Information Visualization (2001) 0000, 00–00

www.palgrave-journals.com/ivs

Pixel bar charts: a visualization technique for very large multi-attributes data sets†

Daniel A. Keim[1,3,4]
Ming C. Hao[1]
Umesh Dayal[1]
Meichun Hsu[1,2]

[1]Hewlett-Packard Research Labs, Palo Alto, California, U.S.A.; [2]CommerceOne, Pleasanton, California, U.S.A.; [3]AT&T Research Labs, Florham Park, NJ, U.S.A.; [4]University of Constance, Germany.

Correspondence:
Daniel A. Keim, AT&T Shannon Research Lab, 180 Park Avenue, P.O. Box 971, Florham Park, NJ 07932, U.S.A.
Tel: +1 973 360 8482; Fax: +1 973 360 8077;
E-mail: Keim@research.att.com

Received: 14 August 2001
Revised: 19 September 2001
Accepted: 19 September 2001

Abstract
Simple presentation graphics are intuitive and easy-to-use, but show only highly aggregated data presenting only a very small number of data values (as in the case of bar charts) and may have a high degree of overlap occluding a significant portion of the data values (as in the case of the x-y plots). In this article, we therefore propose a generalization of traditional bar charts and x-y-plots, which allows the visualization of large amounts of data. The basic idea is to use the pixels within the bars to present detailed information of the data records. Our so-called *pixel bar charts* retain the intuitiveness of traditional bar charts while allowing very large data sets to be visualized in an effective way. We show that, for an effective pixel placement, we have to solve a complex optimization problem. We then present an algorithm which efficiently solves the problem. Our application to a number of real-world e-commerce data sets shows the wide applicability and usefulness of our new idea, and a comparison to other well-known visualization techniques (parallel coordinates and spiral techniques) shows a number of clear advantages.

Keywords: Information visualization; multi-dimensional data visualization; visual data exploration and data mining; very large multi-attributes data sets

Introduction

Because of the fast technological progress, the amount of data that is stored in computers increases very quickly. Researchers from the University of Berkeley estimate that every year about 1 Exabyte of data is generated, of which a large portion is available in digital form. Today, computers typically record even simple transactions of everyday life, such as paying by credit card, using the telephone, and shopping in e-commerce stores. This data is collected because business people believe that it is a potential source of valuable information and may provide a competitive advantage.

Finding the valuable information hidden in the data, however, is a difficult task. Visual data exploration techniques are indispensable to solving this problem. In most data mining systems, however, only simple graphics, such as bar charts, pie charts, and x-y plots are used to support the data mining process. While simple graphics are intuitive and easy-to-use, they either:

- show highly aggregated data and actually present only a very small number of data values (as in the case of bar charts or pie charts); or
- have a high degree of overlap which may occlude a significant portion of the data values (as in the case of x-y plots)

The usefulness of bar charts is especially limited if the user is interested in relationships between different attributes such as product type, price,

number of orders, and quantities. The reason for this limitation is that multiple bar charts for different attributes do not support the discovery of interesting subsets, which is one of the main tasks in mining customer transaction data.

For an analysis of large volumes of e-commerce transactions,[1] the visualization of highly aggregated data is not sufficient. What is needed is to present an overview of the data but at the same time show detailed information for each data item.

In this article, we describe a new visualization technique called *pixel bar chart*. The basic idea of pixel bar charts is to use the intuitive and widely used presentation paradigm of bar charts, but also use the available screen space to present more detailed information. By coloring the pixels within the bars according to the values of the data records, very large amounts of data can be presented to the user. To make the display more meaningful, two parameters of the data records are used to impose an ordering on the pixels in the x- and y-directions. Pixel bar charts can be seen as a generalization of bar charts. They combine the general idea of xy plots and bar charts to allow an overlap-free, non-aggregated display of multi-attribute data.

Information visualization techniques

Over the last decade, a large number of information visualization techniques have been developed, and some of the techniques are closely related to pixel bar charts. Information visualization techniques can be classified according to three different criteria (see Figure 1).

The first criterion is the 'data type to be visualized' (compare the taxonomy of B. Shnelderman[2]), such as:

- one-dimensional – e.g., temporal data
- two-dimensional data – e.g., geographical maps
- multi-dimensional data – e.g., relational tables
- text and hypertext – e.g., news articles
- hierarchies and graphs – e.g., web-sites
- algorithms and software – e.g., software debugging

One-, two- or multi-dimensional data may have real, integer, ordinal (enumeration with ordering) or nominal (enumeration without ordering) data dimensions. Bar charts require a partitioning of the data into a small number of partitions, which are used to form the bars. Enumeration data types with a small number of items are most commonly used for this purpose.

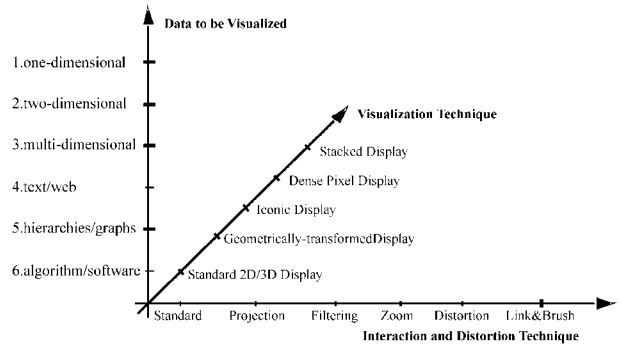

Figure 1 Information visualization classification

The second criterion according to which the techniques can be classified is the 'visualization technique' used:

- *standard 2D/3D displays* which use standard 2D or 3D visualization techniques such as x-y-plots or landscapes for visualizing the data;
- *geometrically-transformed displays* which use geometric transformations and projections to produce useful visualizations. Parallel coordinates,[3,4] projection pursuit,[5] and the various techniques for effectively visualizing graphs[6] belong to this category;
- *icon-based displays* which visualize each data item as an icon (e.g., stick figures or color icons[7,8]) and the data values as features of the icons;
- *dense pixel displays* which visualize each dimension value as a colored pixel and group the pixels belonging to each dimension into an adjacent area.[9–12] By arranging and coloring the pixels in an appropriate way, the resulting visualization provides detail information on local correlations, dependencies and hot spots;
- *stacked displays* which visualize the data partitioned in a hierarchical fashion (e.g.[13–15]). In case of multi-dimensional data, the data dimensions to be used for building the hierarchy have to be selected appropriately.

Pixel bar charts are a combination of standard techniques (bar charts and x-y plots) and pixel-oriented display techniques.

In addition to the visualization technique, for an effective data exploration it is necessary to use some 'interaction and distortion techniques'. The interaction techniques allow the user to directly interact with the visualization. Examples of interaction techniques include filtering, zooming, and linking.[16–18] Interaction techniques allow dynamic changes of the visualizations according to the exploration objectives, and they also make it possible to relate and combine multiple independent visualizations. Closely related examples are Value Bars[19] which are a special type of scrollbars mapping attribute values onto bars. Since each attribute value is mapped to a rectangular region, the number of data items which are visualized at one point of time is rather limited. Note that connecting multiple visualizations by interactive techniques, provides more information than considering the component visualizations independently. Interactive distortion techniques help in the data exploration process by providing means for focusing while preserving an overview of the data.[20] The basic idea of distortion techniques is to show portions of the data with a high level of detail while

others are shown with a lower level of detail. Well-known and widely used distortion techniques are spherical and hyperbolic distortions[21,22] or the TableLens approach for multi-attribute tabular data.[23]

From bar charts to pixel bar charts

Bar charts are a common method for visualizing large data volumes since they are intuitive and easy to understand. Figure 2 illustrates the use of regular bar charts to visualize customer e-commerce sales transaction data. In Figure 2a, the height of the bars represents the number of customers for 12 different product categories, and in Figure 2b, the width of the bars corresponds to the number of customers. Equal-width bar charts are widely used. The advantage of equal-height bar charts is that they make better use of the available screen space, but this comes at the disadvantage that the presented numbers are harder to compare.

In general, bar charts require a high degree of data aggregation and actually show only a rather small number of data values (only 12 values are shown in Figure 2). Therefore, for the exploration of large multidimensional data, they are of limited value and do not show important information such as:

- data distributions of multiple attributes
- local patterns, correlations, and trends
- detailed information, e.g., each customer's profile (age, income, location, etc.)

