Eurographics Conference on Visualization (EuroVis) 2012
S. Bruckner, S. Miksch, and H. Pfister
(Guest Editors)

Volume 31 (2012), Number 3

ConnectedCharts: Explicit Visualization of Relationships between Data Graphics

C. Viau[1] and M. J. McGuffin[1]

[1]École de technologie supérieure, Canada

Abstract

Multidimensional multivariate data can be visualized using many different well-known charts, such as bar charts, stacked bar charts, grouped bar charts, scatterplots, or pivot tables, or also using more advanced high-dimensional techniques such as scatterplot matrices (SPLOMs) or parallel coordinate plots (PCPs). These many techniques have different advantages, and users may wish to use several charts or data graphics to understand a dataset from different perspectives. We present ConnectedCharts, a technique for displaying relationships between multiple charts. ConnectedCharts allow for hybrid combinations of bar charts, scatterplots, and parallel coordinates, with curves drawn to show the conceptual links between charts. The charts can be thought of as coordinated views, where linking is achieved not only through interactive brushing, but also with explicitly drawn curves that connect corresponding data tuples or axes. We present a formal description of a design space of many simple charts, and also identify different kinds of connections that can be displayed between related charts. Our prototype implementation demonstrates how the connections between multiple charts can make relationships clearer and can serve to document the history of a user's analytical process, leading to potential applications in visual analytics and dashboard design.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms

1. Introduction

The use of multiple views onto the same data, or related data, is a common approach in visualization. Users are given multiple perspectives of their data, allowing them to benefit from the advantages of each view, and to compare across views. To help the user understand the relationships between views, some form of *linking* is required. Two common approaches are to use color or drawing line segments or curves to link corresponding data elements. However, if there are many data elements and all links are displayed at once, we either quickly run out of distinguishable colors, or suffer from line clutter. For this reason, linking is often only displayed in response to mouse motion: hovering over an element causes the links between that element and related ones to be displayed (either in the form of a color highlight or line or curve segments). This approach is elegant and can be applied to even large data sets, but it constrains the user to exploring relationships one element at a time, requiring the user to interact with the data, possibly for an extended period of time, to extract useful visual feedback.

We propose a new technique, called ConnectedCharts (Figures 1, 2), situated in the middle ground between the extremes of linking all elements, and only linking the element under the mouse cursor. In our work, curves are drawn between charts (data graphics), showing the correspondence between data elements (tuples) when possible, or otherwise between axes. Occlusion is avoided within each chart by "anchoring" the curves to the edges or axes of the chart. Also, rather than drawing all possible links, only those that have been created by the user (in the process of creating the charts) are displayed.

We have applied our technique to visualizing multidimensional multivariate data, such as data from a relational database table. Some very flexible visualizations of such data already exist, for example, FLINA [CvW11], which supports combinations of scatterplots and parallel coordinate plots (PCPs), and Polaris [STH02] / Tableau [MHS07],

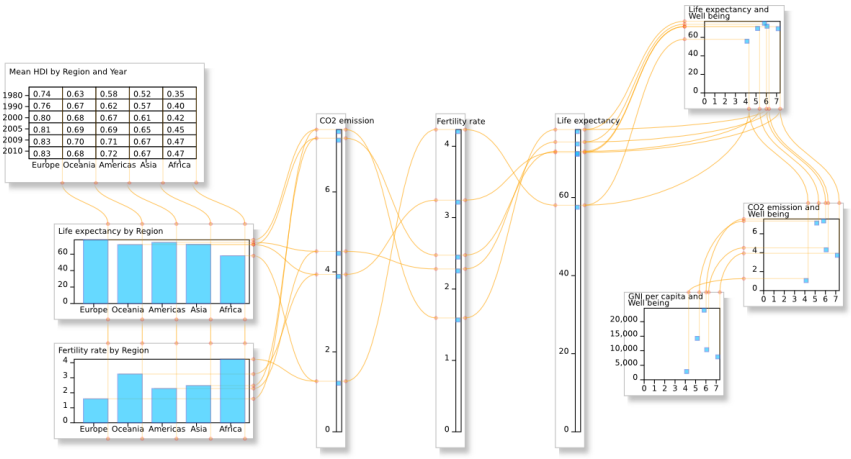

Figure 1: *A ConnectedChart showing a text table, bar charts, parallel coordinate axes, and scatterplots. (Real data set.)*

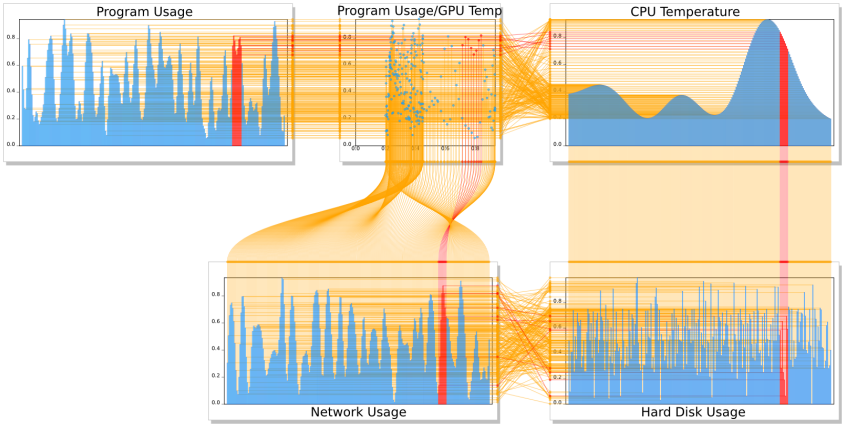

Figure 2: *A ConnectedChart with four barcharts, and a scatterplot, displaying 5 dependent variables that are functions of time. In this case, GPU temperature is a smooth, slowly varying function of time, hence the connection between the scatterplot's horizontal axis and the barchart below it exhibits an interesting, continuous variation. (Synthetic data set.)*

which can display multiple charts of the same type within a tabular grid. ConnectedCharts supports scatterplots and PCPs, but unlike FLINA, it also supports bar charts and other 2D charts, and also allows data tuples to be aggregated differently in each chart. Compared to Polaris / Tableau, ConnectedCharts offers more flexibility in that different kinds of charts can be instantiated at once, and can be positioned freely within a 2D space instead of being limited to a grid.

