Dynamic Aggregation with Circular Visual Designs

Mei C. Chuah
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-2145
mei+@cs.cmu.edu
http://www.cs.cmu.edu/~mei

Abstract

One very effective method for managing large data sets is aggregation or binning. In this paper we consider two aggregation methods that are tightly coupled with interactive manipulation and the visual representation of the data. Through this integration we hope to provide effective support for the aggregation process, specifically by enabling: 1) automatic aggregation, 2) continuous change and control of the aggregation level, 3) spatially based aggregates, 4) context maintenance across different aggregate levels, and 5) feedback on the level of aggregation.

1 Introduction

New technologies have made it much easier for us to collect and disseminate information. As the information available to us grows, so does the need for more powerful tools to manipulate this information. A very effective method for dealing with large data sets is aggregation. Data aggregation simplifies large data sets by summarizing groups of data elements and representing such groups with a single graphical symbol or glyph.

In this paper we explore two visual representations that are tightly coupled with the aggregation technique. We believe that the close marriage between the interactive display and the aggregation method will improve the effectiveness of the summarization process. Specifically, our interactive displays enhance the usability of aggregation in five primary ways:

1. Automatic aggregation of elements: Our dynamic visualizations automatically aggregate and de-aggregate the data as necessary so that the greatest possible level of data detail is shown. Aggregation is performed in two cases: a) if an object is occluded, or b) if an object is too small to be perceived. Objects that fall into either of these two categories are aggregated with neighboring objects. Some systems allow users to choose the elements for aggregation either through enumeration or through a data query; while such techniques are more flexible, they also tend to require the user to perform significant manipulation of the visualization elements.

2. Interactive control of the aggregation level: One of the primary difficulties of data aggregation is picking the best aggregate level at which to view the data set. Meaningful patterns can often be arrived at only after repeated cycles of aggregating and visualizing the data. The visualizations presented in this paper support interactive, continuous aggregation control so that users may rapidly perform the “aggregate and visualize” cycle. In addition, changes in the level of aggregation occur continuously, providing better context on data pattern changes across different aggregate levels.

3. Aggregation of spatial neighbors: Our visual displays are designed so that neighboring data values are mapped to spatially contiguous graphical elements. When there are too many elements to fit within a given space, the elements are aggregated with their closest spatial neighbor. By limiting aggregation to spatially proximal objects, it will be much easier to deduce the contents of an aggregate and subsequently interpret it.

4. Context maintenance across aggregation levels: When the level of aggregation changes, its visual representation must naturally also change. Such changes can sometimes make it difficult to maintain visual context from one level of aggregation to the next. Our interactive displays are crafted so that context is naturally maintained across aggregate levels. This is achieved by encoding the data with a graphical property that remains constant across aggregate levels.

5. Feedback on level of aggregation: It is important to give feedback on the degree of aggregation so that users do not erroneously compare or draw conclusions between value sets that are aggregated at different levels. This becomes especially important when a value set is divided into multiple subsets, each with its own aggregation level, as is commonly done to focus on particular sections of the data while maintaining context of the surrounding elements.

In this paper we combine circular visual displays with dynamic aggregation to provide easy access to large data sets at different levels of detail. There are other techniques that allow users to interactively control the level of detail including PAD++[2], distortion techniques[4,8], and Table Lens[11]. Our approach is different from these previous approaches in terms of the visual displays used and the interactive metaphors employed to control the different detail levels.

Subsequently we present two interactive visualizations for dynamically manipulating aggregates. One display is a circular histogram for showing non-hierarchical data and the other is an aggregate tree-map for showing hierarchical data. We will also show how both these techniques can be combined to form a visual display containing aggregates within aggregates. Such a display is especially useful for relating attributes of objects to the hierarchical structure in which the objects belong. We will conclude by discussing some limitations of these techniques as well as how the techniques can be combined with other interactive visualization metaphors.

2 SolarPlot (Circular Histogram)

In this visualization, values are plotted around the circumference of a circle. For example, Figure 1 shows ticket sales data for a hypothetical movie theatre over a period of 30 years. In this case the circumference encodes the time in which a particular ticket was sold (i.e. the *date-sold* attribute of the *ticket* object). Data is encoded from 0 degrees (which is at the mid-bottom point of the circle) to 360 degrees.

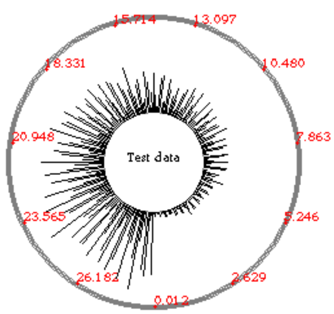

Figure 1: Highly aggregated ticket sales data

1. Automatic aggregation

The level of detail with which we are able to show the data values in this display is automatically determined by the size of the circle circumference. A smaller circle will most likely have many different *date sold* values fall on the same circumference pixel. When more than one value falls on the same pixel, we plot a higher spoke emanating from that pixel. For example in Figure 1, more tickets are sold in the later dates than in the earlier dates. A larger circular plot of the same data (e.g. Figure 2) usually results in shorter spokes. This is because when more space is available, there is a smaller probability that values will fall on the same pixel of the circle perimeter. Because the heights of the spokes encode the number of tickets sold at a particular date, the solarPlot visualization used in this example resembles a circular frequency distribution (or a circular histogram).