Basic idea of pixel bar charts

Pixel bar charts are derived from regular bar charts (see Figure 2a). The basic idea of a pixel bar chart is to present the data values directly instead of aggregating them into a few data values. The approach is to represent each data item (e.g. a customer) by a single pixel in the bar chart. The detailed information of one attribute of each data item is encoded into the pixel color and can be accessed and displayed as needed.

One important question is: How are the pixels arranged within each bar? Our idea is to use one or two attributes to separate the data into bars (dividing attributes) and then use two additional attributes to impose an ordering within the bars (see Figure 3 for the general idea). The pixel bar chart can therefore be seen as a combination of the traditional bar charts and the x-y diagrams. The result is a visualization in which one pixel corresponds to one customer. If the partitioning attribute is redundantly mapped to the colors of the pixels, we obtain the regular bar chart shown in Figure 4a (Figure 4b shows the equal-height-bar-chart which we will explain in the next section). Pixel bar charts, however, can be used to present large amounts of detailed information. The one-to-one correspondence between customers and pixels allows us to use the color of the pixels to represent additional attributes of the customer – for example, sales amount, number of visits, or sales quantity.

In Figure 4a, a pixel bar chart is used to visualize thousands of e-commerce sales transactions. Each pixel in the visualization represents one customer. The number of customers can be as large as the screen size (about 1.3 million). The pixel bar chart shown in Figure 4a uses product type as the dividing attribute and number of visits and dollar amount as the x- and y-ordering attributes. The color represents the dollar amount spent by the corresponding customer. High dollar amounts correspond to bright colors, low dollar amounts to dark colors.

Space-filling pixel bar charts

One problem of traditional bar charts is that a large portion of the screen space cannot be used due to the differing heights of the bars. With very large data sets, we would like to use more of the available screen space to visualize the data. One idea that increases the number of displayable data values is to use equal-height instead of equal-width bar charts. In Figure 2b, the regular bar chart of Figure 2a is shown as an equal-height bar chart. The area (width) of the bars corresponds to the attribute shown, namely the number of customers.

If we now apply our pixel bar chart idea to the resulting bar charts, we obtain space-filling pixel bar charts which use virtually all pixels of the screen to display customer data. In Figure 4b, we show an example of a space-filling pixel bar chart which uses the same dividing, ordering, and coloring attributes as the pixel bar chart in Figure 4a. Again, each customer is represented by one pixel.

Note that pixel bar charts generalize the idea of regular bar charts. If the partitioning and coloring attributes are identical, both types of pixel bar charts become scaled versions of their regular bar chart counterparts. The pixel bar chart can therefore be seen as a generalization of the regular bar charts but they contain significantly more information and allow a detailed analysis of large original data sets.

Multi-pixel bar charts

In many cases, the data to be analyzed consists of multiple attributes. With pixel bar charts we can visualize multiple attribute values using pixel bar charts that use different color mappings but the same dividing and ordering attributes. This means that the arrangement of data items within the corresponding bars of multi-pixel bar charts is the same, i.e., the colored pixels corresponding to the different attribute values of the same data item have a unique position within the bars. In Figure 5, we show an example of three pixel bar charts with product type as the partitioning attribute and number of visits and dollar amount as the x- and y-ordering attributes. The attributes which are mapped to color are dollar amount spent, number of visits, and sales quantity.

Note that the pixels in corresponding bars in multiple bar charts are related by their position, i.e., the same data record has the same relative position within each

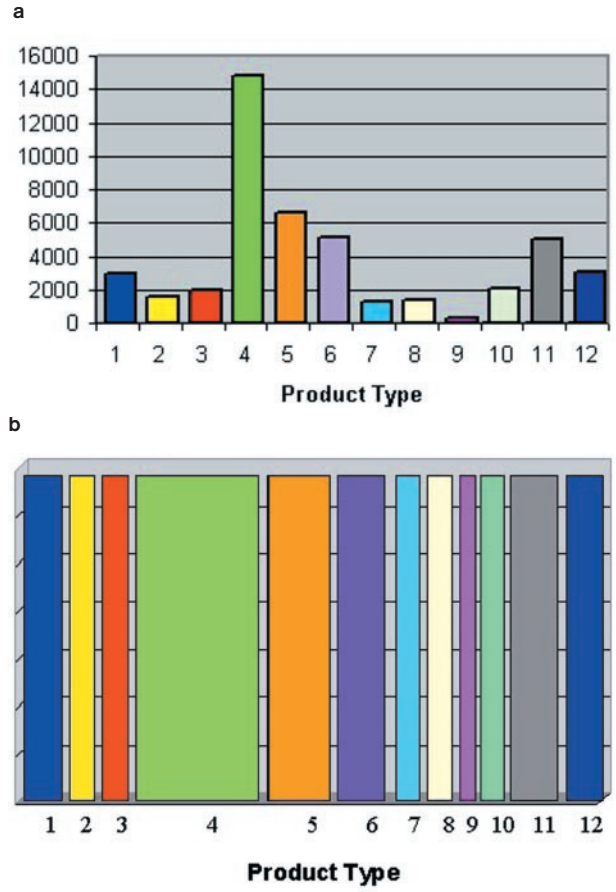

Figure 2 Regular bar charts. (a) Equal-width bar chart. (b) Equal-height bar chart

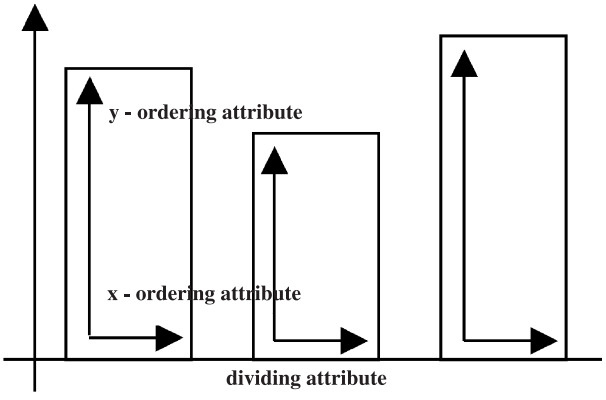

Figure 3 Pixel bar charts

of the corresponding bars. It is therefore possible to relate the different bar charts and detect correlations.

Formal definition of pixel bar charts

In this section we formally describe pixel bar charts and the problems that need to be solved in order to implement an effective pixel placement algorithm.

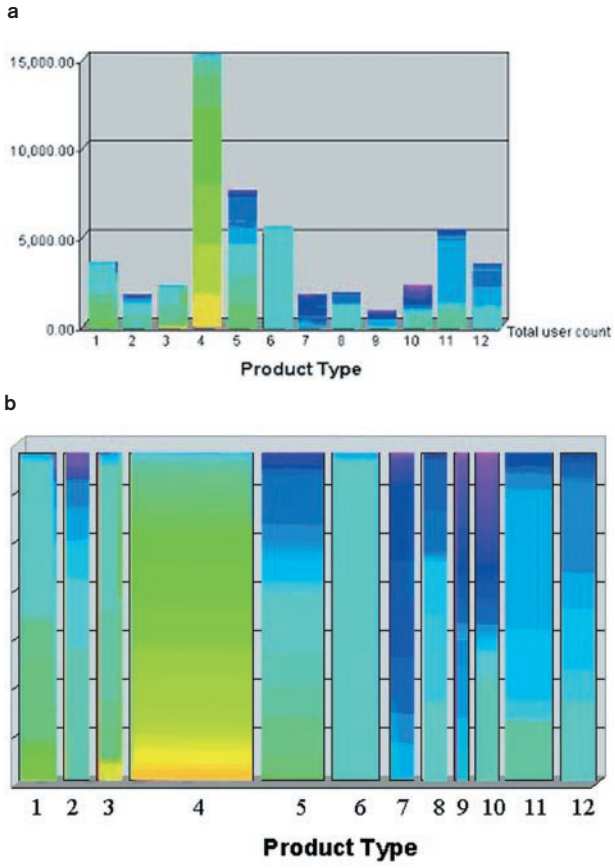

Figure 4 Pixel bar charts. (a) Equal-width pixel bar chart. (b) Equal-height pixel bar chart

Definition of pixel bar charts

For a general definition of pixel bar charts, we need to specify the:

- dividing attributes (for between-bar partitioning)
- ordering attributes (for within-bar ordering)
- coloring attributes (for pixel coloring)

In traditional bar charts there is one dividing attribute that partitions the data into disjoint groups corresponding to the bars. In space-filling bar charts, the bars correspond to a partitioning of the screen according to the horizontal axis (x) (see Figure 6).

We may generalize the definition of space-filling pixel bar charts by allowing more than one dividing attribute, i.e. one for the horizontal axis (D_x) and the one for the vertical axis (D_y) (see Figure 7).

