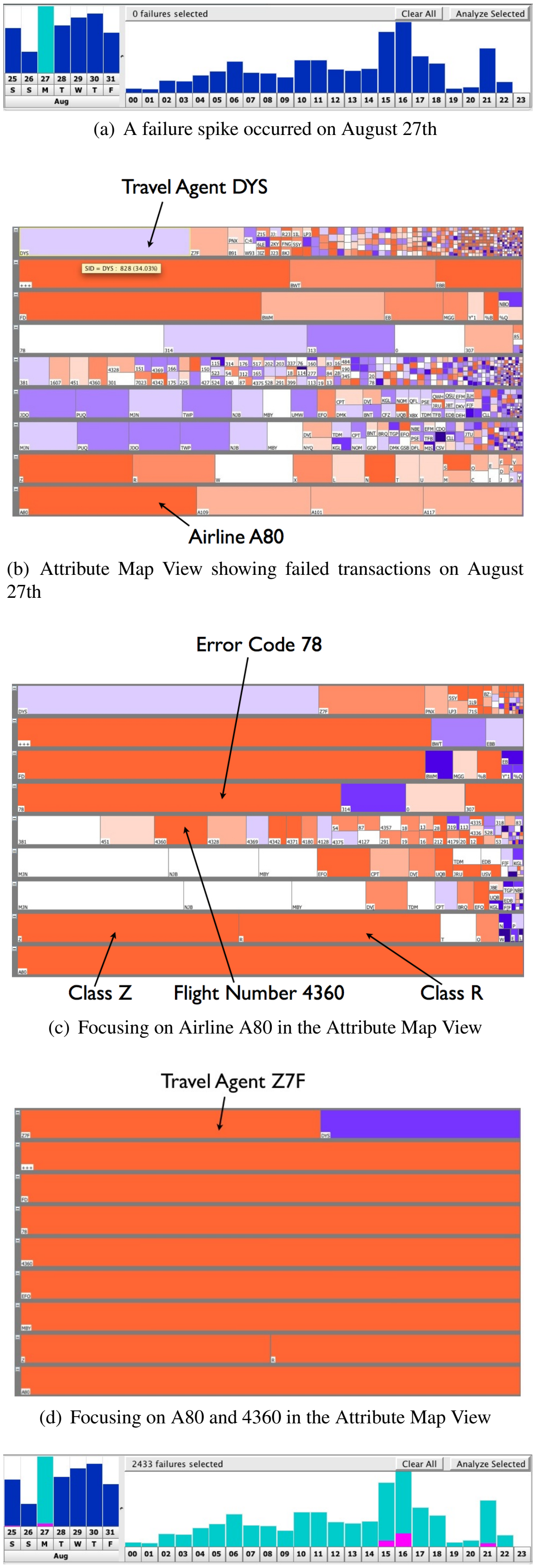
8 Related Work

Various techniques for representing and interacting with time series data are available. Aigner et al. outline a framework that addresses the visualization of time-oriented data [1]. A common approach is to represent time as a linear ordered axis [11, 19] and plot data cases by their timestamps. This linear representation helps identify overall trends and peaks in time. Considering the massive scale of the data and the nature of the tasks we are supporting, the ordering of individual failure data cases is not so important. Instead, we focus on the trends and extrema in terms of aggregation of failed transactions by time intervals.

Techniques for analyzing time series data are available for analyses at data item and aggregation levels. Systems such as the Map of the Market [27] perform historical trend analysis on individual data cases and use pre-defined color scales to show trending properties of data items. Analysis at an aggregated level often clusters and abstracts data cases exhibiting similarities [1]. In our problem, the focus is not on individual failures, but the temporal trend aggregated by attribute values.

