# Value-Cell Bar Charts for Visualizing Large Transaction Data Sets

Daniel A. Keim, *Member*, *IEEE Computer Society*, Ming C. Hao, *Member*, *IEEE*, Umeshwar Dayal, *Member*, *IEEE Computer Society*, and Martha Lyons

**Abstract**—One of the common problems businesses need to solve is how to use large volumes of sales histories, Web transactions, and other data to understand the behavior of their customers and increase their revenues. Bar charts are widely used for daily analysis, but only show highly aggregated data. Users often need to visualize detailed multidimensional information reflecting the health of their businesses. In this paper, we propose an innovative visualization solution based on the use of value cells within bar charts to represent business metrics. The value of a transaction can be discretized into one or multiple cells: high-value transactions are mapped to multiple value cells, whereas many small-value transactions are combined into one cell. With value-cell bar charts, users can 1) visualize transaction value distributions and correlations, 2) identify high-value transactions and outliers at a glance, and 3) instantly display values at the transaction record level. Value-Cell Bar Charts have been applied with success to different sales and IT service usage applications, demonstrating the benefits of the technique over traditional charting techniques. A comparison with two variants of the well-known Treemap technique and our earlier work on Pixel Bar Charts is also included.

**Index Terms**—Information visualization, multivariate visualization, visualization techniques, methodologies.

## 1 Introduction

The rapid increase of automatically collected data has led to the availability of large volumes of business transaction data. Many research efforts have focused on how to turn the raw data into valuable information. For the exploration of large multidimensional data, analysts want to identify problem areas at a glance, with the possibility to easily drill into problem areas to get detailed information. Therefore, we need to present an overview of the data and at the same time show detailed information for each data item. For the exploration of large volumes of multiattribute data, the current charts and tables are not able to show important information such as

- data distribution of multiple attributes,
- patterns, correlations, trends, and exceptions, and
- detailed information, for example, each sales transaction with price, location, time, and so forth.

### 1.1 Related Work

Bar Charts are the most simple presentation graphics; however, bar charts show only aggregated values such as total sales for each month, as shown in Fig. 1. In bar charts, due to the aggregation, valuable information often gets lost. The usefulness of bar charts is especially limited if the user is interested in relationships between the different attributes such as product price, number of orders, and quantities. The reason for this limitation is that multiple bar charts for different attributes do not support the discovery and correlation of interesting subsets, which is one of the main tasks in mining transaction data.

A number of visualization techniques have shown the usefulness of visual data exploration [4], [5], [6] for discovering patterns and trends in multidimensional data. For example, Tableau's visual spreadsheet [11] and SpotFire's visual data exploration interface [10] are widely used by business managers to make daily decisions. Space-filling techniques such as squarified treemaps [2] and screen-filling curves [12] are also used for visualizing hundreds or even thousands of data records.[1] For visualizing even larger amounts of data, pixel-oriented techniques, which use each pixel to represent one data value, can be used. In the VisDB system [8], for example, each pixel is arranged and colored to indicate an item's relevance to a user query. Another well-known technique that uses pixel-level visualization is the Seesoft software visualization technique [15], which maps each line of source code to a line of pixels. Pixel techniques have also been used to build interactive decision tree classifiers based on a visualization of the training data [1].

### 1.2 Two Ideas to Solve the Problem

To explore large transaction databases, we previously introduced Pixel Bar Charts [7] as a technique for visualizing detailed transaction data. Pixel Bar Charts are derived from bar charts and present the transaction directly instead of aggregating them into a few average values.

The approach is to represent each transaction (for example, the price of a transaction) by a single pixel in the bar chart, as shown in Fig. 2. The detailed information of

- *D.A. Keim is with the Computer and Information Science Department, University of Konstanz, 178957 Konstanz, Germany. E-mail: keim@informatik.uni-konstanz.de.*
- *M.C. Hao and U. Dayal are with the HP Palo Alto Laboratories/Advanced Database Program, Hewlett Packard Company, 1501 Page Mill Road, Palo Alto, CA 94304-1126. E-mail: {ming.hao, umeshwar.dayal}@hp.com.*
- *M. Lyons is with the HP Palo Alto Laboratories/HP Service Headquarter, Hewlett Packard Company, 1501 Page Mill Road, Palo Alto, CA 94304-1126. E-mail: martha.lyons@hp.com.*

[1] We discuss two treemap variants in more detail in Section 5.

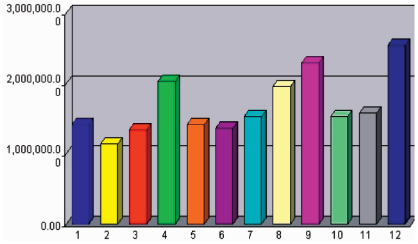


Fig. 1. Monthly sales bar chart showing aggregated data values.

an attribute such as the price of a transaction is encoded in the pixel color. The real value can be displayed as needed. The size of a bar corresponds to the number of transactions, but not the total value of the transactions as in traditional bar charts. For example, store managers can see in the Pixel Bar Chart in Fig. 2 that there are a large number of green and yellow colored transactions (under $50) in all bars, but do not see how much (in dollars) these low sales contribute to their business.

While monitoring and analyzing business operations, analysts should be able to visualize the data based on both the number of transactions (as in pixel bar charts) and the total value of the transactions (as in traditional bar charts) to be able to identify the transactions that are important to their business. In this paper, we therefore combine the two concepts and introduce the concept of value-cell bar charts. Value-Cell Bar Charts represent a transaction value (the price of an invoice) using a rectangular cell inside a bar, as shown in Fig. 3. The height of each bar corresponds to the total transaction value as in traditional bar charts, but the important individual transactions (transactions with a high value) are still directly visible within the bars. Transactions with high values correspond to multiple cells, whereas multiple small transactions may be represented by one cell.

In Section 2, we describe the basic concepts of Value-Cell Bar Charts and the algorithm to generate them. In Section 3, we provide several application examples and show the usefulness of value-cell bar charts, and, in Sections 4 and 5, we compare them to Pixel Bar Charts and two Treemap variants.

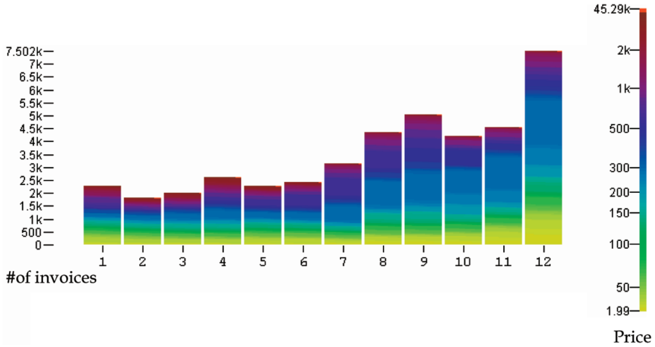


Fig. 2. Pixel bar chart showing sales transaction distribution ($x$-axis: month; $y$-axis ordering: *price*; *color*: *price*).

## 2 Our Approach—The Basic Idea of Value-Cell Bar Charts

The basic idea of Value-Cell Bar Charts is to partition the bars of traditional bar charts into cells. Each cell corresponds to a fixed portion of the aggregated value of the bar. The number of cells corresponding to a transaction depends on its value, for example, the price of the invoice. High-value transactions are mapped to multiple cells. Many low-value transactions may be combined into one cell. Value-Cell Bar Charts use fixed width bars.