Next, we need to specify an attribute for ordering the pixels within each pixel bar. Again, we can do the ordering according to the x- and the y-axis, i.e., along the horizontal (O_x) and vertical (O_y) axes inside each bar (see Figure 8).

Finally, we need to specify an attribute for coloring the pixels. Note that in multi-pixel bar charts

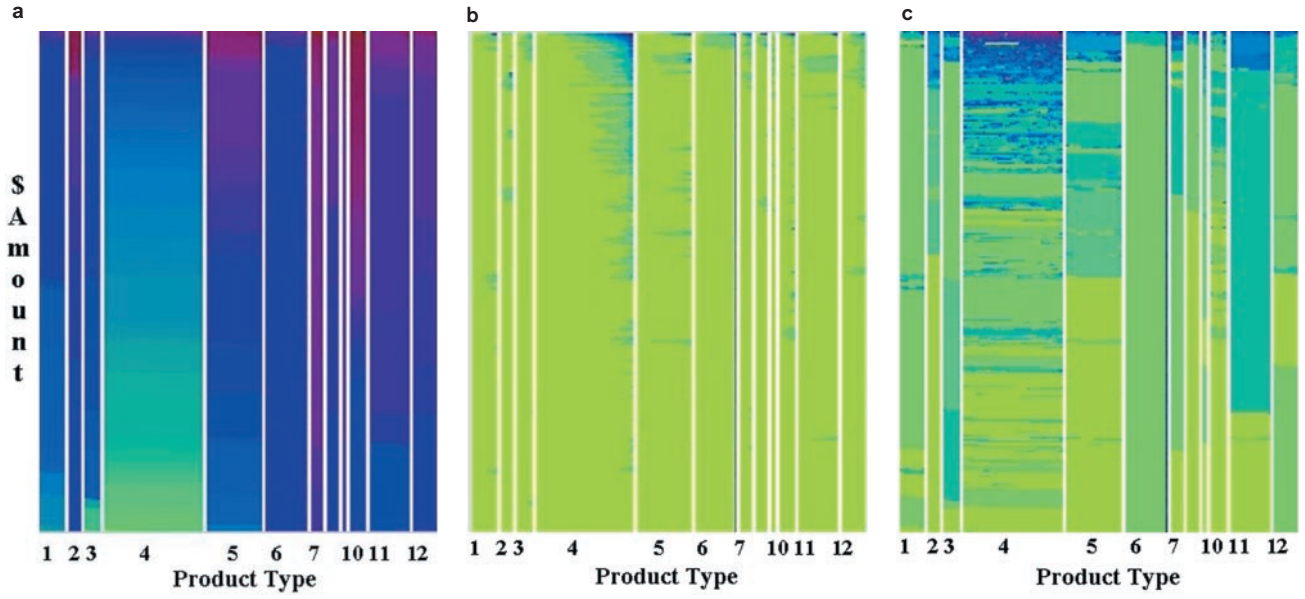

Figure 5 Multi-pixel bar charts. (a) Color=dollar amount. (b) Color=no. of visits. (c) Color=quantity

we may assign different attributes to colors in different bar charts, which enables the user to relate the different coloring attributes and detect partial relationships among them. Note that the dividing and ordering attributes have to be the same in order to do that.

Let $DB=\{d_1,...,d_n\}$ be the data base of n data records, each consisting of k attribute values $d_i = \{a_1^i, ..., a_k^i\}, a_l^i =\in A_l$, where A_l is the attribute domain of value a_l. Formally, a pixel bar chart is defined by a five tuple:

$$< D_x, D_y, O_x, O_y, C >$$

where D_x, D_y, O_x, O_y, $\in \{A_l, ..., A_k\} \cup \perp$(the $\perp$ element is used if no attribute is specified) and D_x/D_y are the dividing attributes in x-/y-direction, O_x/O_y are the ordering attributes in x-/y-direction, and C is the coloring attribute. The dividing attributes D_x and D_y are used to partition the data into a small number of partitions. Enumerative data types are mostly used for this purpose. If real or integer dimensions are used for the partitioning, a limited number of ranges of data values has to be determined. The ordering attributes O_x and O_y must have an ordering defined on them, i.e. only continuous or ordinal data types can be used.

The multi-pixel bar charts of sales transactions shown in Figure 5, for example, are defined by the five-tuple:

$$< \text{product type}, \perp, \text{no. of visits}, \text{dollar amount}, \text{C} >$$

where C corresponds to different attributes, i.e., number of visits, dollar amount, quantity.

Formalization of the problem

The basic idea of pixel bar charts is to produce dense pixel visualizations which are capable of showing large amounts of data on a value by value basis without aggregation. The specific requirements for pixel displays are:

- dense display, i.e., bars are filled completely
- no-overlap, i.e., no overlap of pixels in the display
- locality, i.e., similar data records are placed close to each other
- ordering, i.e., ordering of data records according to O_x, O_y

To formalize these requirements we first have to introduce the screen positioning function:

$$f : A_1 \times ... \times A_k \rightarrow Int \times Int,$$

which determines the x-/y-screen positions of each data record d_i, i.e., $f(d_i)=(x,y)$ denotes the position of data record d_i on the screen, and $f(d_i).x$ denotes the x-coordinate and $f(d_i).y$ the y-coordinate. Without loss of generality, we assume that $O_x=A_1$ and $O_y=A_2$. The requirements can then be formalized as:

1. *Dense display constraint*
   The dense display constraint requires that all pixel rows (columns) except the last one are completely filled with pixels. For equal-width bar charts, the width w of the bars is fixed. For a partition p consisting of $|p|$ pixels, we have to ensure that:

$$\forall\, i = 1..w, \forall j = 1..\lfloor |p|/w \rfloor : \exists d_i \text{ with } f(d_i) = (i, j)$$

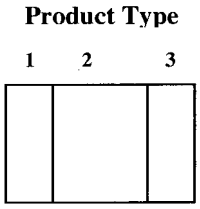

Figure 6 Dividing attribute on x-axis (e.g., D_x=*Product Type*)

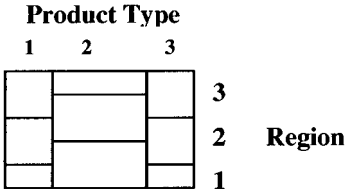

Figure 7 Dividing attributes on x- and y-axis (e.g., D_x=*Product Type*, D_y=*Region)*

For equal-height bar charts of height h the corresponding constraint is:

$$\forall\ i = 1..\lfloor |p|/h \rfloor : \forall j = 1..h : \exists d_i \text{ with } f(d_i) = (i,j)$$

2. *No-overlap constraint*
The no-overlap constraint means that a unique position is assigned to each data record. Formally, we have to ensure that two different data records are placed at different positions, i.e.:

$$\forall\ d_i, d_j \in DB : i \neq j \Rightarrow f(d_i) \neq f(d_j).$$

3. *Locality constraint*
In dense pixel displays the locality of pixels plays an important role. Locality means that similar data records are placed close to each other. The partitioning in pixel bar charts ensures a basic similarity of the data records within a single bar. In positioning the pixels within the bars, however, the locality property also has to be ensured. For the formalization, we need a function $sim(d_i, d_j) \rightarrow$ [0...1] which determines the similarity of two data records and the inverse function of the pixel placement function f^{-1}, which determines the data record for a given (x,y)-position on the screen. An example for the similarity function is the normalized weighted average of the O_x and O_y attribute values. The locality constraint can then be expressed as:

$$\sum_{x=1}^{w}\sum_{y=1}^{h=1} sim(f^{-1}(x,y), f^{-1}(x,y+1)) +$$
$$\sum_{x=1}^{w-1}\sum_{y=1}^{h} sim(f^{-1}(x,y), f^{-1}(x+1,y)) \rightarrow \min$$

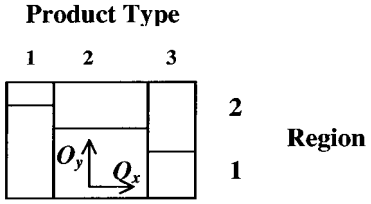

Figure 8 Ordering attributes on x- and y-axis (e.g., O_x=*Dollar Amount*, O_y=*Quantity)*

Note that in general it is not possible to place all similar pixels close to each other while respecting the dense display and no-overlap constraints. This is the reason why the locality constraint is formalized as a global optimization constraint.