A user may use ConnectedCharts to explore a data set, creating new charts to answer new questions, with connections displayed to the previous charts. In such a scenario, the connections may serve to record and retrace the history of a user's analytical steps. A set of charts and their connections may also be designed and presented to an end-user, for use as a dashboard, in which case the connections elucidate the relationships between the views of the data, even when related charts are not side-by-side.

Although ConnectedCharts can result in many line segments or curves being displayed (Figure 2), the same is also true of parallel coordinate plots, where the line segments or curves serve to depict distributions, relationships, correlations, clusters, and inverse correlations. ConnectedCharts can be thought of as a generalization of parallel coordinate plots, that allow the advantages of these connections to be leveraged not just for 1-dimensional axes, but for many kinds of charts (Figure 3).

Our contributions include (1) a formal description of a design space of charts based on plots of 2D rectangles, general enough to subsume scatterplots, bar charts, Gantt charts, and variants; (2) an analysis of the types of connections that can

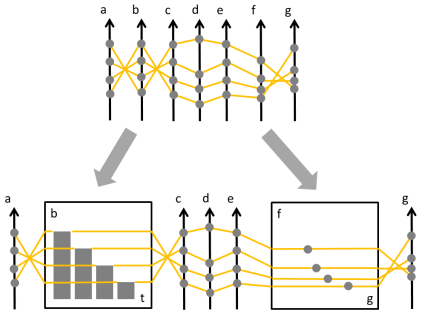

Figure 3: *One of the potential uses of ConnectedCharts is to enrich parallel coordinate displays by expanding axes interactively. Here, the b axis is converted into a barchart showing time t, and the f axis is converted into a scatterplot against g. ConnectedCharts allow charts to be intermixed with parallel coordinate axes.*

be displayed between charts; (3) a demonstration of how ConnectedCharts can be used to produce scatterplot matrices (SPLOMs), PCPs, and hybrid combinations of these and other charts; (4) a generalization of the Attribute Relation Graph of Interest (ARGOI) presented in [CvW11].

2. Background

2.1. Linking and coordination across views

A great many visualizations involve multiple views of data that are somehow *linked* to convey a relationship between the views. Buja et al. [BMMS91] give several techniques for this: linking with color (e.g., drawing corresponding elements with the same color), linking "by drawing lines connecting [corresponding] points", and linking "over time" with a "smooth animation" from one view to another. Wong and Bergeron [WB97] point out that linking can also be done by aligning axes in different views, as is done with the scatterplots in a scatterplot matrix (SPLOM) [Har75].

At least some of these linking techniques can be performed interactively. For example, it may not be feasible to draw *all* data elements in colors that distinguish the corresponding subsets of points from each other, nor may it be useful to draw all line segments between corresponding points. However, if the mouse cursor hovers over an element, the corresponding elements could then be indicated with a highlight color or line segments. An example of such interactive linking is "brushing and linking" [BC87].

Interactive linking of views is also called *coordination* [WBWK00, Rob07], which is also very common in visualizations. North [Nor00] presents a software framework for this, and distinguishes 3 types of coordination: selection ↔ selection (i.e., selection in one view causes a selection in another), selection ↔ navigation, and navigation ↔ navigation.

Of particular relevance to the current work is linking done using line segments or curves. Interactive linking across views with line segments dates back at least to work by Ted Nelson in the early 1970s [Nel99]. Line segments between views have been used for meta-visualization of the coordination of views [Wea05, TIC09]. There are also several examples of line segments and curves drawn between corresponding data elements in different views. "M and N" plots [DF80] are an early example: in a "2 and 2" plot, points in corresponding 2D scatterplots are connected to convey 4-dimensional tuples. Parallel coordinates [Ins85, Weg90] use line segments to connect tuples across multiple axes, and each axis can be thought of as a 1D "view" of the data. VisLink [CC07] is a general framework for connecting data elements across views. More recent examples of connecting elements across views include [AS07, VMCJ10]. There is also evidence that connecting elements allows a user to find elements faster than with simple highlighting [SWS*11].

With ConnectedCharts, we can make use of interactive color highlighting, but also display static curves to link together charts. These curves are distinct from previous work in a few ways: first, rather than statically display all possible linking curves between corresponding elements, we only display those that the user has established through their interactions with the charts, avoiding excessive clutter; second, the connecting curves between charts are "anchored" to the axes or edges of charts, avoiding clutter or occlusion within each chart; third, we allow for several kinds of charts to be connected, and identify several kinds of connections that can be shown (see section 5).

Note that we do not propose ConnectedCharts as a replacement to brushing and linking. Instead, we see these two approaches as having complementary advantages (Figure 4): brushing and linking is flexible and scales well to large data sets, but requires the user to invest time moving a pointing device over the data. ConnectedCharts, however, can reveal relationships at a glance prior to any interaction, and even without reading axis labels, just like how the connective lines in parallel coordinates reveal relationships between their axes.

2.2. Support for history in visualization

Recent work [HMSA08, SvW08] has presented support for navigating a user's history of their views of data. The connections in ConnectedCharts can be used to convey and retrace a user's history of analytical steps, and unlike previous work, the views and their (historical) connections are displayed in the same 2D space, rather than in separate viewports. In our prototype, the user can zoom in on a single view or zoom out to see its connections and other views. Furthermore, the user is free to edit connections, meaning that they may no longer reflect the user's true history; this can be both an advantage and a disadvantage.