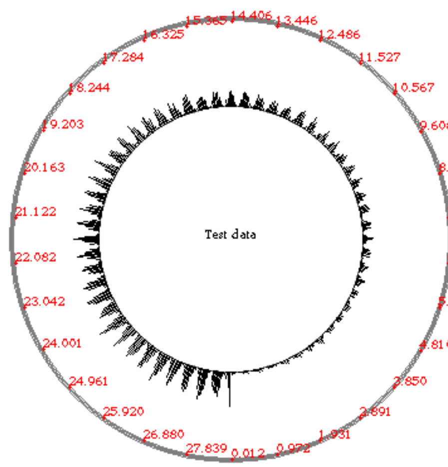

Figure 2: Less aggregated ticket sales data

2. Interactive control of the aggregation level

Interactive control is provided so that the circle may be continuously expanded or contracted. By interactively expanding and contracting the solarPlot we can view different levels of aggregation. Different aggregation levels may reveal different patterns within the data set. From the highly aggregated display in Figure 1 we notice that the sales distribution is increasing with time. The less aggregated display (Figure 2), however, shows that there is a periodic cyclic trend, peaking out during the summer and winter seasons of each year. The outer gray circle around the histogram is used to interactively scale the absolute heights of the histogram bars.

3. Aggregation of spatial neighbors

Aggregation is performed on close spatial neighbors. This is so that users may easily determine the elements within an aggregate group. In addition adjacent values will probably be related in the data set so aggregating them will hopefully still produce useful data trends. For this reason, we believe this representation is most appropriate for encoding ordered values where the positions of the elements on the circumference of the solarPlot have information significance.

4. Context maintenance across aggregation levels

In the solarPlot, data values are mapped to different angles on the circle. This directional graphical property remains the same throughout all the different aggregate levels. For example in Figure 1, the 15th year is represented at 180 degrees. Similarly in Figure 2, which shows a different aggregation level of the same data set as Figure 1, the 15th year is still represented at 180 degrees. Thus the degrees per value stays constant no matter the level of detail.

5. Feedback on level of aggregation

The circular representation shows the level of aggregation by the size of the circle (or the radius of the sector). A large radius indicates a lower aggregate level (because there is more perimeter space) and a smaller circle indicates a higher level of aggregation. It is important to note that we cannot compare the heights of elements that are aggregated at different levels because more highly aggregated elements will naturally have a higher frequency distribution and vice versa. Feedback on the aggregation level is therefore extremely important so that users do not erroneously compare differently aggregated objects.

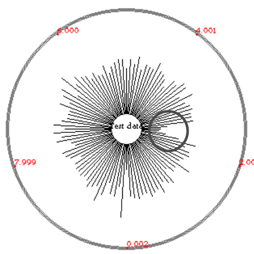

Figure 3: Sales data for hypothetical company

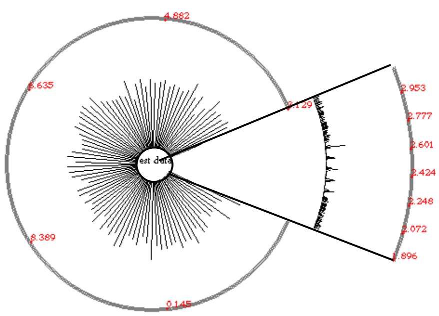

Figure 4: Same data as Figure 3 but with a focus interval

We can also focus on particular attribute value intervals and expand the level of detail for only those intervals. This can be done by marking the sector that we want to control separately and then expanding it independently of the other parts of the circle. For example, Figure 3 shows sales data for a hypothetical company over time. From the highly aggregated data we can see that in general the company sales have been rather flat or constant with time. There is, however, an area of lower sales shown in Figure 3. By marking that time period and expanding it (Figure 4) we can see that it contains five spikes. As it happens, these spikes indicate periods of great advertising. Based on the distribution we may surmise that although advertising increases the sales for a small window of time, it is followed by periods of inactivity (low sales) right after the peaks. This resulted in lower overall sales in that period compared to other periods. In conclusion, the advertising campaign was not too successful. By focussing solely on the problematic time interval we are able to diagnose the problem without having to expand the entire display (which would require significantly more screen space).

A more common representation for frequency distributions is a two dimensional histogram (*rectilinear histogram*), like the one shown in Figure 5. In this representation, the *x-axis* encodes the attribute values and the *y-axis* encodes the frequency of particular attribute values. The circular representation, however, has several advantages for interactively manipulating and controlling aggregates:

1. Greater maximum level of detail

Here we define the maximum level of detail to be the highest level of data detail that may be viewed on a visualization within a CRT screen area. In the solarPlot representation, the maximum level of detail is limited by the maximum circle circumference. Assuming that the CRT screen has a resolution of 1280 by 1024 pixels, and the data set size is 10,000 elements, we can have up to 3199 bins within the CRT screen with 3 elements per bin (assuming a flat distribution). On the other hand, the *rectilinear histogram* can only have a maximum of 1280 bins per CRT screen. The 3199 bins estimate is admittedly an optimistic estimate because we are assuming a flat distribution. However, even if we increased the size of each bin from 3 to 300, we would still be able to show 1332 bins.