4. *Ordering constraint*
The last constraint which is closely related to the locality constraint is the ordering constraint. The idea is to enforce a one-dimensional ordering in x- and y-direction according to the specified attributes O_x=A_1 and O_y=A_2. Formally, we have to ensure:

$$\forall i,j \in 1..n : a_1^i > a_1^j \Rightarrow f(d_i).x > f(d_j).x$$

$$\forall i,j \in 1..n : a_2^i > a_2^j \Rightarrow f(d_i).y > f(d_j).y$$

Note that ordering the data records according to the attribute and placing them in a row-by-row or column-by-column fashion may easily fulfill each one of the two constraints. Ensuring both constraints at the same time, however, may be impossible in the general case. We can formalize the constraint as an optimization problem:

$$\sum_{x=1}^{w}\sum_{y=1}^{h-1}(f^{-1}(x,y).a_1 - f^{-1}(x,y+1).a_1 +$$
$$|f^{-1}(x,y).a_1 - f^{-1}(x,y+1).a_1|)/2 +$$
$$\sum_{x=1}^{w-1}\sum_{y=1}^{h}(f^{-1}(x,y).a_2 - f^{-1}(x+1,y).a_2 +$$
$$|f^{-1}(x,y).a_2 - f^{-1}(x+1,y).a_2|)/2 \rightarrow \min$$

For a perfect ordering of the data in x- and y-direction, the above optimization function provides zero. If the ordering constraint is not satisfied for one or more positions, the function sums up all the differences at those positions.

Note that there may be a trade-off between the x- and the y-ordering constraint. In addition, the optima for the locality and the ordering constraints are in general not identical. This is due to the fact that the similarity function may induce a different optimization criterion than the x-/y-ordering constraints. For solving the pixel placement problem, we therefore have to solve an

optimization problem with multiple competing optimization goals. The problem is a typical complex optimization problem which is likely to be NP-complete and can therefore only be solved efficiently by a heuristic algorithm.

Pixel placement algorithm

For the generation of pixel bar charts, we have to:

- partition the data set according to D_x and D_y
- determine the pixel color according to C
- place the pixels of each partition in the corresponding regions according to O_x, O_y

The partitioning according to D_x and D_y is simple and straightforward to implement, and therefore does not need to be described in detail here. The color mapping is also simple and maps high data values to bright colors and low data values to dark colors. The pixel placement within one bar, however, is a difficult optimization problem because it requires a two-dimensional ordering. In the following, we describe our heuristic pixel placement algorithm which provides an efficient solution to the problem. The basic idea of the heuristic pixel placement algorithm is to partition the data set into subsets according to O_x and O_y, and use those subsets to place the bottom- and left-most pixels. This provides a good starting point which is the basis for the iterative placement of the remaining pixels. The algorithm works as follows (see Figure 9):

1. For an efficient pixel placement within a single bar, we first determine the one-dimensional histograms for O_x and O_y, which are used to determine the α-quantiles of O_x and O_y. If the bar under consideration has size $w \times h$ pixels, we determine the $1/w,\ldots,(w-1)/w$-quantiles for the partitioning of O_x, and the $1/h,\ldots,(h-1)/h$-quantiles for the partitioning of O_y. The quantiles are then used to determine the partitions $X_1,\ldots,X_w$ of O_x and $Y_1,\ldots,Y_h$ of O_y. The partitions $X_1,\ldots,X_w$ are sorted according to O_y and the partitions $Y_1,\ldots,Y_h$ according to O_x.

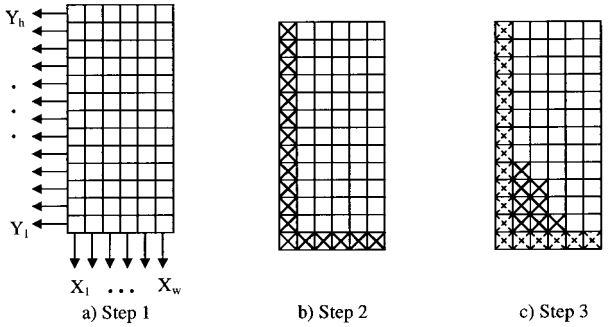

Figure 9 Pixel placement algorithm

2. We start now by placing the pixel in the lower-left corner, i.e. position (1,1), of the pixel bar:

$$f^{-1}(1,1) = \left\{ d_s | \underset{d_s \in X_1}{min} \{d_s.a_2\} \right\} \text{ or}$$
$$f^{-1}(1,1) = \left\{ d_s | \underset{d_s \in Y_1}{min} \{d_s.a_1\} \right\}$$

   depending on which d_s provides the overall lower value. Next, we place all pixels in the bottom and left pixel rows/colums of the bar. This is done as:

$$f^{-1}(i,1) = \left\{ d_s | \underset{d_s \in Y_i}{min} \{d_s.a_2\} \right\} \forall i = 1..w$$

$$f^{-1}(1,j) = \left\{ d_s | \underset{d_s \in Y_j}{min} \{d_s.a_1\} \right\} \forall j = 1..h$$

3. The final step is the iterative placement of all remaining pixels. This is done starting from the lower left to the upper right. If pixels at positions (*i-1, j*) and (*i, j-1*) are already placed, the pixel at position (*i, j*) is determined as:

$$f^{-1}(\mathrm{i},j) = \left\{ d_s | \underset{d_s \in X_i \cap Y_j}{min} \{sum(d_s.a_1, d_s.a_2)\} \right\}$$
$$if\ X_i \cap Y_j \neq < \text{empty set} >$$

   where $sum(a_1, a_2)$ is the weighted sum of a_1 and a_2. Because we have partitioned and sorted the data as mentioned in step 1, the pixel to be placed at each position can be determined in O(1) time if $X_i \cap Y_j \neq <$ empty set $>$. If $\mathrm{X_i} \cap \mathrm{Y_j} = <$ empty set $>$, we have to iteratively extend the partitions X_i and Y_j and consider:

$$d_s \in (X_i \cup X_{i+1}) \cap Y_j.$$

   If this set is still empty, we have to consider:

$$d_s \in (X_i \cup X_{i+1}) \cap (Y_j \cup Y_{j+1})$$

   and so on, until a data point to be placed is found. Note that this procedure is efficient due to the data partitioning and sorting done in step 1.

Figure 9 exemplifies the three steps of the pixel placement algorithm. If n is the number of pixels to be placed, the complexity of determining the quantiles and sorting in step 1 is *O(n logn)*. In step 2 and 3, the pixels can be directly placed (for $X_i \cap Y_j \neq <$ empty set $>$) resulting in a complexity of *O(n)*. The total complexity of the algorithm is therefore *O(n logn)*.

The pixel bar chart system

To analyze large volumes of transaction data with multiple attributes, pixel bar charts have been integrated into a visual data mining system.[24] The system uses a web browser with a Java activator to allow real-time interac-

tive visual data mining on the web. The web interface is based on standard HTML and Java applets, which are used to explore relationships and retrieve data within a region of interest. The server is integrated with the data warehouse and the mining engine. The user at the client side visually explores the data by dynamically accessing the large multi-attribute transactions with complex relationships through HTML pages in a web browser.

System architecture and components

The pixel bar chart system connects to a data warehouse server and uses the database to query for detailed data as needed. The data to build the pixel array is kept in memory to support real-time manipulation and correlation. As illustrated in Figure 10, the pixel bar chart system architecture contains three basic components:

1. *Pixel array ordering and grouping*
   A pixel array is constructed from the pixel bar chart five tuple specification. One pixel represents one data record, i.e., a customer. The partitioning algorithm assigns each data record to the corresponding bar according to the partitioning attribute(s), and the pixel placement algorithm positions the pixel on the display according to the heuristics presented above.

2. *Multiple linked pixel bars*
   In multi-pixel bar charts, the position of the pixels belonging to the same data record remains the same across multi-pixel bar charts for correlation. The colors of the pixel correspond to the value of the selected attributes (such as price, number of orders, etc.).

3. *Interactive data exploration*
   This system provides simultaneous browsing and navigation of multiple attributes with a real time response time of 10–100 milliseconds.