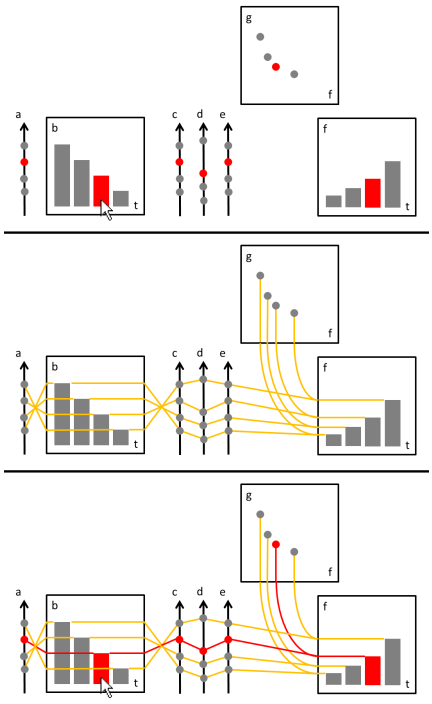

Figure 4: *Brushing and connections are complementary approaches. Top: brushing without connections. The user must interactively roll over data elements with a pointing device to discover relationships. Notice the parallel coordinate axes in the center, that can be brushed, but that do not display connections, making them less informative than usual parallel coordinates. Middle: Displaying connections between charts and between parallel coordinate axes allows the user to see common variables, correlations (e.g., between variables c, d, and e), and inverse correlations (between a and b, and between b and c) without any mouse interaction. Bottom: The advantages of both approaches combined.*

2.3. Multiple views of multidimensional multivariate data

Dimensional anchors [HGP99] and FLINA [CvW11] allow for many combinations and variants of scatterplots and parallel coordinate plots (PCPs) to be displayed of multivariate data. As will be seen, ConnectedCharts also allow for combinations of scatterplots and PCPs. However, ConnectedCharts also supports other kinds of charts (such as bar charts, stacked bar charts, and grouped bar charts) and supports aggregated data. We also generalize the notion of ARGOI presented in [CvW11].

Polaris [STH02], and its successor Tableau [MHS07], allow for multiple charts to be displayed of multidimensional multivariate data, arranged in a grid of rows and columns; the charts are implicitly linked by the alignment of their axes. As presented in the literature, the charts in the grid are always of the same type (e.g., a grid of scatterplots, or of bar charts). ConnectedCharts, in contrast, allows charts of different types to be freely positioned within a 2D space, giving the user more freedom. Nevertheless, grid-like arrangements are still permissible in ConnectedCharts (e.g., Figure 12).

Product plots [WH11] allow many kinds of charts to be generated based on a few variants of rectangles: *bars*, *spines*, *tiles* and *flucts*. The charts in ConnectedCharts are also based on plots of rectangles. ConnectedCharts does not support all the kinds of charts in product plots, but supports others (such as scatterplots) and could eventually be extended to support all product plots.

3. Data Model

Many data sets can be thought of as a table where each column is an attribute (or field) and each row is a tuple. Typically, some of the attributes are best thought of as *independent*, while the others are *dependent*. These are often referred to as dimensions and measures, respectively, in the database literature, and correspond to the domains and codomains of a function. For example, in Figure 5(left), color and petals are independent, percentage is dependent, and the data set can be thought of as a function, or mapping, from (color, petals) pairs to percentages.

Color	Petals	Percentage
Blue	4	10
Blue	5	25
Purple	4	12
Purple	5	14
Red	4	22
Red	5	17

➜

Color	Percentage
Blue	35
Purple	26
Red	39

Figure 5: *An example flower data set. Left: the raw data $d : Color \times Petals \mapsto Percentage$. Right: the data aggregated over petals, yielding $d^{[Petals]} : Color \mapsto Percentage$.*

As another example, consider a table d with 6 columns: 4 independent variables corresponding to product P, region R, year Y, and month M, and 2 dependent variables corresponding to sales S and expenses E. The table can be thought of as a mapping $d : P \times R \times Y \times M \mapsto S \times E$. Written in another way, the sales $s \in S$ and expenses $e \in E$ are a function $(s,e) = d(p,r,y,m)$ of product $p \in P$, region $r \in R$, year $y \in Y$ and month $m \in M$.

It is common to also distinguish between *categorical* attributes (also called nominal, finite, or discrete) such as a set of products, and *quantitative* attributes (also called metric, or continuous) such as real numbers. For example, Mackinlay et al. [MHS07] distinguish between quantitative dependent, quantitative independent, and categorical data fields, which they denote as Qd, Qi, and C, respectively. In our work, we note that quantitative independent variables must be discretized to a finite number of values to fit in computer memory, and therefore independent variables can always be

thought of as discrete, and therefore "categorical" in some sense. For example, time is normally thought of as quantitative, but must be discretized when used as an independent variable (e.g., reduced to a set of months or days). Thus, for simplicity, we treat all independent variables as categorical, meaning discrete or discretized. On the other hand, the dependent variables, such as sales, are quantitative in the most general case. Hence, we define $C_1,\ldots,C_M$ as the M (categorical) domains, and $Q_1,\ldots,Q_N$ as the N (quantitative) codomains in the dataset. We can then think of a dataset d as a mapping $d : C_1 \times \ldots \times C_M \mapsto Q_1 \times \ldots \times Q_N$, with $(q_1,\ldots,q_N) = d(c_1,\ldots,c_M)$.

A common operation in database systems is aggregating (also called rolling-up or projecting), which removes one of the independent variables C_i in the dataset and replaces the dependent variables with aggregations (e.g., sums, or averages) over the values of the removed variable. Figure 5(right) shows an example of this. In general, aggregating dataset d over the domain C_i can be defined as

$$d^{[C_i]}(c_1,\ldots,c_{i-1},c_{i+1},\ldots,c_M) = \sum_{c^\star \in C_i} d(c_1,\ldots,c_{i-1},c^\star,c_{i+1},\ldots,c_M)$$

Returning to the earlier example involving sales and expenses, an aggregation over both year and month would be defined as

$$(s,e) = d^{[Y,M]}(p,r) = \sum_{(y,m)\in Y\times M} d(p,r,y,m)$$

In general, any dataset d can be aggregated along any combination of its independent variables, yielding a collection of tables $d, d^{[C_1]}, d^{[C_2]}, d^{[C_1,C_2]}, \ldots, d^{[C_1,\ldots,C_M]}$. Each of these tables can be visualized using various different charts.