### 2.1 Value-Cell Definition

Value-Cell Bar Charts are designed for visualizing any type of transaction data. Transaction data sets are usually large multiattribute data sets with some categorical or nominal

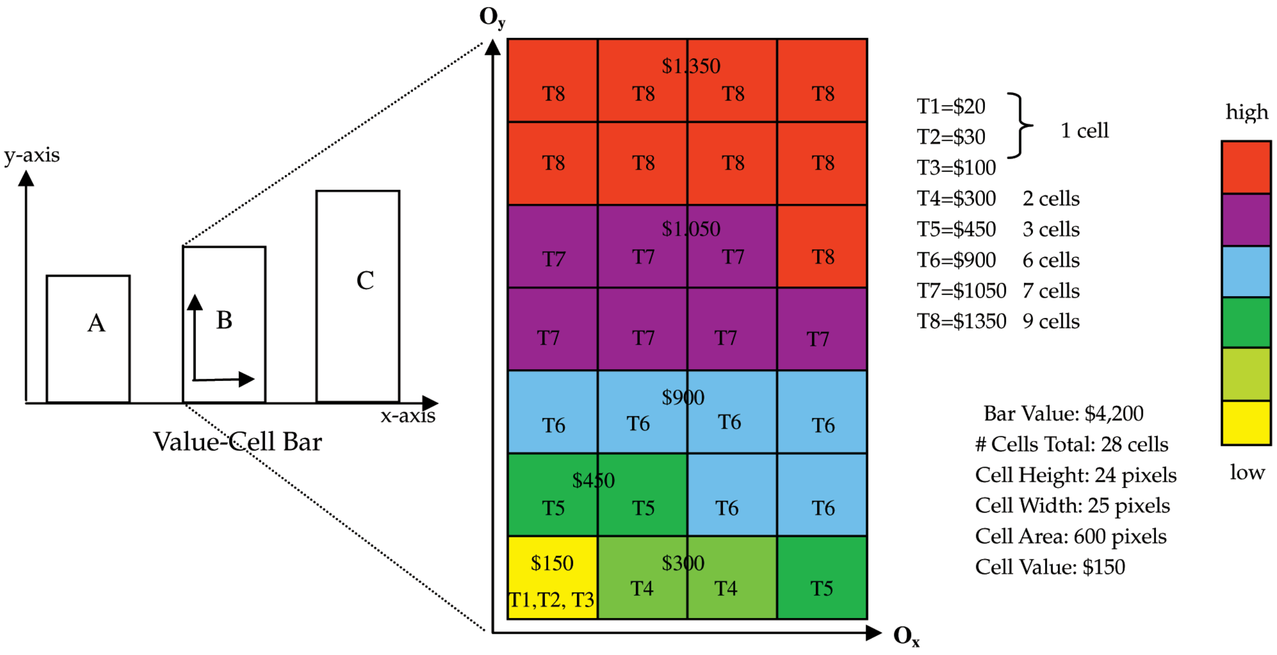


Fig. 3. Definition of value-cell bar charts.

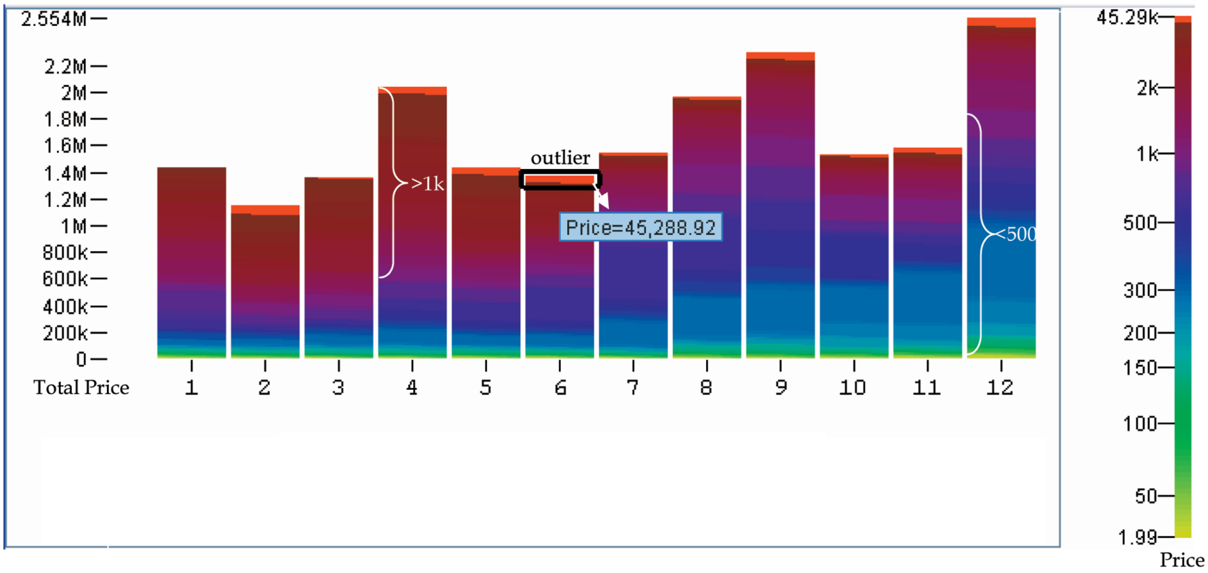


Fig. 4. Monthly value-cell bar chart showing sales value distribution and an outlier ($x$-axis: month; $y$-axis ordering: *price*; *color*: *price*).

attributes and some numeric attributes. The categorical or nominal attributes (for example, month) are used for partitioning the data into bars, whereas the numeric attributes (for example, price) are used to determine the height of the bars. To show individual transaction values, which are not shown in traditional bar charts, all bars are partitioned into **value cells**, as indicated in Fig. 3. A cell represents a certain value and is visually shown as a small rectangle of fixed size (not necessarily with a border). The value cell size is defined by the total value of the bar ($BarValue$), the bar area ($BarWidth * BarHeight$), the cell value ($CellValue$), and the number of Cells in X-direction within the bar ($\#CellsX$). Under the assumption that there are no rounding effects, the cell size can then be computed as

$$CellWidth = \frac{BarWidth}{\#CellsX},$$
$$CellHeight = \frac{CellValue}{CellWidth} \cdot \frac{BarWidth \cdot BarHeight}{BarValue}.$$

The bar width is fixed. The bar height varies according to the total bar value. Now, considering rounding effects, we obtain

$$\#CellsY = Round\left(\frac{BarHeight}{CellHeight}\right).$$

The total number of cells is

$$\#CellsTotal = \#CellsX \cdot \#CellsY.$$

$CellValue$ and $\#CellsX$ are parameters of the algorithm, which can be automatically adapted according to the input data. For example, in Fig. 3, the width of B bar is 100 pixels, the height is 168 pixels, and the bar total value is $4,200. The parameter $\#CellsX$ is set to 4 and $CellValue$ is $150. In this case, we get 25 pixels as $CellWidth$ and 24 pixels as $CellHeight$. The cell size is 600 pixels (25 × 24) as shown in the B bar.

The cell color corresponds to a value of the transaction. In the special case, when multiple transactions are aggregated into one cell, the color corresponds to the average value of the transactions contained in the cell. We use a color map ranging from yellow for low values, through green, blue, and burgundy, to red for high values. In Fig. 3, for example, the transactions T1, T2, and T3 are combined into one yellow cell, whereas transaction T8 is represented by nine red cells.

The color map has been carefully chosen to show the main variance of the data by the brightness of the color scheme while still using a wide range of colors. The only exception are the red colors on the top end of the color scheme in Figs. 4, 9, 10, and 11, which were added to show high exception values. Note that the choice of the color scheme depends on the specific application needs, and different color maps can be used in our system to fulfill those needs.

### 2.2 Value-Cell Algorithm

The value-cell placement within one bar is an interesting problem. The task is to minimize the error introduced by mapping transaction values to cells. In doing this, there is a trade-off between minimizing the overall error for the height of the bar corresponding to the aggregated value and the discretization error for each transaction. If we simply round each transaction value to the closest number of cells (normal rounding), we may introduce a very large overall error in cases where most transaction values have to be rounded up (or rounded down) by a significant margin. Our algorithm tries to keep the overall error small by considering the overall rounding error in each step. The algorithm determines the number of cells for each transaction t ($\#\mathrm{Cells}_t$) such that all cells together fill up the bar, that is, we have a total of

$$\sum\nolimits_t \#Cells_t = \#CellsTotal$$

cells, and the sum of all discretization errors

$$\sum\nolimits_t |DiscretizationError_t|$$

should be minimal.