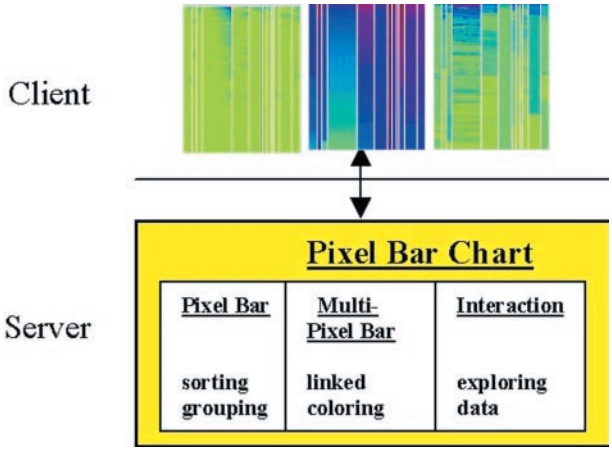

Figure 10 System architecture and components

Interactive data analysis

Interactivity is an important aspect of the pixel bar chart system. To make large volumes of multi-attribute data sets easy to explore and interpret, the pixel bar chart system provides the following interaction capabilities:

- visual querying
- layered drill-down/detail-on-demand
- multiple linked visualizations
- zoom in and out of the pixel bar charts

The attributes used for partitioning (D_x, D_y), ordering (O_x, O_y), and coloring (C) can be selected and changed at execution time. For identifying correlations, a subset of data items in a pixel bar chart can be selected to get the pixels corresponding to related attribute values highlighted within the same display. A drill-down technique allows the viewing of all related information after selecting a single data item. When multi-pixel bar charts are used, pixels reside at the same location across all the charts with different attributes. In addition for discovering correlations and patterns, the user may select a single data item to relate all its attribute values.

Application and evaluation

The pixel bar chart technique has been prototyped in several e-commerce and IT applications at Hewlett Packard Laboratories. It has been used to visually mine large volumes of sales transactions and customer shopping activities at HP shopping web sites. In addition, it has been used to analyze searching behavior in the HP IT Resource Center.

Customer analysis

The pixel bar chart system has been applied to explore customer buying patterns and behaviors. In Figure 11, the pixels of the bar chart represent customers making transactions on the web. In the resulting pixel bar chart, customers with similar purchasing behaviors (i.e., product type, geographical location, dollar amount, number of visits, and quantity) are placed close to each other. A store manager can use the visualization to rapidly discover customer buying patterns and use those patterns to target marketing campaigns.

Figure 11 shows the four attributes of 106,199 customer buying records. The four pixel bar charts of Figure 11 are constructed as follows:

- *Product type* is the dividing attribute D_x (12 product types)
- *Dollar amount* is the x-ordering attribute O_x
- *Region* is the y-ordering attribute O_y (10 United States regions)
- *Region, dollar amount, number of visits* and *quantity* are the four coloring attributes C

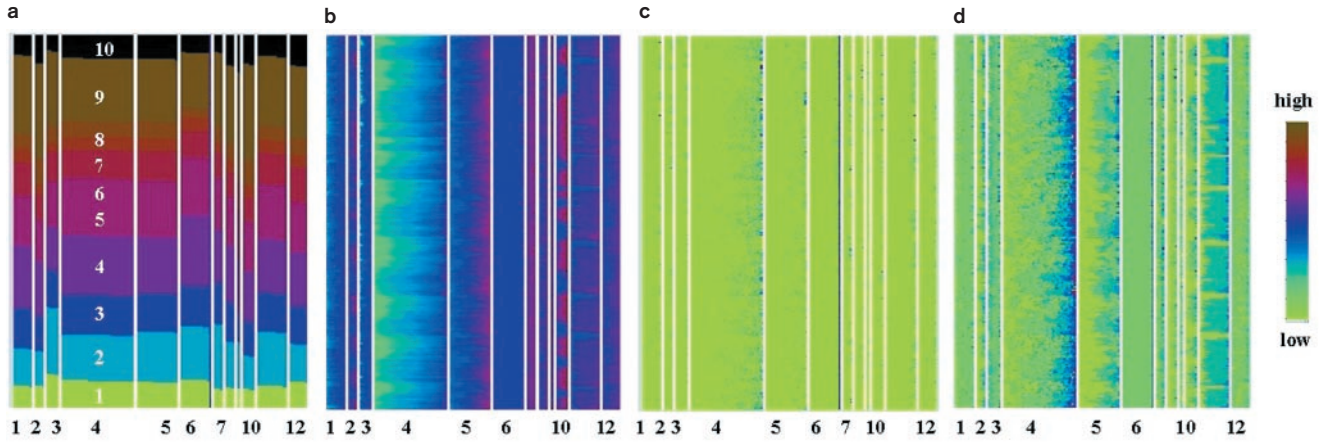

Figure 11 Multi-pixel bar chart for mining 106,199 customer buying transactions. (D_x = *Product Type*, $D_y = \perp$, O_x = *dollar amount*, O_y = *region*, C). (a) Color: region. (b) Color: dollar amount. (c) Color: no. of visits. (d) Color: quantity

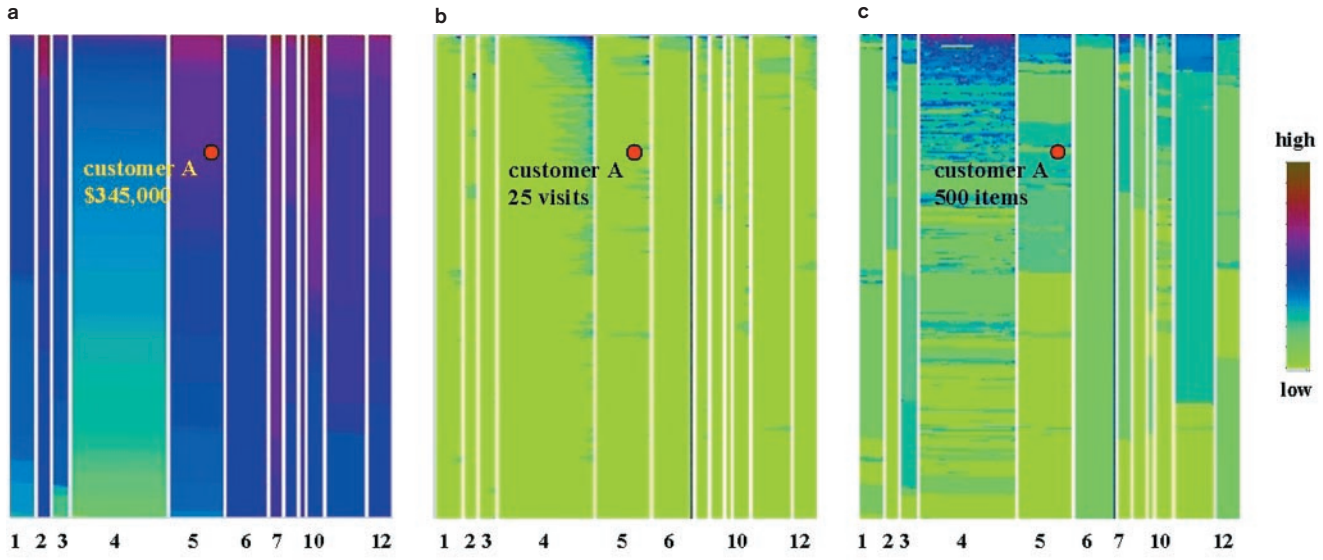

Figure 12 Multi-pixel bar chart for mining 405,000 sales transaction records. (D_x = *Product Type*, $D_y = \perp$, O_x = *no. of visits*, O_y = *dollar amount*, C). (a) Color: dollar amount. (b) Color: no. of visits. (c) Color: quantity

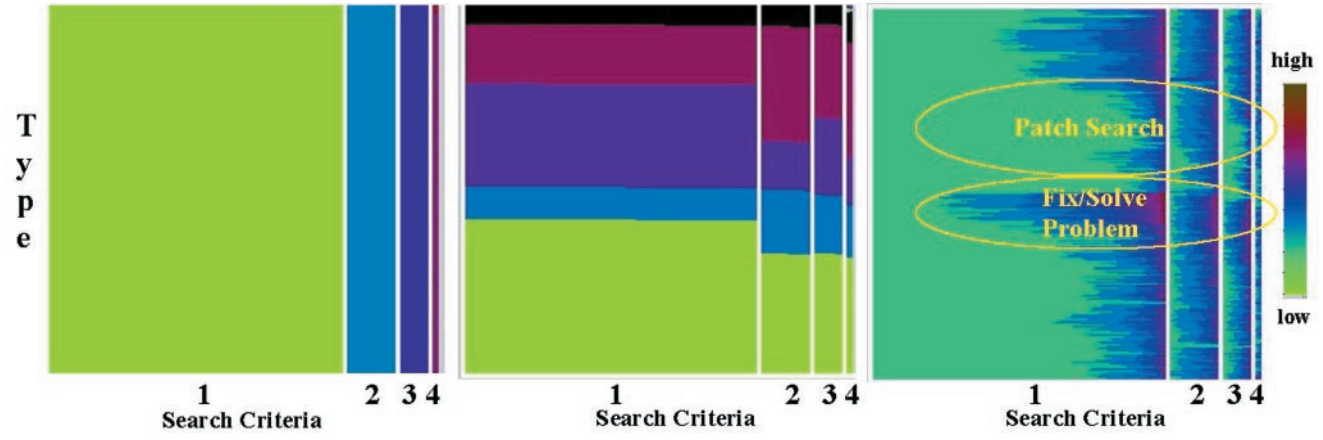

Figure 13 Pixel bar charts for mining 106,199 IT resource center customer activities. (D_x = *Search Criteria*, $D_y = \perp$, O_x + *No. of Keywords*, O_y = *Search Type*, C). (a) Color: search criteria. (b) Color: search type. (c) Color: no. of keywords