4. A Design Space of Charts

Before identifying different kinds of connections that can be shown between data graphics or charts, we first define the different kinds of charts to consider. Rather than consider all possible charts or visualizations, we have identified a small number of "ingredients" that are amenable to analysis, but that can be combined to yield a rich design space containing many commonly known charts, as well as a few novel kinds of charts.

We assume that each chart shows all the tuples of a data set d, which may or may not have been aggregated over some independent variables. We further assume that each tuple is represented graphically as a rectangle in the chart, where each rectangle has a position (x,y), width w, and height h. The chart maps each tuple $(c_1,\ldots,c_M,q_1,\ldots,q_N)$ to a rectangle (x,y,w,h). For example, in a bar chart, each rectangle has $x = c_i$ for some categorical variable c_i (ignoring scaling factors) and $y = 0$, as well as width $w = K$ for some constant K, and height $h = q_j$ for some quantitative variable q_j (again, ignoring scaling factors). We can therefore write that the rectangles are given by $(x,y,w,h) = (c_i,0,K,q_j)$. Converting this to a shorthand notation, we can define bar charts as charts with a mapping of the form $(C,0,K,Q)$ (Figure 6, upper left).

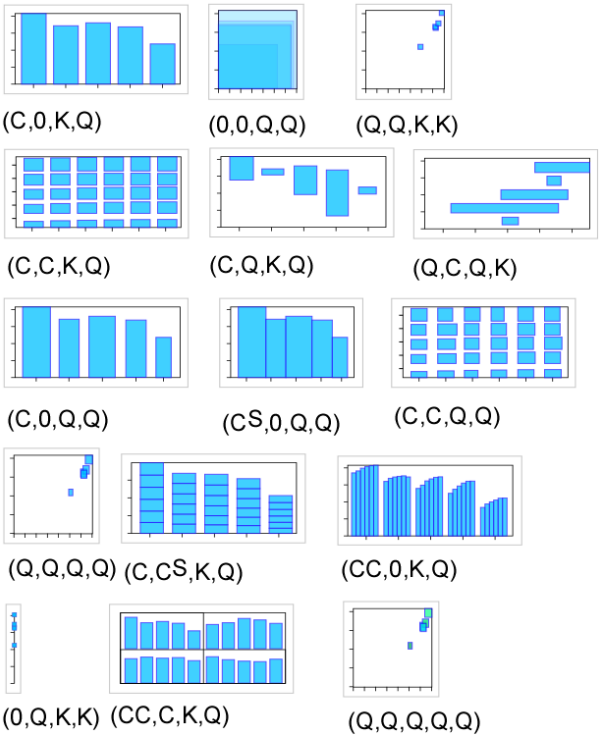

Figure 6: *Examples of charts in the design space. Well-known charts include bar charts $(C,0,K,Q)$, scatterplots (Q,Q,K,K), Gantt charts (Q,C,Q,K), stacked bar charts (C,C^S,K,Q), grouped bar charts $(CC,0,K,Q)$, parallel bar charts (CC,C,K,Q). Interesting variants include bar charts with variable width $(C,0,Q,Q)$, and scatterplots with rectangular glyphs (Q,Q,Q,Q).*

As another example, a scatterplot of two quantitative variables q_i and q_j corresponds to the mapping $(x,y,w,h) = (q_i,q_j,K,K)$, where the "points" in the scatterplot are actually small $K \times K$ squares. In shorthand, this is (Q,Q,K,K) (Figure 6, upper right).

The chart (C,C,Q,Q) (Figure 6) might be called a "table of 2D bars" with $(x,y,w,h) = (c_i,c_j,q_k,q_l)$. Note that, when we write (C,C,Q,Q), we imply that each C and each Q is distinct (i.e., has a different subscript). In contrast, a chart of the form $(x,y,w,h) = (c_i,c_i,q_j,q_j)$ redundantly encodes c_i and q_j twice, plotting squares along a diagonal. Such a redundant chart is not captured by our shorthand notation, but is not so interesting anyway.

We allow independent variables to be *nested* in the com-

putation of x or y. For example, if the categorical variables are stored as positive integers (that is, $c_i \in C_i = \{1,2,\ldots,|C_i|\}$ for all i), and we wish to plot a grouped bar chart where each group of bars corresponds to a value of C_i, and each bar within a group corresponds to a value of C_j, we might define $x = c_i + \frac{1}{|C_j|} c_j$ (ignoring margins or scaling factors). We then say that C_j is nested within C_i, and denote such a grouped bar chart as $(C_iC_j, 0, K, Q_k)$ or simply $(CC, 0, K, Q)$. Another example of a chart involving nesting is parallel bar charts (C_iC_j, C_k, K, Q_l) (see (CC, C, K, Q) in Figure 6), where C_i and C_k establish the columns and rows, respectively, of a grid, and within each cell of the grid is a $(C_j, 0, K, Q_l)$ bar chart. This nesting of variables corresponds directly to the notions of "hierarchical axis" of Mihalisin et al. [MTS91], "dimensional stacking" in LeBlanc et al. [LWW90], and also to the "cross" operator in the table algebra of Polaris [STH02].

The last ingredient in our design space allows rectangles to be *stacked* along a categorical variable C_i; we denote this stacking with C_i^S. For example, a stacked bar chart described with (C_1, C_2^S, K, Q_1) has the y of each rectangle equal to the sum of the heights of rectangles whose tuples have smaller values of c_2.