Almost every attribute of a failed transaction is a categorical variable. For date of flight, although it is temporal in nature, the aggregation of failed transactions grouped by individual dates is more important than trends or cyclic patterns, thus it is treated as categorical too. Visualizing categorical data has been extensively explored in the past. The Parallel Sets technique [4] shows that by managing attributes independently and aggregating categorical values within an attribute by frequency, interactive visual analysis can reveal patterns within a complex data set. The FOCUS system [23], which later evolved into InfoZoom [22], displays relational databases as compact tables where attributes are displayed in rows, and the values are aggregated by frequency. Database queries are translated into simple user interactions based on the `focus+context` technique. EZChooser [29] uses similar representations with richer interaction techniques that are also found in systems such as the Attribute Explorer [21]. The Pixel Bar Chart [15] is a visualization technique for very large multi-attribute data that makes effective use of screen space. In these systems, however, the temporal aspect is not given emphasis and often is treated as a common variable just like the others.

Data searching and browsing by faceted metadata is also a related topic. Metadata about dataset items is grouped into multiple orthogonal categories called “facets”, which may be aligned with the categorical attributes in the failure data. Based on the principle that people can recognize information much more easily than they can recall it, systems usually start with an overview showing all the facet values that might be useful to organize the items. The Flamenco system [30] displays the number of dataset items associated with each of the facet values to guide the users through the searching and browsing process.



(a) A failure spike occurred on August 27th

(b) Attribute Map View showing failed transactions on August 27th

(c) Focusing on Airline A80 in the Attribute Map View

(d) Focusing on A80 and 4360 in the Attribute Map View

(e) A surge of the value combination in the Timeline View on August 27th

Fig. 7. A use-case scenario illustrating how an interesting attribute value combination is identified

In FacetMap [20], facets with wider applicability to the data items are given larger screen size. Users interact with the interface to build up incremental queries for searching and browsing. FacetZoom [9] allows smooth and continuous navigation across levels in hierarchical facets and makes effective use of limited screen space. Faceted browsing is an insightful idea that inspires our design, yet again in these systems, although time often appears as an attribute in a given data set, it is given no special treatment and is not distinguished from other variables. Furthermore, the task supported by a faceted interface is primarily browsing and searching for specific data items, thus most faceted systems divide their interfaces between a main viewing area and a secondary facet area. The relationships between the facet values are not the focus. The Travelport analysts have expressed clear disinterest in individual failed transaction details, so our design places an emphasis on the identification of the combination of attribute values as the causes of failure problems, and historical analysis on attribute values and combinations of attribute values is necessary.

While research projects exploring time-series data or categorical data are abundant, not many have combined visual analyses of these two types of data for specific problem domains. The TimeWheel [26] is an attempt to visualize time-oriented multivariate data using a multi-axes representation.The time axis is placed in the center of the display while a set of axes that represent attributes is circularly arranged around the time axis. In the three dimensional view the TimeWheel can be rotated to bring any attribute into focus. The axis-based representations of the attributes resemble the parallel coordinates technique [13], and do not necessarily portray categorical data well. FacetLens [17] is a continuation of the FacetMap project where time is incorporated as a "linear facet". Our design shares many ideas with it, but our work involves more in-depth trending analysis on facet values and the relationships between them.

In terms of case studies, the IDS Rainstorm [8] and MieLog systems [25] focus on time series log data that share similar characteristics with the Travelport data. IDS Rainstorm uses visualizations to help network system administrators analyze massive amounts of alarm data generated from intrusion detection systems. The dataset involved is time series alarms with four attributes. The main task supported by the system is the identification of individual alarms that are of high severity and time-based analysis is not crucial for this task. MieLog tries to support easier discovery of unusual log messages based on time trends and frequency. In contrast, the concept of "severity" or "abnormality" is not associated with an individual transaction in the failure data. We are not trying to find specific unusual transactions. Instead, our goal is to identify the causes of abnormality in aggregations of failed transactions. Furthermore, the failure data has many more attributes and all the attributes are categorical in nature.