Many important facts may be discovered in Figure 11. In the bars for the different attributes, the user may observe for example the following facts:

1. Region *attribute*
   There are 10 different colors to represent 10 different regions (labeled 1–10 in Figure 11a) in the United States. The colored wave indicates the number of customers in each region. Region 4 (largest area) is found to have the largest number of customers. Region 7 (smallest area) has the least number of customers across all product types.

2. Dollar amount *attribute*
   Product types 5, 7 and 10 have the most top dollar amount sales (blue and brown). The dollar amount sales of product types 6 and 7 have a very small variance across all regions (solid blue/brown).

3. Number of visits *attribute*
   The blue color distribution in product type 4 indicates that customers of this product type (consumables) come back more often than customers of other product types.

4. Quantity *attribute*
   The green color of product type 6 indicates that in this category all customers bought the same number of items across all regions. It is also obvious that product type 4 customers have the largest quantities.

By relating the different bars of the multi-bar chart of Figure 11, the user may observe important additional facts such as, for example, the following clusters and trends:

- The top dollar amount customers come back more frequently and purchase larger quantities
- Region 4 has the most customers but region 9 is the most profitable with more frequent visits and larger quantities

Sales transaction analysis

One of the common questions electronic store managers ask is how to use the customer purchase history for improving product sales and promotion. Product managers want to understand which products have the top sales and who are their top dollar amount customers.

An e-commerce manager, for example, needs to answer questions as to which product types have the highest dollar amount customers, how often the customers come back and for which products. The analyses may also be used to determine which products may be impacted when the store issues coupons.

While regular bar charts provide aggregated information on the number of customers by product type (Figure 2), the corresponding pixel bar charts include important additional information such as the dollar amount distribution of the sales. More specifically, a pixel bar chart provides the following additional information:

- Dollar amount *versus* product distribution
- Each customer's detailed information can be drilled down as needed

Figure 12 illustrates an example of a multi-pixel bar chart of 405,000 multi-attribute web sales transactions. The dividing attribute (D_x) is again product type; the ordering attributes are number of visits and dollar amount (O_x and O_y). The colors (C) in the different bar charts represent the attributes dollar amount, number of visits, and quantity. From Figure 12, the following information about the web sales transactions can be obtained:

1. Product type 7 and product type 10 have the top dollar amount customers (dark colors of bar 7 and 10 in Figure 12a)

2. The dollar amount spent and the number of visits are clearly correlated, especially for product type 4 (linear increase of dark colors at the top of bar 4 in Figure 12b)

3. Product types 4 and 11 have the highest quantities sold (dark colors of bar 4 and 11 in Figure 12c)

By clicking on a specific pixel (A), we may find out that customer A visited 25 times, bought 500 items, and spent $345,000 on product type 5.

It is also interesting that there are clusters of darker colors in bar 4 of Figure 12c, which means that there are certain ranges of dollar amount sales for which the quantity tends to be higher than in other segments. This observation is unexpected and may be used to identify the corresponding clusters of sales transactions and make use of the information to further increase the sales. Note that this information cannot be detected by regular bar charts.

Searching behavior analysis

In a third application, we used the pixel bar chart technique to analyze customer search behavior on HP's electronic support site (http://itrc.hp.com). As illustrated in Figure 13, each pixel represents one search transaction. We mapped the search criteria used (Boolean, AllWords, AnyWord, Phrase, labeled 1–4 in Figure 13) together with the selected search type (Product Search, Solve/Fix Problem, Patch Search, ...) and the number of used keywords (i.e., printer, patch, ...) to generate the pixel bar chart. The pixel bar chart technique places customers with similar searching behaviors next to each other based on the above-described parameters. Using the pixel bar chart we are able to visualize the log entries from one

month (several 100 thousands of search record entries) into one consolidated display. The visualization allows us to detect certain customer behaviors as described in detail in the following paragraphs.

In our current search engine interface implementation, our customers use an average of 1.7 keywords per search. This is not enough to return relevant results. Based on our current research we enlarged the search engine query box to increase the number of used keywords in the query. But the average number of used keywords per search was not raised significantly. Using pixel bar charts we were able to easily identify the searches with a much higher average number of keywords (about six keywords).

In further analyzing the visualization we found a customer search cluster for the search type 'Fix/Solve a problem' (marked area in Figure 13c). Based on the data we can derive that the number of keywords used also depends on the search type used. In the next product releases, this information will be used to enhance the user interface to make the usage of more keywords in corresponding searches easier.

In addition, we discovered that most searches for a patch consisted of one keyword string, which probably represented the patch id number and not a real query string (second marked area in Figure 13c). This seems to be a common search type which should also be directly supported by the user interface. Note also that in most of the searches the user did not change the search criteria value settings on the screen (the big green area in Figure 13a), which shows the high importance of the default settings.

The pixel bar chart was the first visual data mining technique that allowed us to visualize such huge data sets in a limited space, providing a good overview but still retaining the capability to drill down into details, because each pixel represents an individual search record. Currently we are using pixel bar charts to analyze and compare the customer search behavior obtained from different HP search engines and customer segments. We expect to get a better understanding of important design issues to be able to further improve the interfaces.

Comparison with other techniques

In this section, we compare the Pixel Bar Chart technique with other well-known information visualization techniques, namely the Parallel Coordinate technique[3,4] and Generalized Spiral technique.[9,11] We applied all three techniques to e-customer purchasing data representing one year of activities at one of the HP e-commerce web-sites. The data set consists of 150,000 data records with the attributes customer, date, purchase amount, number of visits, quantity, and location (region/state). For the comparison we represent the same input data as parallel coordinate visualization (see Figure 14), generalized spiral visualization (see Figure 15), and multi-pixel bar chart (see Figure 16). We deployed the three techniques in experimental studies with application experts. Due to the proprietary nature of the data and the limited number and availability of the application specialists, the experiments do not qualify as formal user studies. In the following, we try to survey our results and exemplify them by comparing visualizations of all three techniques in Figures 14–16.

Let us first briefly introduce the Parallel Coordinate and Generalized Spiral techniques. The *Parallel Coordinate technique* maps the high-dimensional data records onto the 2D screen by:

- using n equidistant axes which are parallel to one of the screen axes and correspond to the dimensions
- scaling the axes to the (min, max) – range of the corresponding dimension
- representing every data item as a polygonal line which intersects each of the axes at the point which corresponds to the value for the dimension

The *Generalized Spiral* technique is a pixel-oriented technique which – similar to the Pixel Bar Chart technique – maps each data value to a colored pixel and presents all data values belonging to one dimension (attribute) in separate subwindows. In case of the Generalized Spiral technique the relevance of the data items with respect to a user-specified query are mapped to color. The arrangement of the data items centers the most relevant data items in the middle of the window, and less relevant data items are arranged in a spiral-shape to the outside of the window. To retain the local clustering of data pixels, which gets lost on a one-pixel-wide spiral, the spiral shape arrangement is combined with a local Hilbert curve (for details see[9]).

In using the three techniques (Parallel Coordinates, Generalized Spiral, and Pixel Bar Chart), interactivity (e.g., in selecting certain query ranges or drilling down to the actual data records) is very important for an effective data exploration. This, however, is difficult to convey by static screen shots. For the visualizations presented in Figures 14–16, we carefully selected useful static displays representing interesting steps in the interactive analysis.