To more precisely define our design space, we note that, in our shorthand notation, each of x and y may be given by any of the following: 0 (zero), Q, C, C^S, or a sequence of one or more C followed by a final Q, C, or C^S. In addition, each of w and h may be given by K (a constant) or Q. For example, the chart $(C_1C_2Q_1, C_3C_4Q_2, Q_3, Q_4)$ would be a table of columns (C_1), sub-columns (C_2), rows (C_3) and sub-rows (C_4), within which each cell contains a (Q_1 versus Q_2) scatterplot of 2D bar glyphs encoding Q_3 and Q_4 in their width and height, respectively. In theory, it is possible to automatically enumerate the possible charts in this design space, however there are (countably) infinitely many unless some limit is imposed on the nesting depth.

Our design space could be extended in a few straightforward ways. Each rectangle, in addition to having a position, width, and height, could also be given a variable color α, yielding 5 parameters (x, y, w, h, α). A scatterplot of rectangular glyphs with variable width, height, and color is shown in the lower right corner of Figure 6. Another extension would add a text label t to each rectangle, to show a numeric value or a string. For example, the chart $(x, y, w, h, t) = (C_1, C_2, Q_1, K, Q_1)$ would be a table of horizontal bars, where the width and text label of each bar redundantly show the same quantitative value, allowing a user to quickly scan for interesting values (by looking at the bars) and then read precise values using the text labels: this kind of chart might be called a "back bar chart", since the bars appear behind the text labels.

Figure 7 shows the charts implemented in our prototype. In addition to the charts based on plotting rectangles from our design space, we added a "text table" chart to enable displaying precise numeric quantities.

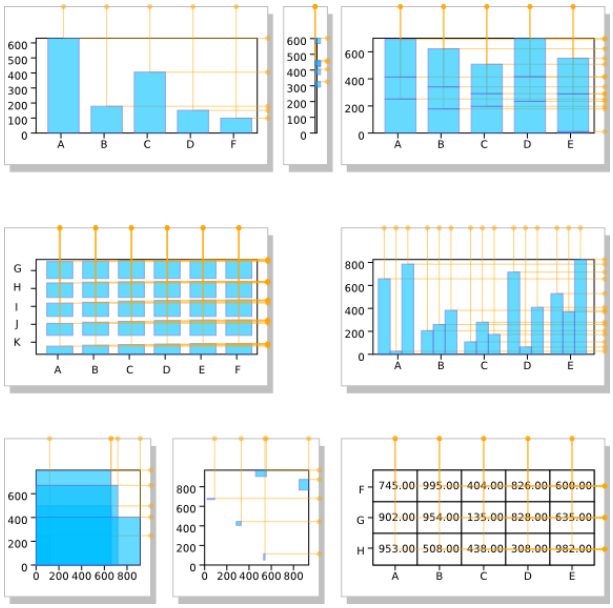

Figure 7: *Examples of the charts supported by our prototype: bar chart; 1D axis plot; stacked bar chart; table of bars; grouped bar chart; a scatterplot variant where the location of tuples, instead of being shown with points, is shown by the upper-right corner of overlapping rectangles; a scatterplot whose marks are rectangular glyphs with variable width and height; text table. Normal scatterplots are also supported. Normally, the orange lines within each chart are only drawn in the direction of neighboring connected charts; in these examples, we assume a neighboring chart is situated to the north and to the east, so the orange lines extend up and right.*

5. Types of Connections

The groundwork laid by the previous sections allows us to now analyze the types of connections that we may want to display between charts. We distinguish two types of connections: those between corresponding *tuples*, and those between corresponding *axes*.

To display connections between corresponding *tuples*, we must have the same tuples in the two charts. This is of course not the case if the data has been aggregated differently in the two charts. For example, in Figure 8, the text table and grouped bar chart show some data set $d : Region \times Year \mapsto \ldots$, whereas the charts along the top row show $d^{[Year]}$ (i.e., aggregated over the Year variable, leaving Region as the only distinguishing categorical variable), and the bar chart in the left bottom corner shows $d^{[Region]}$ (leaving Year as the only distinguishing categorical variable). The datasets d, $d^{[Year]}$,

and $d^{[Region]}$ have 30, 5, and 6 tuples, respectively, and it would not make sense to try to connect the 5 tuples in a chart showing $d^{[Year]}$ to the 6 tuples in some other chart showing $d^{[Region]}$. If, however, the data in the two charts is aggregated the same way, we can connect the tuples in one chart to the tuples in the other. There are two sub-cases to consider: connecting tuples through a Q axis, and connecting tuples through a C axis. Two examples of connections through Q are shown in the top row of Figure 8, where the Q axes are either the same, or different (notice how the connections between different Q axes is similar to the connections in parallel coordinates). To connect through a C axis, if it is the same C axis in both charts, we recommend simply connecting the values of the C axes (see Figure 8), since there are presumably multiple tuples for each value of the C axis, and connecting all tuples would only create more connective lines or curves without any benefit.

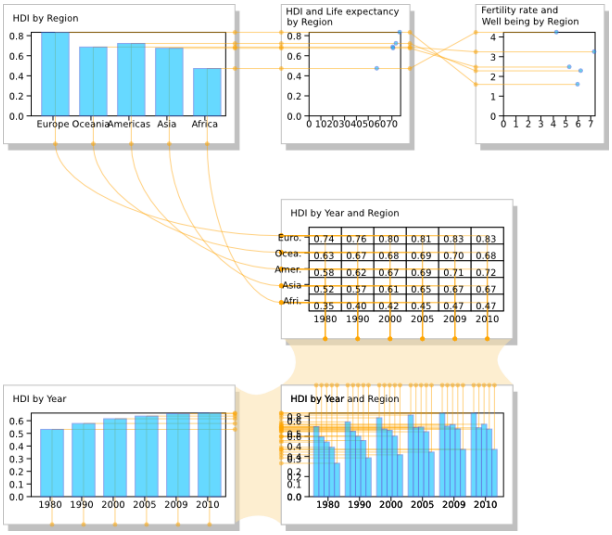

Figure 8: *Examples of different connections between charts. Top left to top center: tuples are connected across the same Q axis (Mean HDI). Top center to top right: tuples are connected across different Q axes (Mean HDI and Fertility rate). Top left to middle row: values of a C axis (Region) are connected. Middle row to bottom center: an entire C axis is connected to another C axis (Year). Bottom left to bottom center: an entire Q axis is connected to another Q axis (Mean HDI).*

If the data in the two charts is not aggregated in the same way, then it is not possible to connect tuples. However, it is still possible to connect axes. Two examples are shown in Figure 8, one for a C axis, and another for a Q axis.