The algorithm to determine the Value-Cell Bar Chart tries to determine the optimal number of cells for each transaction.

This may lead to an error in the total height of the bar, which may be too high in cases of positive rounding effects or too low in cases of negative rounding effects. Since errors in the total height are not tolerable, in these cases, we have to adjust the number of cells for some transactions. The algorithm uses two lists, the PositiveErrorList and the NegativeErrorList, which are both sorted according to the discretization error, to adjust the number of cells for transactions that have the highest discretization error. If the total bar size is too high, we iteratively take the first (*GetFirst*) transaction from the *NegativeErrorList* until the total bar size is correct; if the total bar size is too low, we use the *PositiveErrorList* instead. The whole algorithm works as follows:

```
1. Calculate $CellWidth$ and $CellHeight$ (see Section 2.1).
2. Sort the transactions $t$ by increasing transaction value $Value_t$.
3. For each transaction $t$ calculates:
     $\#CellsOpt_t = Round(\frac{Value_t}{CellValue})$,
     $\#CellsMin_t = \lfloor\frac{Value_t}{CellValue}\rfloor$ $\#CellsMax_t = \lceil\frac{Value_t}{CellValue}\rceil$,
     $DiscretizationError_t = \frac{Value_t}{CellValue} - Round(\frac{Value_t}{CellValue})$.
4. For each transaction $t$:
        $\#Cells_t = \#CellsOpt_t$
      Combine small cells;
5. If
   $\sum_t \#Cells_t <> \#CellsTotal$
   then
   For each transaction $t$:
     If
       $\#CellsOpt_t == \#CellsMin_t$
     then Insert (t, PositiveErrorList)
     else Insert (t, NegativeErrorList)
6. If $\sum_t \#Cells_t > \#CellsTotal$
   then
   Sort NegativeErrorList according to decreasing
       $|DiscretizationError_t|$
     Repeat
       $tmp = Getfirst(NegativeErrorList)$
       $\#Cells_{tmp} := \#CellsMin_{tmp}$
     until
   $\sum_t \#Cells_t = \#CellsTotal$
   else
   Sort PositiveErrorList according to decreasing
       $|DiscretizationError_t|$
   Repeat
       $tmp = Getfirst(PositiveErrorList)$
       $\#Cells_{tmp} := \#CellsMax_{tmp}$
   until
       $\sum_t \#Cells_t = \#CellsTotal$
7. Draw the cells starting in the bottom left corner with the
   transaction with the lowest value and proceeding to the
   upper right corner for the transaction with the highest
   value. The cell color corresponds to the value of the
   transaction contained in the cell. If the cell contains more
   than one transaction, then the cell color is the average
   value of all transactions in the cell.
```

As illustrated in Fig. 3, the B bar contains eight transactions (T1-T8). The yellow cell, at the bottom left of the bar, contains three transactions (T1, T2, and T3) corresponding to transaction values \$20, \$30, and \$100. The nine top red cells represent transaction T8 (\$1,350). The mapping of the other transactions (T4-T7) into cells is listed at the right side of Fig. 3. Note that, in this simple example, no rounding effects occur, and therefore, the total bar size corresponds exactly to the desired bar size, which means that Steps 5 and 6 will not be executed.

It can be shown that the algorithm always terminates and that $\sum_T |DiscretizationError_T|$ is minimal. The complexity of the algorithm is $|T| \log |T|$ since the algorithm is dominated by the sorting (Steps 2 and 5), where $T$ is the total number of transactions.

### 2.3 Advantages of Value-Cell Bar Charts

The advantages of Value-Cell Bar Charts for visualizing transaction data are

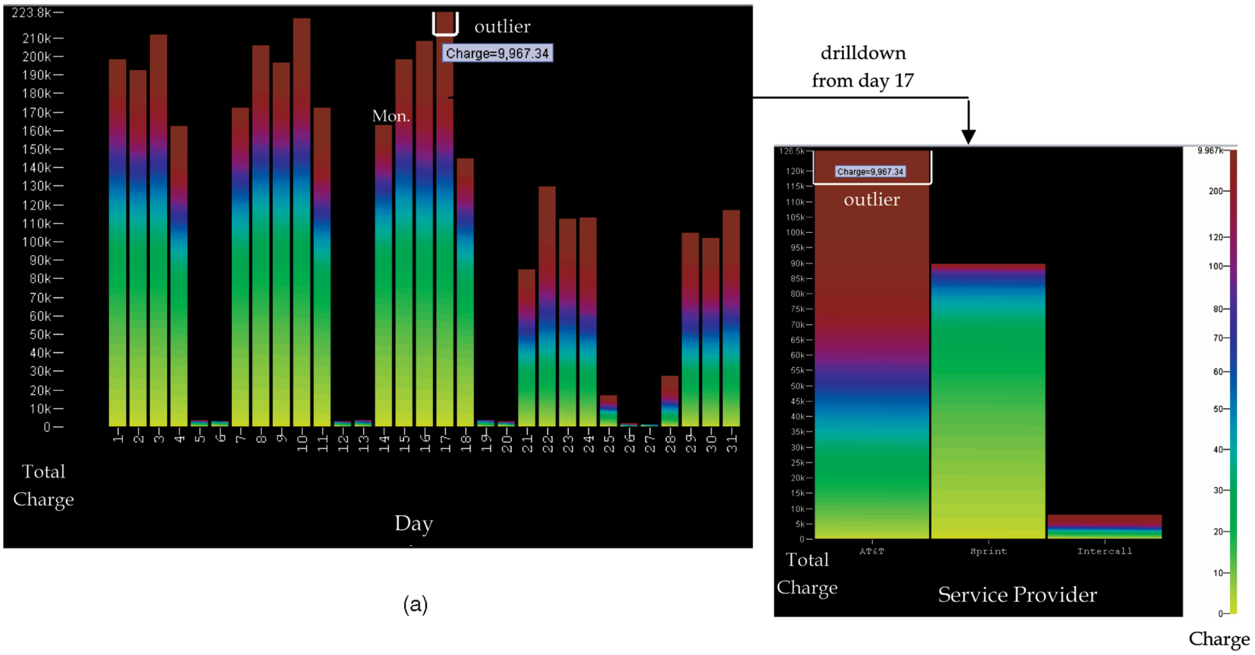
- bars heights of Value-Cell Bar Charts corresponds to aggregated transaction values (as in normal bar charts);
- transaction value distributions, correlations, and outliers are consistently visualized within the value-cell bar chart;
- transactions that contribute the most to the total value can be easily identified since high-value data are mapped into multiple cells and multiple low value data into one cell; and
- multiple variables can be shown by multiple value-cell bar charts with the same layout, allowing an analysis of their relationships.

Note that Value-Cell Bar Charts are different from existing visualization techniques: On one side, they differ from bar charts since they show each individual transaction as long as its value corresponds to one cell. Even stacked bar charts show only few data values (for example, a few categories) per bar, but not individual transactions. On the other side, value-cell bar charts differ from space-filling layouts such as treemaps or Pixel Bar Charts since they retain the powerful and widely used bar chart paradigm that the bar height corresponds to the aggregated transaction value.[2] The comparisons provided in Sections 4 and 5 will show the differences more clearly.

## 3 Applications

We applied the Value-Cell Bar Chart technique to a number of real-world sales and service usage (for example, audioconference and telephone) data sets. For example, the sales team uses Value-Cell Bar Charts to determine which parts of the sales contribute the most and to compare sales patterns and trends for planning sales promotions. The IT service managers also use Value-Cell Bar Charts to observe daily usage distributions and correlations and to drill into problem areas to get detailed information to optimize the cost.