In Figure 14a, we focus on the month June and color all polygonal lines of that month in green. This allows us to detect that the purchase activities in June span the full range of the purchase amounts and region/state code attributes but only a small range of the quantity and number of orders attributes. There is one exception to this general observation, namely the transaction with the highest quantity that is easily visible in Figure 14a. It is further possible to discover that the values of the purchase amount attribute in June are not evenly distributed but it is impossible to see whether this is normal or unusual compared to the other months. In Figure 14b, we show a second example, now focusing on a medium purchase amount. Again it is easy to discern that medium purchase amounts occur in all months and regions/states but are only related to small-medium number of orders and quan-

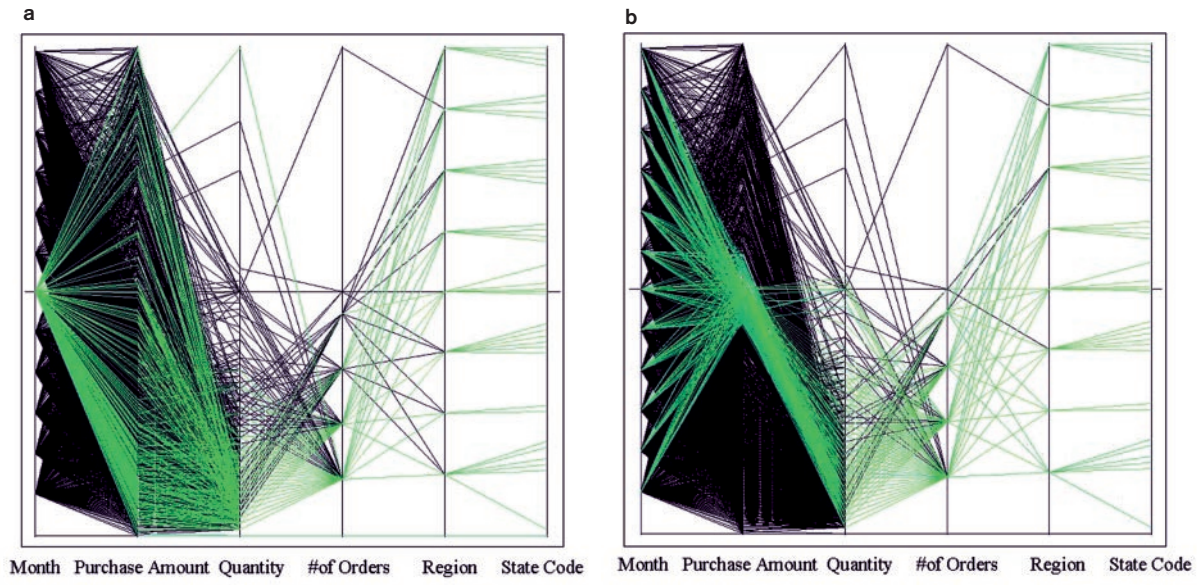

Figure 14 Parallel coordinate visualization of 150,000 E-customer purchasing activities. (a) Focus on one month. (b) Focus on one purchase amount (medium price)

tities. However, due to the high degree of overlap in Figure 14a,b (there are 150,000 polygonal lines!), it is impossible to discern:

- the distribution of attribute values of the colored (or non-colored) polygonal lines
- whether a polygonal line represents a single customer or a large cluster of customers
- the polygonal lines which are occluded by other polygonal lines

and so on. While the Parallel Coordinates technique is a useful interactive exploration technique it falls short of providing a useful overview of the full data set.

In case of the Generalized Spiral technique the user can obtain similar information as from the Parallel Coordinate visualizations. In Figure 15a,b, we used a simple query (all query values were set to zero) and weighting which resulted in a focus on the month attribute (Figure 15a) and purchase amount attribute (Figure 15b). Figure 15a clearly shows the correlation between the month, purchase amount and state attributes, but little correlation to the number of orders and quantity attributes. Figure 15b shows the dependencies from the purchase amount attribute. It is easily visible that there is little correlation to the month and region attributes except for the very small purchase amounts which seem to be clustered in one region (colored green). In the pixel area corresponding to the number of orders attribute there is a clear trend that high (dark) pixel occur more frequently at the outside. This observation reveals that there is a correlation between the purchase amount attribute and the number of orders attribute. A similar correlation but less strong can be detected for the quantity attribute.

These observations show that the pixel-oriented Generalized Spiral technique has clear advantages over the Parallel Coordinate technique, especially with respect to providing a global overview of correlations and trends. However, it also has some disadvantages: It is, for example, more difficult to focus on certain months or price ranges and relate the area corresponding to a specific month (e.g., June) to the corresponding areas of the other attributes. Also, the exceptional high value of the quantity attribute shown in Figure 14a is difficult to discern in the Generalized Spiral visualization of Figure 15a.

The pixel bar chart technique retains all advantages of the Generalized Spiral technique, but at the same time provides a number of additional advantages. The Pixel Bar Chart technique allows the user:

- to focus on certain attribute ranges such as a specific month or purchase amount range
- to identify the distribution of attribute values for specific subsets of the data such as purchase amount for different months
- to relate the attribute values of different attributes for specific subsets of the data

The multi-pixel bar chart shown in Figure 16 is constructed as follows:

- *Time* is the dividing attribute on the x-axis
- *Purchase amount* is the y-ordering attribute
- *Number of visits* is the x-ordering attribute
- *Month, Purchase amount, number of visits*, and *quantity* are the four coloring attributes

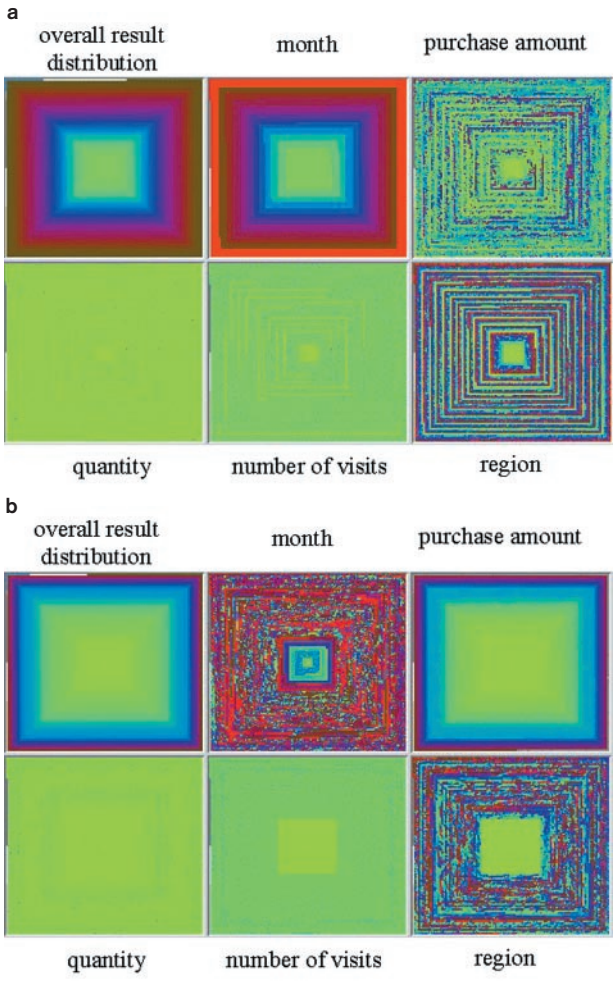

Figure 15 Spiral visualization of 150,000 E-customer purchasing activities. (a) Sorted according to month. (b) Sorted according to purchase

In Figure 16, the following facts – some of which are difficult to detect in the Parallel Coordinate and Generalized Spiral visualizations – can be observed:

1. The month December has the largest number of customers, while February, March, and May have the least number of customers
2. The months February to May have the most top purchase amounts
3. The purchase amount of month December is in the medium price range, although it has the most customers
4. In months March to June customers come back more often than in other months. Christmas customers are mostly one-time customers
5. Most costumers buy more than one item (quantity). The earlier months (February to August) tend to have higher quantities
6. Customers with high purchase amounts tend to come back more frequently and buy more items

Most of these observations cannot be as easily detected by using the Parallel Coordinates or Generalized Spiral techniques. Pixel Bar Charts are advantageous since they:

- provide additional information on the data distributions of the dimensions
- show patterns, correlations and trends between small subsets of the data
- allow users to access detailed information about single customers