Note that we intentionally do not draw connections involving a C variable nested within another C variable. We also do not draw connections involving a Q variable mapped to rectangle width or height, unless the x or y, respectively, of the rectangle is equal to zero. For example, the Q variable in a $(C,0,K,Q)$ bar chart can be connected to another Q variable in another chart (as shown in Figure 8, top left to top center), because the y of rectangles in that chart is equal to zero. However, in a scatterplot of glyphs (Q_1,Q_2,Q_3,Q_4), we do not draw connections from Q_3 or Q_4 to an axis in another chart. We did consider ways of graphically depicting such connections, but in the end decided that they would be too confusing.

6. ConnectedCharts Prototype

Our prototype is implemented in JavaScript using Data-Driven-Documents (D3) [BOH11], a graphical toolkit that allows data to be bound to graphical elements like SVG shapes. In most of our charts, the graphical "marks" are rectangles, so defining a chart with D3 is done by mapping the data to the rectangle's attributes. Smoothly animated transitions between different kinds of charts, although not currently implemented, would be straight forward to do, since the attributes of the rectangles could simply be animated from one chart to another.

Figure 7 shows most of the implemented charts. These charts are positioned within a 2D space that can be zoomed and panned. Each chart can be dragged, cloned, and deleted, and hovering over rectangles in a chart causes related elements to highlight. A popup menu (Figure 9) allows the (independent and dependent) variables of the chart to be edited.

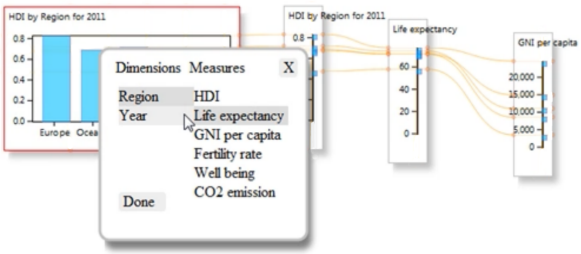

Figure 9: *A popup menu allows the variables associated with a chart to be changed, which can cause the type of chart to change.*

Given a set of independent and dependent variables associated with a chart, a small set of rules determine which mappings are appropriate for the chart. The following table lists these rules: the first column is the number of independent variables, the 2nd column is the number of dependent variables, and the third column lists the appropriate chart types. Notice that, in every case, the number of independent variables is equal to the number of Cs appearing in the mapping, and the number of dependent variables is equal to the number of Qs.

*	1	1D axis $(0,Q,K,K)$
1	1	bar chart $(C,0,K,Q)$
*	2	scatterplot (Q,Q,K,K) overlapping rectangles $(0,0,Q,Q)$
1	2	bar chart whose bars have variable width and height $(C,0,Q,Q)$
*	3	scatterplot of glyphs with variable height (Q,Q,K,Q)
*	4	scatterplot of glyphs with variable width and height (Q,Q,Q,Q)
2	1	stacked bar chart (C,C^S,K,Q) grouped bar chart $(CC,0,K,Q)$ table of bars (C,C,K,Q) text table $(x,y,t) = (C,C,Q)$

As an example, if there is 1 independent variable and 2 dependent variables associated with the chart, then the rules for (*,2) and (1,2) would apply, allowing for a scatterplot, overlapping rectangles, or a bar chart whose bars have variable width and height. Selecting the chart and hitting the spacebar allows the user to cycle through these 3 chart types.

A typical workflow could start with a single chart, which could be cloned with a shift-drag. Then, in the cloned chart, the user could choose new variables for each axis with the right-click menu, switch chart types with the spacebar, then connect the charts with a drag and drop from one chart's axis to the other.

6.1. Data

The dataset shown in Figures 1, 8, 10, 11, and 12 is based on the Human Development Report by the United Nations Development Programme (UNDP) (`http://hdr.undp.org/en/reports/global/hdr2011/download/`). In this dataset, countries were grouped by Region to form a first independent variable (C), and Year was used as a second independent variable. Dependent variables (Q) include HDI (Human Development Index), GNI (Gross National Income) per capita, Life expectancy, and Fertility rate.

6.2. Examples

Figure 10 illustrates one way that ConnectedCharts can be used to successively analyze a dataset, and how the connections reveal the history of the user's analytical process. The configuration in Figure 10 starts with a text table of Human Development Index (HDI) for each region over a number of years. The table of bars makes it easier to perceive some overall patterns. This table is then "split", so to speak, into two stacked bar charts, one for each combination of one categorical variable "slicing" the other. Finally, ordinary bar charts are produced (by aggregating), to emphasize the totals, for easy comparison and detection of trends. Each step of the process is recorded in the form of the connections, making it easy to follow the same exploratory path.

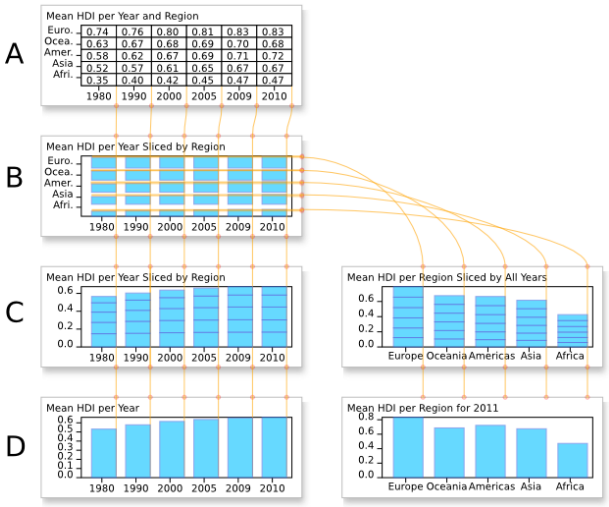

Figure 10: *A: HDI for each year and region. B: The same data, shown with parallel bars instead of numbers, to enable quick, visual inspection. C: The same bars stacked along years, and along regions. D: Equivalent bar charts, aggregated by year, and by region.*

Figure 11 illustrates another analytical process. Finally, Figure 12 shows that entire PCPs and SPLOMs can be instantiated, by appropriately connecting together the relevant charts. Furthermore, many variations and mixtures of these can be created, incorporating bar charts and their variants, as the user sees fit.