[2] The development of Value-Cell Bar Charts was triggered by the daily use of Pixel Bar Charts in a sales analysis application where the users wanted the bar height to correspond to the aggregated sales value instead of the number of transactions.



(a)

(b)

Fig. 5. (a) Audioconference daily charge distribution showing the light and dark green areas contribute the most to the total charges. ($x$-axis: day; $y$-axis ordering: *charge*; *color*: charge). Showing 1) the long conferences (burgundy) at the top of the bars, above 3,000 minutes, correlate to high charges (no spotted colors), 2) the short conferences (yellow and green, under 500 minutes) at the bottom of the bars correlate with low charges (no spotted colors), and 3) a large number of spotted blue colors is in the green and blue areas (1,000 to 2,000 minutes) with low correlation to charges. (b) A drill down from day 17 showing that Sprint has the largest green area (charges under $30) ($x$-axis: service provider; $y$-axis ordering: *charge*; *color*: charge). Showing 1) the conferences under 30 participants (green and yellow) correlate to low charges (less spotted colors) and 2) the conference above 100 participant (burgundy and blue) not correlate to high charges (spotted colors).

### 3.1 Sales Value Distribution Analysis

Fig. 1 illustrates a traditional monthly sales bar chart showing the total sales price by month.

When only aggregated values are shown, important information may get lost. Fig. 2 illustrates a Pixel Bar Chart showing the monthly sales volume distribution colored with each transaction value. In Fig. 4, we use the same data (42,074 transactions) to construct a Value-Cell Bar Chart, showing the sales value distributions. The size of a bar represents the total value of a month. The value of a transaction is discretized into one or multiple cells. The large red and burgundy areas contain transactions with high values above $1,000. The reason why the red and burgundy areas occupy more space is because they contribute more to the total sales amount than the low-value transactions in the green and yellow areas. Each month has the high-value transactions (large areas in red).

In Month 4, the high-value transactions (red and burgundy) contribute an unusual high amount to the total sales. However, in Month 12, the medium and low transactions under $500 (blue, green, and yellow areas) contribute more to the sales than the high-value transactions above $1,000 (red and burgundy areas). Users can drill down to the detailed information on an outlier such as the highest transaction value, $45,288,920 in Month 6.

### 3.2 IT-Service Usage and Correlation Analysis

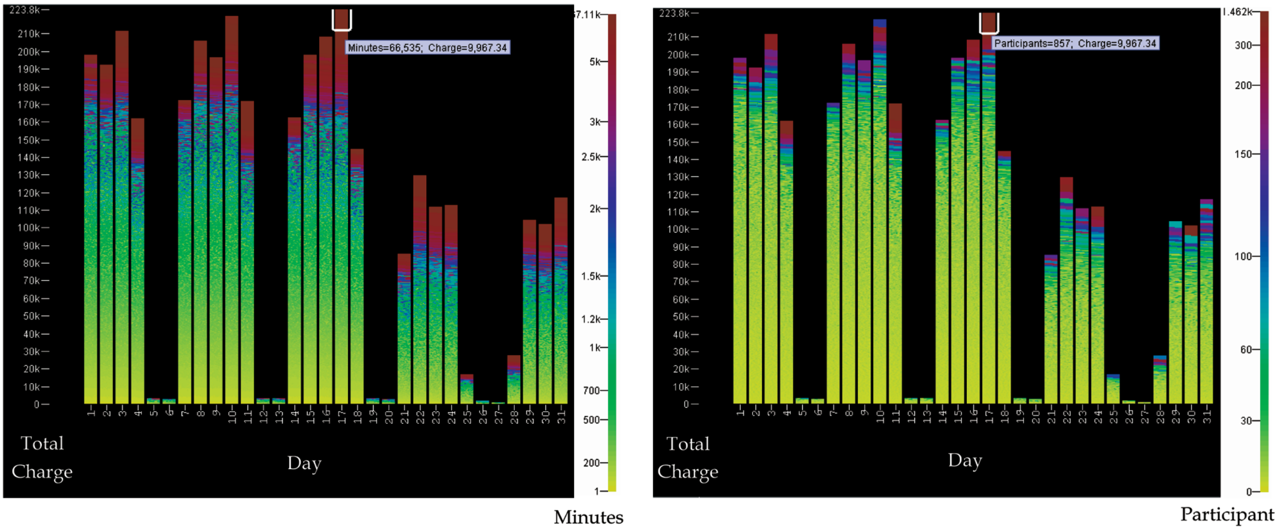
In an audioconference application (233,238 conference records), service managers want to analyze the usage patterns and correlations among different attributes (that is, charges, the length of the phone conferences, and the number of participants) to detect potential cost savings. Examples of what customers typically ask are listed as follows:

- What are the users calling behaviors? What are the charges? Which day has the most calls?
- Who is my best service provider?
- Does the conference charge correlate with the length of the conferences and the number of participants?

From Value-Cell Bar Chart in Fig. 5a, service managers can easily answer the first question, "The large light and dark green areas ($10-$40) show substantial contributions to the total conference charges." The weekly usage pattern indicates that Tuesdays through Thursdays has higher usages than Mondays and Fridays. Another interesting observation is a drop in yellow cells (under $10) in the last two weeks.

To answer the second question, service managers can drill down into a day to see a breakdown of costs by service providers, as illustrated in Fig. 5b. They can quickly see that Sprint has the largest green area with much smaller charges (under $30) than the other two service providers (AT&T and Intercall).

Value-Cell Bar Charts can be used to visualize correlations between different attributes. Figs. 6a and 6b show the correlation between the charges of a conference and the length of the conference or the number of participants. In Fig. 6a, cells are arranged by charges from the lowest/bottom to the highest/top. Each cell is colored by the length of the conference. Service managers can see by looking at the pattern of colors in the bars that the longer length of



(a)

(b)

Fig. 6. (a) Correlation between audioconference charges and minutes ($x$-axis: day; $y$-axis ordering: charge; *color*: minutes). Showing 1) the long conferences (burgundy) at the top of the bars, above 3,000 minutes, correlate to high charges (no spotted colors), 2) the short conferences (yellow and green, under 500 minutes) at the bottom of the bars correlate with low charges (no spotted colors), and 3) a large number of spotted blue colors is in the green and blue areas (1,000 to 2,000 minutes) with low correlation to charges. (b) Correlation between audioconference charges and participants ($x$-axis: day; $y$-axis ordering: *charge*; *color*: *participants*). Showing 1) the conferences under 30 participants (green and yellow) correlate to low charges (less spotted colors) and 2) the conference above 100 participant (burgundy and blue) not correlate to high charges (spotted colors).

conferences should have the same changing pace as the higher charged conferences. The high-charge conference at the top of a bar correlates to a very long conference (above 66,000 minutes) and more than 800 participants as shown by white rectangles in Figs. 6a and 6b.

Spotted color cells indicate that the two attributes do not perfectly correlate. Service managers can find the degree of the correlations by comparing the spotted areas. By observing Figs. 6a and 6b, service managers can conclude and answer the third question that conference charges are more correlated to the conference time than the number of participants in the long conferences (above 3,000 minutes). Conference charges are less correlated than the number of participants in the medium conferences (1,000-2,000 minutes).

## 4 Complementary Properties of Value-Cell Bar Charts and Pixel Bar Charts

Pixel Bar Charts can also be used to visualize transaction-level data. However, there is an important difference: Pixel Bar Charts show each transaction independent of its value by one pixel, whereas in Value-Cell Bar Charts, one transaction may correspond to a large number of cells depending on its value. In addition, traditional Pixel Bar Charts are space-filling arrangements, whereas in Value-Cell Bar Charts, the height of the bar corresponds to the aggregated value.