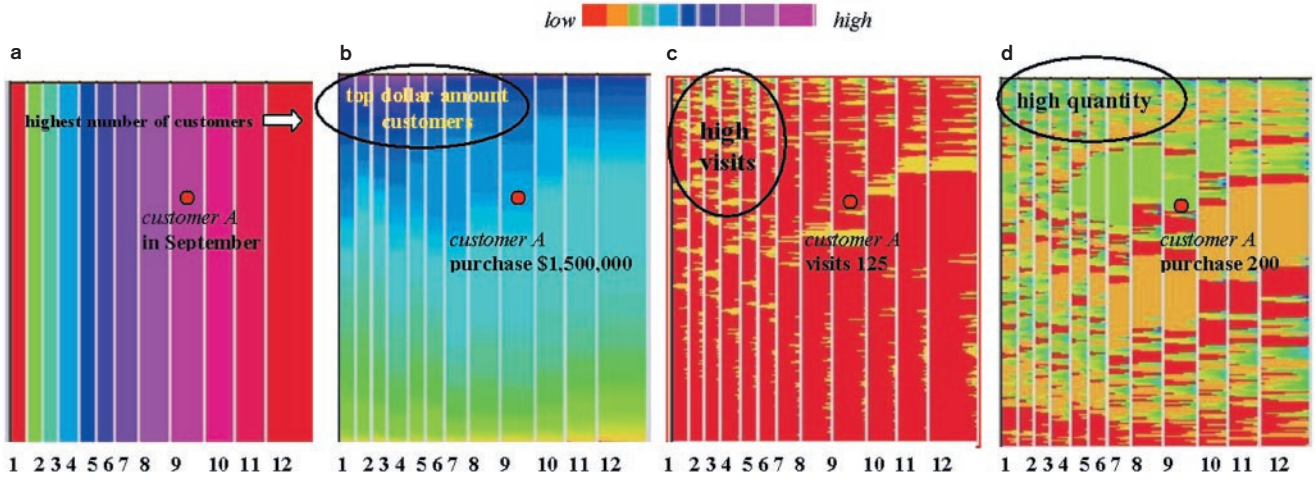

Figure 16 Pixel bar charts for mining over 150,000 E-customer purchasing activities (by month). ($D_x = Month$, $D_y = \perp$, $O_x = no.\ of\ visits$, $O_y = purchase\ amount$, C). (a) Color: month. (b) Color: purchase amount. (c) Color: no. of visits. (d) Color: quantity

In addition, in our experimental studies we observed that – due to the similarity of Pixel Bar Charts to the widely-used bar charts and x-y plots – the pixel bar chart technique was more intuitive compared to the other techniques and was quickly accepted by the application experts. Future work is needed to perform formal user studies evaluating the pixel bar chart technique against a wide range of new and traditional visualization techniques with well-defined tasks and success metrics.

Conclusion

In this article, we presented pixel bar charts, a new method for visualizing large amounts of multi-attribute data. The approach is a generalization of traditional bar charts and x-y diagrams, which avoids the problem of losing information by aggregation and overplotting. Instead, pixel bar charts map each data point to one pixel of the display. For generating the pixel bar chart visualizations, we have to solve a complex optimization problem. The pixel placement algorithm is an efficient and effective solution to the problem. We apply the pixel bar chart idea to real data sets from an e-commerce application and show that pixel bar charts provide significantly more information than regular bar charts. First experimental studies in real world applications show significant advantages of the Pixel Bar Chart technique over existing techniques such as the Parallel Coordinates or Generalized Spiral techniques.

Acknowledgments

Thanks to Sharon Beach of HP Laboratories for her encouragement and suggestions, Julian Ladisch for implementing the Pixel Bar Chart system, Andrian Krug from HP EBusiness R&D, Shu Feng Wei and Brain Ono from HP Shopping for providing data and reviewing the results, Graham Pollock of Agilent Laboratories for his review and comments, Mike Sips for helping with the comparison section, and to Aad van Moorsel and Vijay Machiraju of HP E-Services Software Research Department for their encouragement and support.

References

1 Eick SG. (1999). Visualizing multi-dimensional data with ADVISOR/2000; *VisualInsights.*
2 Shneiderman B. (1996). The eye have it: a task by data type taxonomy for information visualizations. In: *Proc. Visual Languages* (Boulder CO, USA).
3 Inselberg A and Dimsdale B. (1990). Parallel coordinates: a tool for visualizing multi-dimensional geometry. In: *IEEE Symposium on Information Visualization '90* (San Francisco, CA, USA) pp. 361–370.
4 Inselberg A. (1985). The plane with parallel coordinates (Special issue on computational geometry). *The Visual Computer*, **1,** 69–97.
5 Huber PJ. (1985). Projection pursuit. *The Animals of Statistics*, **13:** 435–474
6 Battista GD, Eades P, Tamassia R and Tollis I. (1999). *Graph drawing algorithms for the visualization of graphs.* Prentice Hall.
7 Pickett RM and Grinstein GG. (1988). Iconographic displays for visualizing multidimensional data. In: *IEEE Conf. on Systems, Man and Cybernetics* (Piscataway, NJ, USA), IEEE Press, pp. 514–519.
8 Beddow J. (1990). Shape coding of multidimensional data on a mircocomputer display. In: *IEEE Symposium on Information Visualization '90* (San Francisco, CA, USA) pp. 238–246.
9 Keim, DA. (2000). Designing pixel-oriented visualization techniques: theory and applications. *Transactions on Visualization and Computer Graphics (TVCG)* Vol 16, No 1 pp 59–78.
10 Ankerst M, Keim DA and Kriegel HP. (1996). Circle segments: a technique for visually exploring large multidimensional data sets. In: *IEEE Symposium on Visualization '96, Hot Topics*, (San Francisco, CA, USA).
11 Keim DA and Kriegel HP. (1994). VisDB: database exploration using multidimensional visualization. *Computer Graphics & Applications,* **Sept:** 40–49.
12 Keim DA, Kriegel HP and Ankerst M. (1995). Recursive pattern: a technique for visualizing very large amounts of data. In: *IEEE Symposium on Information Visualization '95* (Atlanta, GA, USA), IEEE Computer Society Press: Chicago, pp. 279–286.
13 LeBlanc J, Ward MO and Wittels N. (1990). Exploring N-dimensional databases. In: *IEEE Symposium of Visualization '90* (San Francisco, CA, USA), pp. 230–237.
14 Robertson G, Card S and Mackinlay J. (1991). Cone trees: animated 3D visualizations of hierarchical information. In: *Proc. ACM CHI Int. Conf. on Human Factors in Computing*, New Orleans, Louisiana, U.S.A. pp. 189–194.
15 Shneiderman B. (1992). Tree visualization with treemaps: a 2D space-filling approach. *ACM Transactions on Graphics,* **11:** 92–99.
16 Anupam V, Dar S, Leibfried T and Petajan E. (1995). DataSpace: 3-D visualization of large databases. In: *IEEE Symposium on Information Visualization '95* (Atlanta, GA, USA), IEEE Computer Society Press: Chicago, pp. 82–88.
17 Ahlberg C, Williamson C and Shneiderman B. (1992). Dynamic queries for information exploration: an implementation and evaluation. In: *ACM CHI Int. Conf. on Human Factors in Computing* (Monterey, CA, USA), pp. 619–626.
18 Buja A, McDonald JA, Michalak J and Stuetzle W. (1991). Interactive data visualization using focusing and linking. In: *IEEE Symposium on Visualization '91* (San Diego, CA, USA), pp. 156–163.
19 Chimera R. (1992). Value bars: an information visualization and navigation tool for multi-attribute listings. In: *ACM Conf. on Human Factors in Computing Systems (CHI '92)* (Monterey, CA, USA), pp. 293–294.
20 Sarkar M and Brown M. (1994). Graphical fisheye views. *Communications of the ACM,* **37:** 73–84.
21 Lamping J, Rao R and Pirolli PA. (1995). Focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In: *ACM CHI Conference on Human Factors in Computing (CHI95)*, pp. 401–408.
22 Lamping J and Rao R. (1994).Laying out and visualizing large trees using a hyperbolic space. In: *ACM Symposium on User Interface Software and Technology* (Marina del Rey, CA, USA), pp. 13–14.
23 Rao R and Card SK. (1994). The table lens: merging graphical and symbolic representation in an interactive focus+context visualization for tabular information. In: *Human Factors in Computing Systems CHI'94* (Boston, MA, USA), pp. 318–322.
24 Hao M, Dayal U, Hsu M, D'eletto R and Becker J. (1999). A Java-based visual mining infrastructure and applications. In: *IEEE Symposium on Information Visualization'99* (San Francisco, CA, USA), pp. 124–127
25 Keim DA, Hao M, Ladisch J, Hsu M and Dayal U. (2001). Pixel bar charts: a new technique for visualizing large multi-attribute data sets without aggregation. In: *IEEE Symposium on Information Visualization'2001* (San Diego, CA, USA).