9 User Feedback and Future Work

SellTrend has been deployed since late December 2008. Through our discussions with the analysts, we find that the greatest benefits SellTrend provides are time saved in foraging information on various attribute values and the easier comparison and exploration across different attribute values. As a result, the analysts can make sense of the data better and faster. As an analyst stated,

> *"The data visualization tool provides [us] the ability to analyze potentially millions of rows of data at a customer and/or supplier level in a single 'view'. By reducing this initial investigation time frame by greater than 90%, we can more quickly direct our resources to the underlying sell failure root cause. Addressing sell failure issues in a timelier manner improves our product quality, improves the customer experience, and provides a more efficient use of Travelport staff."*

While we are confident that SellTrend is a useful tool from the feedback we have gathered throughout the design process, more field work will be necessary to understand the implications of our design decisions, in paricular, how SellTrend is used and may be appropriated in the actual working context. Now that SellTrend has been deployed for several months, we would like to conduct longitudinal observations as well as in-depth interviews to investigate how SellTrend may play a role in shaping the analysts' sense-making strategies in a larger organizational context.

There are also alternative designs we would like to evaluate. In SellTrend, each filtering operation is equivalent to issuing a database query, and users rely on two visual cues, the size and color of the rectangles, to formulate and refine their queries. A consequence of this design decision is that often multiple steps of filtering and rewinding are necessary before the analysts discover an interesting relationship between certain attribute values. To enable the analysts to explore and perceive quickly the relationship between the attribute values, a "brushing" technique used in the Attribute Explorer [21] and EZChooser [29] can be useful. The analysts, for example, can mouse over a particular error code and each rectangle in the other attribute rows shows information about the distribution of the number of failures with this error code. This feature adds more visual information in the Attribute Map View, and it is not clear to us whether the analysts will be overwhelmed by the rich information.

10 Conclusion

We have presented SellTrend, a system designed for interactive visual analysis on time-oriented multivariate categorical data. We document our experience in this iterative design process and discuss general design principles. Having successfully deployed the system at Travelport Corp, we show how it can significantly enhance the analysts' awareness of the transaction performance in realtime as well as their ability to identify potential causes of problems. Our contributions include 1) the integration of techniques such as faceted browsing and trend analysis to shed light on a generalizable data set and problem domain, and 2) a case study of a real world challenge validated by formative evaluation and user-centered design. We speculate that this design may have broader applicability in other domains involving temporal transaction data made up of many categorical attributes.

Acknowledgments

This research is supported in part by a funding from Travelport Corp., the National Visualization and Analytics Center (NVAC$^{\text{TM}}$), a U.S. Department of Homeland Security Program, under the auspices of the Southeast Regional Visualization and Analytics Center and the National Science Foundation via Award IIS-0414667. We thank Venkat Hariharan for helping setting up access to the database and Howard Spellman, Harsh Bhatnagar and Arpita Tillani for helping configuring database queries. We also thank Edward Clarkson for the discussion on faceted browsing and the Information Interfaces Lab members for their comments on earlier drafts of this paper.

References

[1] W. Aigner, S. Miksch, W. Mller, H. Schumann, and C. Tominski. Visual methods for analyzing Time-Oriented data. *IEEE Transactions on Visualization and Computer Graphics*, pages 47–60, 2008.

[2] B. B. Bederson, B. Shneiderman, and M. Wattenberg. Ordered and quantum treemaps: Making effective use of 2D space to display hierarchies. *ACM Transactions on Graphics*, 21(4):833–854, 2002.

[3] B. B. Bederson and M. Wattenberg. Java 1.1 Library of Five Treemap Algorithms. http://www.cs.umd.edu/hcil/treemap-history/.

[4] F. Bendix, R. Kosara, and H. Hauser. Parallel sets: Visual analysis of categorical data. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 133–140, 2005.

[5] C. A. Brewer. Color use guidelines for data representation. In *Proceedings of the Section on Statistical Graphics, American Statistical Association*, pages 55–60, 1999.

[6] M. Bruls, K. Huizing, and J. J. van Wijk. Squarified treemaps. In *Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42, 2000.

[7] R. Chang, M. Ghoniem, R. Kosara, W. Ribarsky, J. Yang, E. Suma, C. Ziemkiewicz, D. Kern, and A. Sudjianto. WireVis: visualization of categorical, Time-Varying data from financial transactions. In *IEEE Symposium on Visual Analytics Science and Technology*, pages 155–162, 2007.