Fig. 7a shows a traditional Pixel Bar Chart [7]. The width of the bar represents the number of transactions (invoices), and the color represents the value (price) attribute from low (yellow, $1.99) to high (burgundy, $45,290). For the purpose of comparison, we implemented a variant of nonspace-filling variant of Pixel Bar Charts with fixed width and the height corresponding to the **number of transactions** not their value. Fig. 7b shows the resulting visualization. The pixels within a bar are sorted and arranged from left to right and bottom to top to show the distribution of transactions relative to the price attribute.

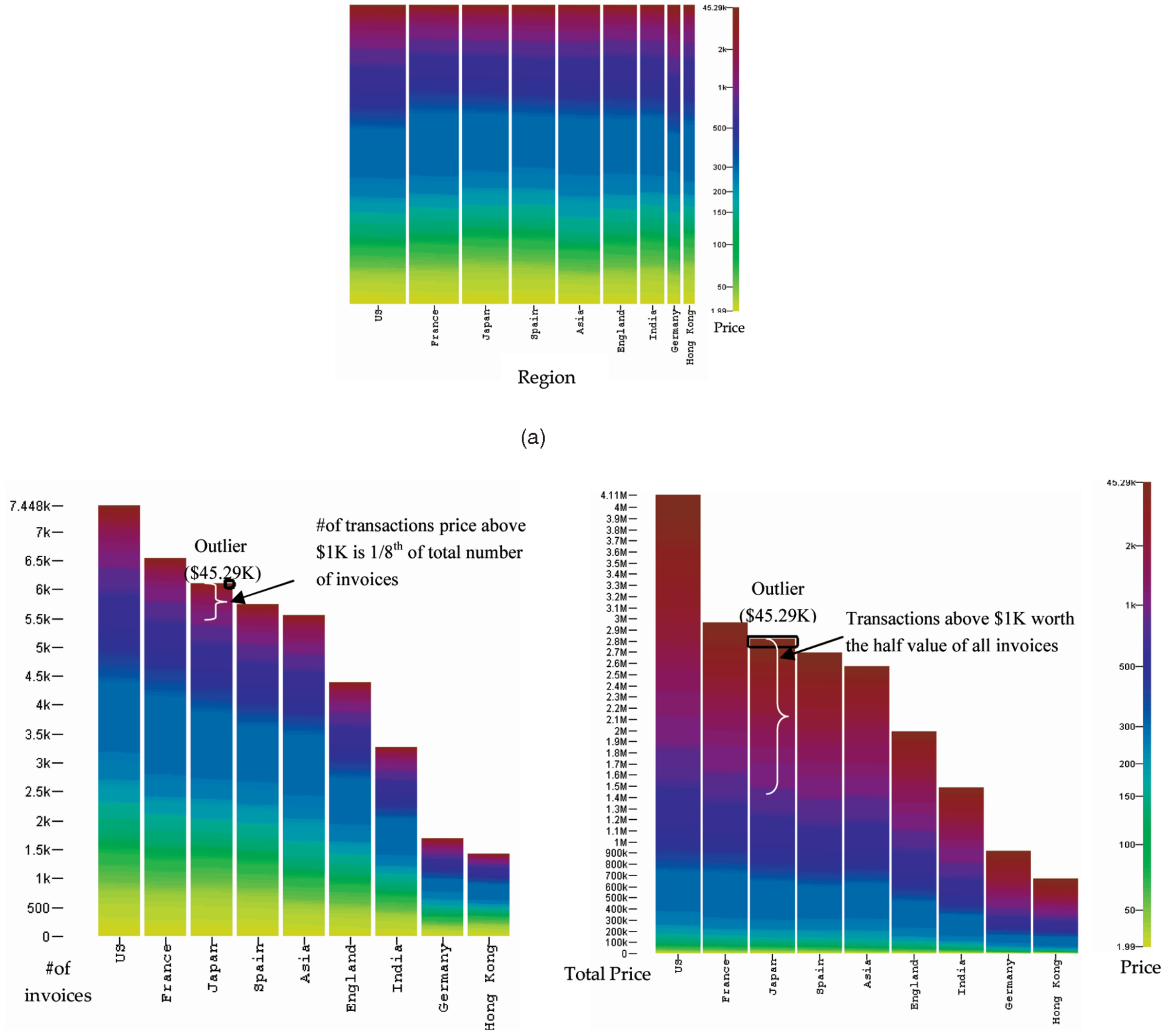
In contrast, Fig. 7c shows a Value-Cell Bar Chart, which provides a view of the **transaction value** distribution. Users can easily find an outlier ($45,290) on the top of the Japan bar with its larger area. Also, users can find that the sales in the burgundy areas have transactions ($1,000) that are more important (larger area) than the sales from the blue areas ($300-$500).

It is interesting to compare Figs. 7b and 7c. For example, there are many transactions under $150 shown as green and yellow areas in Fig. 7b, but their total value is low as shown by the small green and yellow areas in Fig. 7c. Also, Fig. 7b shows that there are only about 1/8th of the transactions are above $1,000 (burgundy). Fig. 7c shows that they worth roughly the half value of all transactions (half the bar area in Japan is burgundy). The key differences between the two charts in Figs. 7b and 7c are summarized in Table 1.

Both Value-Cell Bar Charts and Pixel Bar Charts complement each other: one provides a value view and the other provides a volume view. Users need both of them to have a complete picture of their business, without having to sift through many pages of tables and charts to understand their data.

## 5 Comparison of Value-Cell Bar Charts with Treemaps and Stacked Bar Charts

Treemaps [9] are probably the most well-known space-filling technique for visualizing hierarchically structured data by value. It has been continuously improved with many effective algorithms (for example, [3], [13]). To address the question "Could we use treemaps to achieve the same results as Value-Cell Bar Charts?" we employ strip treemaps and squarified treemaps with and without cushion style rendering. The two variants of treemaps use



(a)

(b)

(c)

Fig. 7. (a) A traditional pixel bar chart where each pixel represents one transaction ($x$-axis: region; $y$-ordering: *price*; *color*: *price*). (b) New pixel bar chart variant where the height corresponds to the number of invoices (transactions). Each pixel represents one invoice. The area of the bar represents the number of invoices ($x$-axis: region; $y$-ordering: price; color: price). (c) Value-cell bar chart one invoice (transaction) is represented by multiple cells or a portion of a cell according to its price (value) ($x$-axis: region; $y$-ordering: *price*; *color*: *price*).

areas and colors to represent data values and are Treemap variants closely related to the Value-Cell Bar Chart technique. The sales data set has two hierarchical levels. The first level is sales regions. The second level is sales transactions, as shown in Fig. 8.

Fig. 9a illustrates a strip treemap showing 357 invoices, and Fig. 9b illustrates a squarified treemap showing 42,074 invoices. Fig. 9c shows a strip cushion treemap with 1,025 invoices, and Fig. 9d shows a squarified cushion treemap showing the same data, as in Fig. 9b. The corresponding Value-Cell Bar Charts are shown in Figs. 9e and 9f.

## 5.1 Comparison to Strip Treemaps

In Fig. 9a, the strip treemap uses an alternating vertical and horizontal splitting to fill each vertical strip (region) with horizontal strips (transactions). Each small strip represents a transaction. The size and color of the strip represents the value of the transaction. In Fig. 9c, a strip treemap without borders but with cushion style rendering is shown for a larger data set, and in Fig. 9e, the corresponding Value-Cell Bar Chart fills the bar (region) with value cells. The color of the cells represents the value of the transactions, and the cells are ordered from bottom to top.