7. Generalizing ARGOIs

Claessen and van Wijk [CvW11] introduced the notion of an Attribute Relation Graph of Interest (ARGOI), where each node is a variable, and each edge is a pair of variables (each edge corresponding to a scatterplot or pair of parallel coordinate axes). Our ConnectedCharts can be modelled by a graph analogous to the ARGOI, but in our case the graph is a hypergraph (i.e., a graph with hyperedges, each of which can be incident on many nodes), since in our case each chart would be a hyperedge, and each chart may display multiple variables. We call this graph the hyper-ARGOI.

The dual of the hyper-ARGOI would also be useful for modeling ConnectedCharts: in this case, each node would be a chart, and each hyperedge would be a variable that is possibly displayed in many charts.

Finally, a third way to model a ConnectedCharts visualization would be with a bipartite graph: one set of nodes for variables, another set of nodes for charts, and edges between a variable and a chart when the former appears in the latter. Notice that all three structures, the hyper-ARGOI, its dual, and the bipartite graph, encode the same information and are isomorphic.

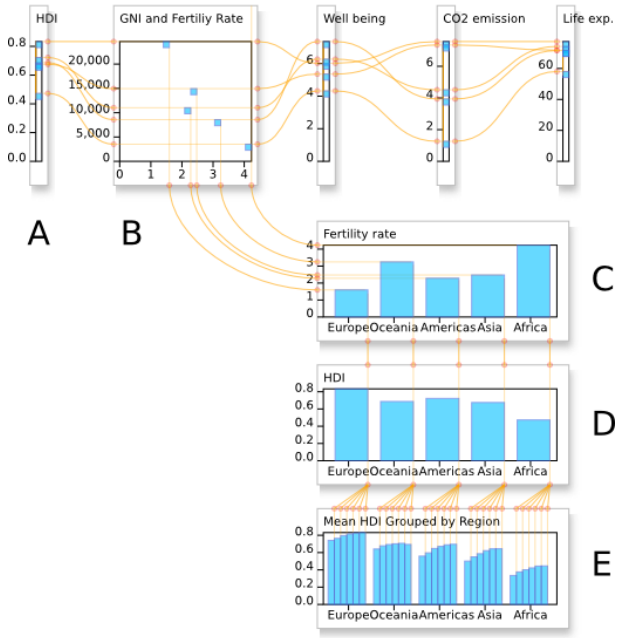

Figure 11: *The top row of charts show a parallel-coordinates-style comparison of variables. Connections between A and B show HDI versus GNI. B: a scatterplot of GNI (vertically) and fertility rate (horizontally) reveals a tuple with low GNI and high fertility. C: connections between B and C reveal the tuple to be Africa, which has the highest fertility rate. D: Africa also has the lowest HDI. E: In fact, Africa's HDI has been the lowest over all years.*

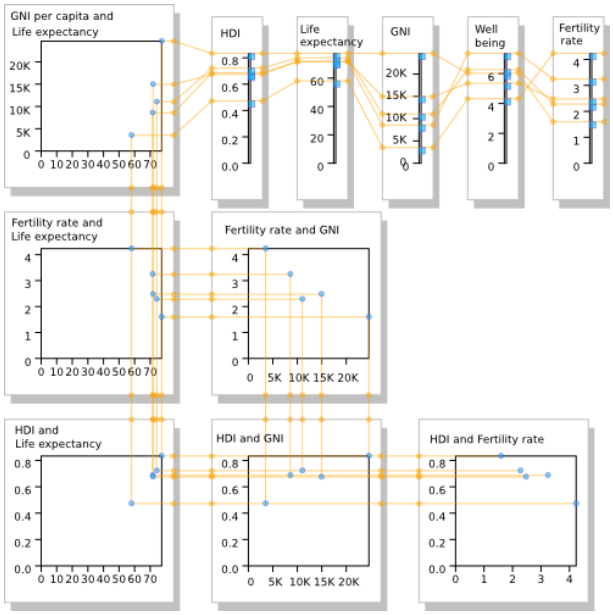

Figure 12: *Both a scatterplot matrix (SPLOM) and parallel coordinates plot (PCP) can be instantiated as a ConnectedChart, as can a hybrid mixture of them, shown here.*

Note that none of these graphs correspond directly to the connections actually *displayed* by ConnectedCharts, since ConnectedCharts only displays the connections established by the user through interaction, ensuring that excessive clutter can be avoided.

8. Conclusions and Future Directions

We have presented a technique for linking together charts using line segments and curves that can be applied to many different kinds of charts, including scatterplots, bar charts, and variants. We have also shown how multiple charts may be assembled to create parallel coordinate plots, scatterplot matrices, and mixtures of these with each other or other types of charts. ConnectedCharts allows for tuples in different charts to be aggregated to different levels, and for charts to positioned freely in a 2D space. Because the ConnectedCharts technique does not require that all possible connections be displayed, the connections that are displayed can be a compromise between showing all possible links and showing only links in response to brushing.

We have also presented a formal description of a design space of charts based on plotting rectangles, and shown how this design space encompasses many useful kinds of charts, including scatterplots of rectangular glyphs and Gantt charts. We have also distinguished different kinds of connections (section 5) that may be displayed between charts, and pointed out that Claessen and van Wijk's [CvW11] notion of ARGOI can be generalized to a hyper-ARGOI.