[8] G. Conti, K. Abdullah, J. Grizzard, J. Stasko, J. A. Copeland, M. Ahamad, H. L. Owen, and C. Lee. Countering security information overload through alert and packet visualization. *IEEE Computer Graphics and Applications*, pages 60–70, 2006.

[9] R. Dachselt, M. Frisch, and M. Weiland. FacetZoom: a continuous multiscale widget for navigating hierarchical metadata. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1353–1356, Florence, Italy, 2008. ACM.

[10] A. Dix and G. Ellis. Starting simple: adding value to static visualisation through simple interaction. In *Proceedings of the working conference on Advanced Visual Interfaces*, pages 124–134, 1998.

[11] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. ThemeRiver: visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, pages 9–20, 2002.

[12] E. L. Hutchins, J. D. Hollan, and D. A. Norman. Direct manipulation interfaces. *Human-Computer Interaction*, 1(4):311–338, 1985.

[13] A. Inselberg. Multidimensional detective. In *IEEE Symposium on Information Visualization*, pages 100–107, 1997.

[14] B. Johnson and B. Shneiderman. Tree-Maps: a space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd conference on Visualization*, pages 284–291, 1991.

[15] D. A. Keim, M. C. Hao, U. Dayal, and M. Hsu. Pixel bar charts: a visualization technique for very large multi-attribute data sets. *Information Visualization*, 1(1):20–34, 2002.

[16] A. Khan. Java Excel API. http://www.andykhan.com/jexcelapi/index.html, Oct. 2008.

[17] B. Lee, G. Smith, G. Robertson, M. Czerwinski, and D. S. Tan. FacetLens: exposing trends and relationships to support sensemaking within faceted datasets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Apr. 2009.

[18] Z. Liu, N. J. Nersessian, and J. T. Stasko. Distributed cognition as a theoretical framework for information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1173–1180, Dec. 2008.

[19] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. LifeLines: visualizing personal histories. *Proceedings of the SIGCHI conference on Human factors in computing systems*, 1996.

[20] G. Smith, M. Czerwinski, B. Meyers, D. Robbins, G. Robertson, and D. S. Tan. FacetMap: a scalable search and browse visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12:797–804, 2006.

[21] R. Spence and L. Tweedie. The attribute explorer: information synthesis via exploration. *Interacting with Computers*, 11(2):137–146, 1998.

[22] M. Spenke and C. Beilken. InfoZoom: Analysing Formula One racing results with an interactive data mining and visualisation tool. In *Proceedings of 2nd International Conference on Data Mining*, pages 455–464, Cambridge University, UK, 2000.

[23] M. Spenke, C. Beilken, and T. Berlage. FOCUS: the interactive table for product comparison and selection. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 41–50, 1996.

[24] J. Stasko, C. Görg, Z. Liu, and K. Singhal. Jigsaw: Supporting investigative analysis through interactive visualization. *Information Visualization*, 7(2):118–132, 2008.

[25] T. Takada and H. Koike. MieLog: A Highly Interactive Visual Log Browser Using Information Visualization and Statistical Analysis. In *Proceedings of the 16th USENIX Conference on System Administration*, pages 133–144, Philadelphia, PA, 2002.

[26] C. Tominski, J. Abello, and H. Schumann. Axes-based visualizations with radial layouts. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1242–1247. ACM New York, NY, USA, 2004.

[27] M. Wattenberg. Visualizing the stock market. *Conference on Human Factors in Computing Systems*, pages 188–189, 1999.

[28] C. Weaver, D. Fyfe, A. Robinson, D. Holdsworth, D. Peuquet, and A. M. MacEachren. Visual exploration and analysis of historic hotel visits. *Information Visualization*, 6(1):89–103, 2007.

[29] K. Wittenburg, T. Lanning, M. Heinrichs, and M. Stanton. Parallel bargrams for consumer-based information exploration and choice. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, pages 51–60, 2001.

[30] K. P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 401–408, 2003.