In Figs. 9a and 9b, we can make the following observations:

- Both strip treemaps and Value-Cell Bar Charts visualize distributions and patterns using size and color.
- Strip treemaps use the equal-height and different-width layout, whereas Value-Cell Bar Charts use the different-height and equal-width layout (same as the regular bar charts). Space-filling layouts such as the strip treemap have the advantage of fully using the available screen space, which leads to larger areas and higher visibility for each transaction. In contrast, the equal width bars of Value-Cell Bar Charts allow

TABLE 1
Comparison of Value-Cell Bar Charts and Pixel Bar Charts

| Features | Pixel Bar Chart | Value Bar Chart |
|---|---|---|
| Bar height | Fixed[3] | Varied |
| Bar width | Varied | Fixed |
| Bar size | Shows the number of transactions (volume) | Shows the total transaction value (same as regular bar chart) |
| Representation of a Transaction | One pixel | Multiple cells or portion of a cell |
| Distribution pattern, and trends | Volume distribution seen by colored pixels | Value distribution seen by colored cells |
| Outliers | Difficult to identify a high value outlier (represented by one colored pixel) | Easy to identify a high value outlier (represented by a large colored area with multiple cells) |
| Number of transactions of a certain value | Easy to identify | Hard to identify |

[3] *Can also be varied as in the new variant shown in Fig. 7b.*

a better comparison of the aggregated transaction value as it is possible with traditional bar charts. For example, it is easier to compare the total value (height) of Germany, India, and Spain in the Value-Cell Bar Chart than comparing their width in the strip treemap.

- Value-Cell Bar Charts allow users to easily identify and compare outliers. In Fig. 9c, for example, it is hard to compare the highest value transaction for England (second partition) with that for the US (last partition). In Fig. 9e, because the Value-Cell Bar Chart uses the consistent (equal) size cells in all bars, it is easier to compare them.
- Value-Cell Bar Charts can scale to large volumes of transactions, whereas with strip treemaps, it becomes difficult to visualize thousands of transactions due to the resulting very thin slices. In Fig. 9a, we only show 375 transactions. The number of transactions can be increased by removing the white borders and using a cushion style rendering, as shown in Fig. 9c, where 1,025 transactions are shown. A corresponding value-cell bar chart is shown in Fig. 9e.

Fig. 10a shows a Cushion Treemap [13] of 40,525 sales transactions partitioned into 23 product types. The cushions improve the traditional treemaps by allowing an easier perception of the boundary between areas. The resulting visualizations work well and are visually appealing. However, it is not possible to draw cushions if the areas become too narrow as in the lower portions of the wider bars in Fig. 10a.

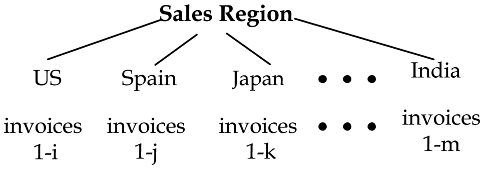


Fig. 8. A sales data structure.

In contrast, the Value-Cell Bar Chart in Fig. 10b—here shown in a variant that distinguishes negative and positive transactions—retains the advantages of regular bar charts. In addition, value-cell bar charts allow an easy correlation analysis by mapping different values to color, as shown in Fig. 10c. Note the low correlation between transaction price and discount across most product types.

### 5.2 Comparison to Squarified Treemaps

Fig. 9b illustrates a squarified treemap with borders, and Fig. 9d illustrates one without borders but with cushions. The squarified treemap uses a recursive algorithm to divide a rectangle (region) into smaller rectangles (transactions). The size and color of a square corresponds to the sales price. The comparison with Value-Cell Bar Charts shows the following:

- Both Value-Cell Bar Charts and squarified treemaps can identify transactions that contribute the most to the total value by their size. High-value transactions have larger sizes than low-value transactions.
- Squarified treemaps [14] were designed to visualize hierarchical information based on aggregated data, and this is what treemaps are most useful for. Value-Cell Bar Charts cannot show multilevel hierarchical data structures in one display. Users may only drill down to see the next levels of the hierarchy.
- Although the Value-Cell Bar Charts have a simple ordering according to the transaction value from bottom to top, the squarified treemaps shown in Figs. 9b and 9d have a 2D ordering from the lower right to upper left, which is clearly visible, but due to its 2D nature, it is more difficult to follow (for example, darker brown areas for the US are in the lower left and upper right areas of the US rectangle).
- The layout of the original treemap was not intended to be used for observing transaction-level information until recently [16]. There are a large numbers of invisible squares in Fig. 9b. After removing the partitioning lines in Fig. 9d, more small rectangles become visible. Value-Cell Bar Charts were designed for transaction-level data and, therefore, scale well to large data sets, as shown in Fig. 9f.

### 5.3 Comparison to Stacked Bar Charts

Finally, we compare the Value-Cell Bar Chart with traditional Stacked Bar Charts. Fig. 11a shows a Stacked Bar Chart of regional sales data, showing aggregated information for certain price ranges. With Value-Cell Bar Charts, we can easily achieve a similar effect by using a color mapping with few distinct colors. However, we can also use a continuous color mapping, which shows the real distribution of values, as shown in Fig. 11b, which is not possible with stacked bar charts. In addition, a direct access