For future work, we are interested in implementing improvements to the user interface of our prototype. For example, a popup menu based on the FlowVizMenu [VMCJ10] might allow for a gestural-style of interaction with the charts, to clone and modify them. Macros might also be implemented to allow the user to quickly instantiate entire PCPs or SPLOMs without having to construct them one chart at a time. Techniques for enhancing parallel coordinates, or reducing clutter, such as edge bundling, might be used to similarly enhance ConnectedCharts.

A more ambitious project would be to automatically instantiate a set of ConnectedCharts, based on some arbitrary dataset that has been read in by the system: which charts would be most informative to the user, and with which connections between them?

9. Acknowledgments

Christophe Viau's Ph.D. studies were supported by an ARC Fellowship from SAP and by NSERC.

References

[AS07] ARIS A., SHNEIDERMAN B.: Designing semantic substrates for visual network exploration. *Information Visualization 6*, 4 (2007), 281–300. 3

[BC87] BECKER R. A., CLEVELAND W. S.: Brushing scatterplots. *Technometrics 29*, 2 (1987), 127–142. (Reprinted in Dynamic Graphics for Statistics, edited by William S. Cleveland and Marylyn E. McGill, Wadsworth 1988). 3

[BMMS91] BUJA A., MCDONALD J. A., MICHALAK J., STUETZLE W.: Interactive data visualization using focusing and linking. In *Proceedings of IEEE Visualization (VIS)* (1991), pp. 156–163, 419. 3

[BOH11] BOSTOCK M., OGIEVETSKY V., HEER J.: D3: Data-driven documents. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 17*, 12 (2011), 2301–2309. 7

[CC07] COLLINS C., CARPENDALE S.: VisLink: Revealing relationships amongst visualizations. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 13*, 6 (2007), 1192–1199. 3

[CvW11] CLAESSEN J. H. T., VAN WIJK J. J.: Flexible linked axes for multivariate data visualization. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 17*, 12 (2011), 2310–2316. 1, 3, 4, 8, 9

[DF80] DIACONIS P., FRIEDMAN J. H.: M and N plots, 1980. 3

[Har75] HARTIGAN J. A.: Printer graphics for clustering. *Journal of Statistical Computation and Simulation 4*, 3 (1975), 187–213. 3

[HGP99] HOFFMAN P., GRINSTEIN G., PINKNEY D.: Dimensional anchors: A graphic primitive for multidimensional multivariate information visualizations. In *Proceedings of workshop on New Paradigms in Information Visualization and Manipulation (NPIVM)* (1999), ACM, pp. 9–16. 4

[HMSA08] HEER J., MACKINLAY J. D., STOLTE C., AGRAWALA M.: Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 14*, 6 (2008), 1189–1196. 3

[Ins85] INSELBERG A.: The plane with parallel coordinates. *Visual Computer 1* (1985), 69–91. 3

[LWW90] LEBLANC J., WARD M. O., WITTELS N.: Exploring N-dimensional databases. In *Proceedings of IEEE Visualization (VIS)* (1990), pp. 230–237. 6

[MHS07] MACKINLAY J. D., HANRAHAN P., STOLTE C.: Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 13*, 6 (2007), 1137–1144. 1, 4

[MTS91] MIHALISIN T., TIMLIN J., SCHWEGLER J.: Visualization and analysis of multi-variate data: A technique for all fields. In *Proceedings of IEEE Visualization (VIS)* (1991), pp. 171–178. 6

[Nel99] NELSON T. H.: Xanalogical structure, needed now more than ever: parallel documents, deep links to content, deep versioning, and deep re-use. *ACM Computing Surveys 31*, 4es (1999), 33. 3

[Nor00] NORTH C. L.: *A User Interface for Coordinating Visualizations based on Relational Schemata: Snap-Together Visualization*. PhD thesis, Department of Computer Science, Department, University of Maryland, College Park, Maryland, 2000. 3

[Rob07] ROBERTS J. C.: State of the art: Coordinated & multiple views in exploratory visualization. In *Coordinated and Multiple Views in Exploratory Visualization (CMV)* (2007), pp. 61–71. 3

[STH02] STOLTE C., TANG D., HANRAHAN P.: Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 8*, 1 (2002), 52–65. 1, 4, 6

[SvW08] SHRINIVASAN Y. B., VAN WIJK J. J.: Supporting the analytical reasoning process in information visualization. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI)* (2008), pp. 1237–1246. 3

[SWS*11] STEINBERGER M., WALDNER M., STREIT M., LEX A., SCHMALSTIEG D.: Context-preserving visual links. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 17*, 12 (2011), 2249–2258. 3

[TIC09] TOBIASZ M., ISENBERG P., CARPENDALE S.: Lark: Coordinating co-located collaboration with information visualization. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 15*, 6 (2009), 1065–1072. 3

[VMCJ10] VIAU C., MCGUFFIN M. J., CHIRICOTA Y., JURISICA I.: The FlowVizMenu and parallel scatterplot matrix: Hybrid multidimensional visualizations for network exploration. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 16*, 6 (2010), 1100–1108. 3, 9

[WB97] WONG P. C., BERGERON R. D.: 30 years of multidimensional multivariate visualization, 1997. Chapter 1 (pp. 3–33) of Gregory M. Nielson, Hans Hagen, and Heinrich Müller, editors, Scientific Visualization: Overviews, Methodologies, and Techniques, IEEE Computer Society. 3

[WBWK00] WANG BALDONADO M. Q., WOODRUFF A., KUCHINSKY A.: Guidelines for using multiple views in information visualization. In *Proceedings of Advanced Visual Interfaces (AVI)* (2000), pp. 110–119. 3

[Wea05] WEAVER C.: Visualizing coordination in situ. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis)* (2005), pp. 165–172. 3

[Weg90] WEGMAN E. J.: Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association 85*, 411 (1990), 664–675. 3

[WH11] WICKHAM H., HOFMANN H.: Product plots. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 17*, 12 (2011), 2223–2230. 4