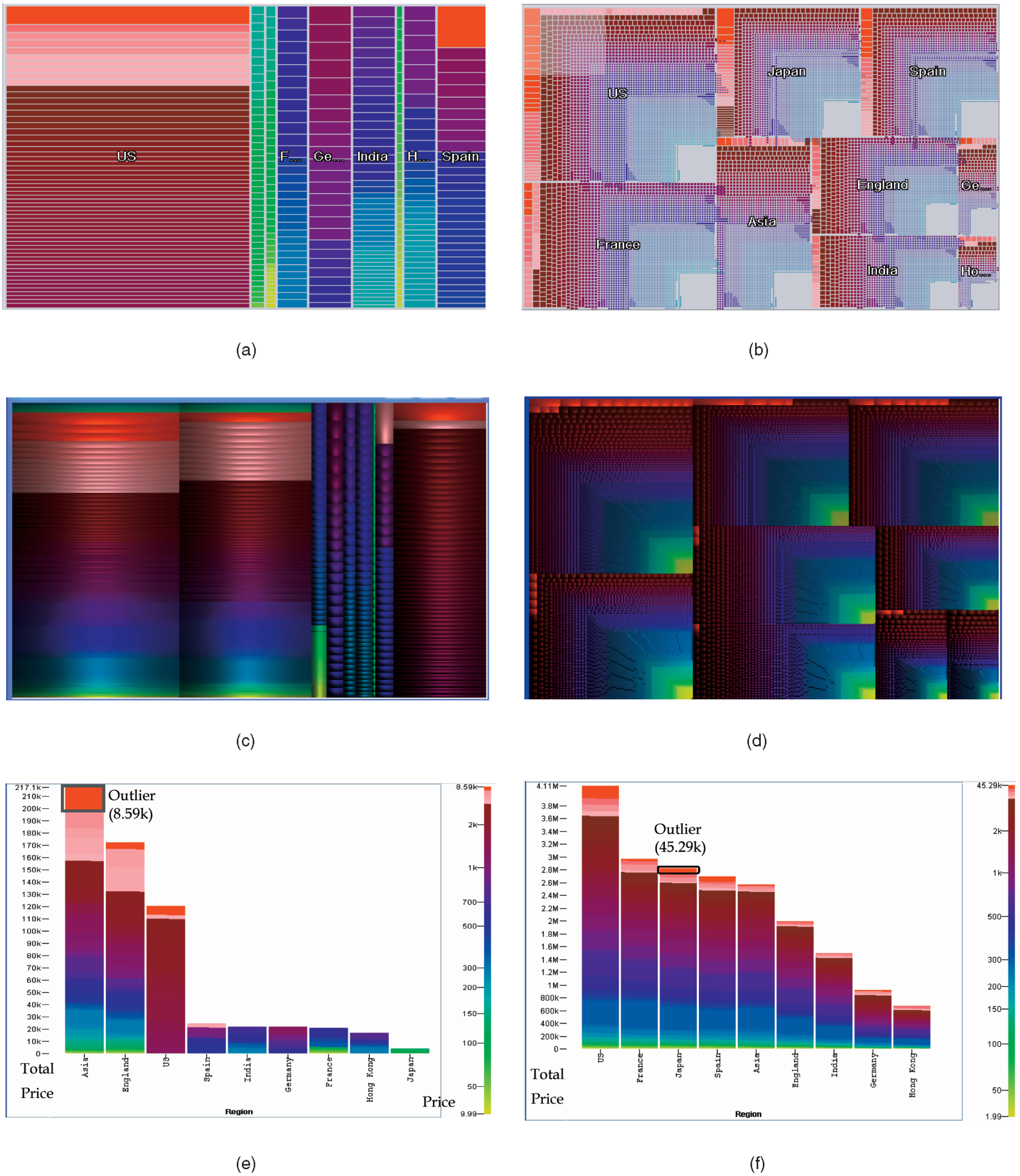
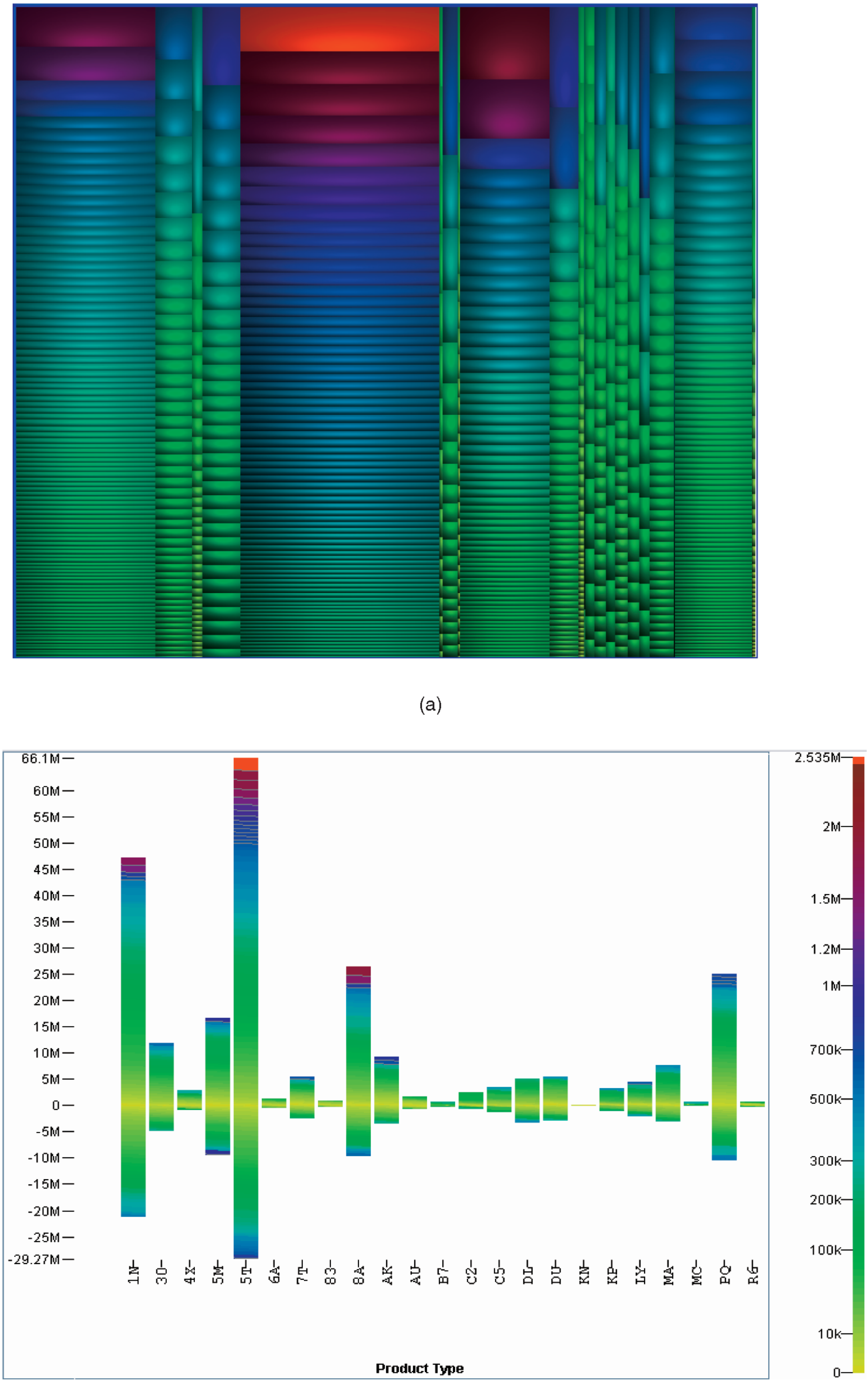


Fig. 9. Comparison with treemaps. (a) A strip treemap for regional sales transactions (first level: region, second level: transaction; *color*: *price*) (357 invoices with white borders). (b) A squarified treemap for regional sales transactions (first level: region; second level: transaction; *color*: *price*) (42,074 invoices with white borders). (c) A strip cushion treemap for regional sales transactions (first level: region, second level: transaction; *color*: *price*) (1,025 invoices without white borders). (d) A squarified cushion treemap for regional sales transactions (first level: region; second level: transaction; color: price) (42,074 invoices without white borders). (e) A value-cell bar chart for regional sales transactions ($x$-axis: region; $y$-ordering: price; color: price, 1,025 invoices). (f) A value-cell bar chart for regional sales transactions ($x$-axis: region; $y$-ordering: price; *color*: *price*, 42,074 invoices).

to the detailed transaction data is possible and additional transaction values may be mapped to color to analyze correlations. Note that the discontinuity of the color mapping used in Fig. 11b leads to a similar effect as a Stacked Bar Chart.

## 6 Conclusion

Value-Cell Bar Charts are simple and easy to use. They allow users to visually analyze large transaction data sets and capitalize on the popularity of the daily used regular bar charts. While retaining the advantages of regular bar

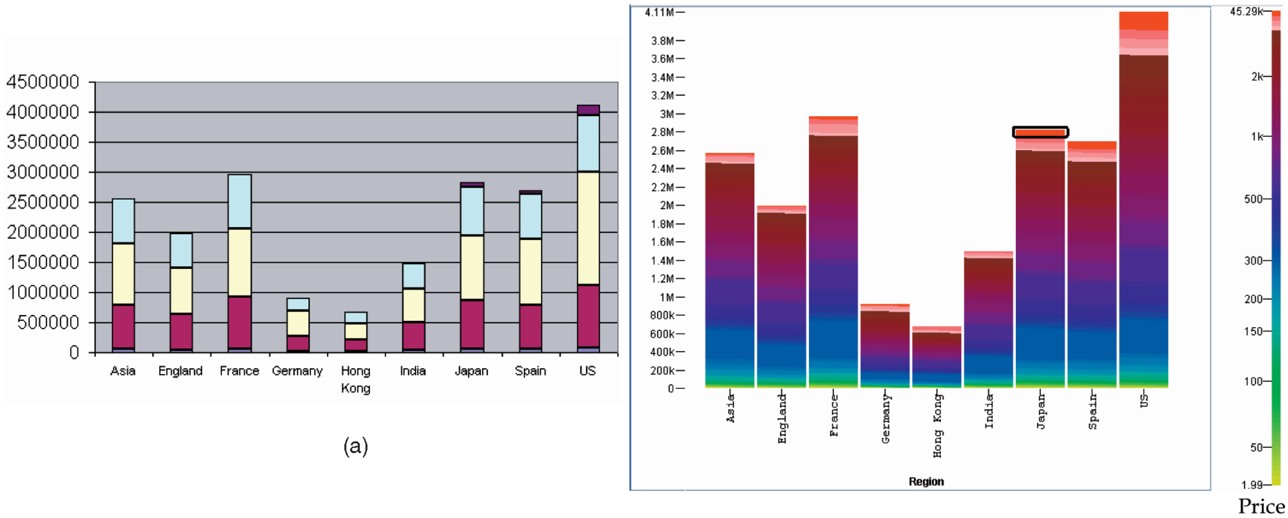


(a)

(b)

Fig. 10. (a) Product type sales analysis cushion treemap (first level: product type; second level: transaction; color: price) (40,525 invoices). (b) Product positive and negative sales analysis value-cell bar chart ($x$-axis: product type; $y$-axis ordering: *price*; color: price) (40,525 invoices). (c) Value-cell bar chart showing low correlation between transaction price and discount many high-price transactions with low discounts (that is, the highest price transaction at the top of the bar, but color yellow means low discount) ($x$-axis: product type; $y$-axis ordering: price; *color*: discount) (40,525 invoices).

charts, they help users to drill down to the transaction level and find the most important transactions in their data. The combination of the Value-Cell Bar Charts and the Pixel Bar Charts provides patterns and trends in both volume and



(a)

(b)

Fig. 11. (a) Stacked bar chart showing sales transaction aggregated for different price ranges. (b) Value-cell bar chart showing real distribution of transaction value (price) using a continuous color mapping.

value aspects. At the same time, it shows the detailed information and allows root-cause detection without requiring the user to click through many charts and listings.

Our applications using real-world sales and IT service usage data show the wide applicability and usefulness of our new technique. Future work will be in the areas of alternative solutions to the rounding and aggregation problems for optimizing cell placements. In addition, we plan to draw outlines or cushions for large transactions within a bar to better visualize their transaction boundaries.

## Acknowledgments

The authors wish to thank to Beth Keer and Mei Chun Hsu of Hewlett-Packard Laboratories for their encouragement, to Nimesh Patel of HP IT division for providing suggestions and data, and to Tobias Schreck and Dominik Morent from the University of Konstanz for their comments and suggestions.

## References

[1] M. Ankest, M. Ester, and H.-P. Kriegel, "Towards an Effective Cooperation of the User and Computer for Classification," *Proc. Sixth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD '00)*, 2000.

[2] M. Bruls, K. Huizing, and J. van Wijk, "Squarified Treemaps," *Proc. Joint Eurographics and IEEE Trans. Visualization and Computer Graphics Symp. Visualization*, 2000.

[3] B. Bederson, B. Shneiderman, and M. Wattenberg, "Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies," *ACM Trans. Graphics*, vol. 21, no. 4, 2002.

[4] S.G. Eick, "Visualizing Multi-Dimensional Data with ADVISOR/2000," *VisualInsights*, 1999.

[5] C. Stolte and P. Hanrahan, "Polaris: A System for Query, Analysis and Visualization of Multi-Dimensional Relational Databases," *Proc. IEEE Symp. Information Visualization (InfoVis '00)*, 2000.

[6] D.A. Keim, "Designing Pixel-Oriented Visualization Techniques: Theory and Applications," *IEEE Trans. Visualization and Computer Graphics*, 2000.

[7] D.A. Keim, M. Hao, J. Ladisch, M. Hsu, and U. Dayal, "Pixel Bar Charts: A New Technique for Visualizing Large Multi-Attribute Data Sets without Aggregation," *Proc. IEEE Symp. Information Visualization*, 2000.

[8] D.A. Keim and H.P. Kriegel, "VisDB: Database Exploration Using Multidimensional Visualization," *IEEE Computer Graphics and Applications*, Sept. 1994.

[9] B. Shneiderman, "Tree Visualization with Tree-Maps: 2-D Space-Filling Approach," *ACM Trans. Graphics*, vol. 11, no. 1, 1992.

[10] *SpotFire*, http//www.spotfire.com, 2006.

[11] *Tableau Software*, http//www.tableausoftware.com, 2006.

[12] M. Wattenberg, "A Note on Space-Filling Visualizations and Space-Filling Curves," *Proc. IEEE Symp. Information Visualization*, 2005.

[13] H. van de Wetering and J. van Wijk, "Cushion Treemaps: Visualization of Hierarchical Information," *Proc. IEEE Symp. Information Visualization*, 1999.

[14] B. Shneiderman, "Treemaps for Space-Constrained Visualization of Hierarchies," http://www.cs.umd.edu/hcil/treemap-history/, Dec. 2005.

[15] S.C. Eick, J.L. Steffen, and E.E. Sumner Jr., "Seesoft—A Tool for Visualizing Line Oriented Software Statistics," *IEEE Trans. Software Eng.*, vol. 18, no. 11, pp. 957-968, Nov. 1992.

[16] R. Vliegen, J.J. van Wijk, and E. van der Linden, "Visualizing Business Data with Generalized Treemaps," *IEEE Trans. Visualization and Computer Graphics*, vol. 12, no. 5, pp. 789-796, 2006.

**Daniel A. Keim** received the PhD degree in computer science from the University of Munich in 1994. He is a full professor in the Computer and Information Science Department, University of Konstanz. He has been an assistant professor in the Computer Science Department, University of Munich, and an associate professor in the Computer Science Department, Martin Luther University Halle. He also worked at AT&T Shannon Research Labs, Florham Park, New Jersey. In the field of information visualization, he developed several novel techniques, which use visualization technology for the purpose of exploring large databases. Dr. Keim has published extensively on information visualization and data mining, he has given tutorials on related issues at several large conferences, including Visualization, SIGMOD, VLDB, and KDD, he was program cochair of the IEEE Information Visualization Symposia in 1999 and 2000, the ACM SIGKDD Conference in 2002, and the Visual Analytics Symposium in 2006. Currently, he is on the editorial board of the *IEEE Transactions on Knowledge and Data Engineering*, the *Knowledge and Information System Journal*, and the *Information Visualization Journal*. He is a member of IEEE Computer Society.

**Ming C. Hao** received the MA degree in mathematics from the City University of New York. She is a senior member of the technical staff at Hewlett-Packard Research Laboratories in Palo Alto, California. Currently, she is working in the area of advance database, business management, and IT services information visualization, In addition to visualization, her field of expertise includes 3D collaboration, database, and window systems. She was a senior scientist at the IBM Almaden Research Center and T.J. Watson Research. She has been awarded more than 30 US patents and has published many papers in the areas of visualization and window systems. She is a member of the IEEE.

**Umeshwar Dayal** received the PhD degree in applied mathematics from Harvard University in 1979. He is a principal laboratory fellow in the Advanced Data Base Program at Hewlett-Packard Laboratories, Palo Alto, California, where he initiated and led research programs in data mining solutions and business process management. He has more than 20 years of research experience in data management. Prior to joining HP Labs, he was a senior researcher at DEC's Cambridge Research Lab, chief scientist at Xerox Advanced Information Technology and Computer Corporation of America, and on the faculty at the University of Texas-Austin. He has published extensively and holds several patents in the areas of database systems, transaction management, workflow systems, and data mining. He is on the editorial board of four international journals, has coedited two books, and has chaired and served on the program committees of numerous conferences. He is a member of board of the VLDB Endowment, the board of the International Foundation for Cooperative Information Systems, and the Steering Committee of the SIAM Data Mining Conference. He is a member of the IEEE Computer Society.

**Martha Lyons** received the master's degree in computer science from the University of Texas, Austin. She is currently a distinguished technologist at Hewlett-Packard Services and has more than 22 years of experience driving R&D initiatives and innovation for HP Services. She is currently responsible for managing HP Services' research agenda and relationship with HP Labs, which is focused on incubating and delivering innovative service offerings and capabilities to HP's customers. Previously, she led the development of a number of solutions, which introduced and extended HP's electronic and automated service capabilities over the Web. She has published on the topics of information retrieval for service and support, innovation, and R&D in IT services, automating services, and support processes, and currently has four patents pending.