On the downside, the solarPlot representation leaves a great area within its inner circle empty. This does not pose a problem when we can position some plots within the inner area of other plots. However, problems arise when we want to view several plots simultaneously at the same level of aggregation. In this case we would not be able to embed plots within plots because all of them would have the same size (i.e. same aggregate level). Thus all the inner areas would be wasted.

2. Better label layouts

In the circular visualization display, data labels may be placed above or below and slightly to the left or right of neighboring labels. In this representation, the maximum number of labels that can be shown is based on the *height* of the label, which is a constant. Horizontal labeling for the rectilinear histogram, in contrast, is based on the *width* of the label, which is often varying in length and is usually larger than the label height. We estimate the solarPlot representation can display approximately *2n* more labels than the rectilinear histogram (where *n* represents the number of characters in the longest label).

Another possibility is to use vertical labels in the rectilinear histogram. A problem with vertical labels, however, is that a significant amount of white-space must be allocated between neighboring labels so that users are not tempted to read the labels horizontally, which is the more common way to read text. As for rotated labels (shown in Figure 5) we still require fairly large inter-label spacing to accommodate for the text rotation. In general both these methods still allow for many fewer labels than the solarPlot.

3. Better feedback on level of aggregation

As mentioned earlier, the solarPlot clearly shows the level of aggregation through the size of the circle. The aggregation level is less clear in the rectilinear histogram, especially when we are aggregating different value intervals at different levels. For example Figure 5 shows the same data as Figure 4 with the same two levels of aggregation. In Figure 4 it is clear which time intervals are more highly aggregated and which are not. In Figure 5, however, it is unclear whether the circled area is lower due to the distribution of the data set or due to different aggregate levels. The only way to determine for certain is to look at the *x*-axis labels, which in this case increase by approximately 3.6 in the circled area and by 7.7 in the other areas.

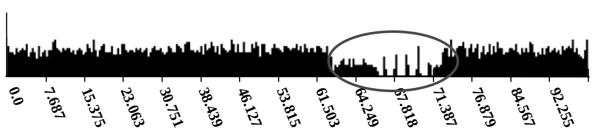

Figure 5: Rectilinear histogram with two different levels of aggregation

It is possible to make aggregated areas on the rectilinear histogram more apparent by using different tick spacings on the *x*-axis depending on the level of aggregation. However, this method is still less visually salient than the size encoding used in the solarPlot. We could also map the level of aggregation to other visual properties. For example we could indicate the level of aggregation by saturating the background of the display differently according to the aggregation level. Another alternative is to encode level of aggregation by using the depth of the bars (i.e. making highly aggregated portions jut out more and less aggregated portions sink into the display). However, these additional graphical mappings may introduce added complexity as well as clutter the display by introducing more ink density.

4. Maintain context across aggregate levels

As we already described, the angle-to-data encoding for the solarPlot remains constant across different aggregate levels. On the other hand, the rectilinear histogram plot maps data values to *x-position*. Unlike the solarPlot angles, the *x-position* for a particular value changes depending on the aggregate level. A higher aggregation level will cause *x-position* values to shift to the left and a lower aggregation level will cause *x-position* values to shift to the right.

An interesting alternative is to integrate the rectilinear histogram with a perspective wall[9]. A hand drawn example of this technique is shown in Figure 6. The perspective wall gives good feedback on the level of aggregation (i.e. the flat wall has the lowest aggregation level and the side walls increase in aggregation level as they taper to the back). The perspective histogram also provides better perceptual context because values on the *x-axis* no longer shift horizontally across different levels of aggregation. The main weakness of this display is that the perspective distortion causes the entire data set to be aggregated differently so that it is no longer possible to compare the height distributions along the side walls. In addition, we can only have a single, spatially continuous focal point. It is not possible to focus on and compare two non-continuous regions.

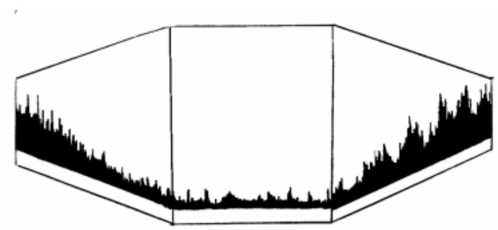

Figure 6: Perspective Histogram

The solarPlot is most useful for viewing global relationships or trends in large data sets for a single data attribute or between two data attributes. It is less appropriate for viewing relationships between multiple attributes. For this purpose we have linked different solarPlots together through *painting*[1] so that attribute relationships can be explored across plots. Nevertheless, a series of aligned rectilinear histograms is probably more effective for exploring relationships across multiple attributes, as was done in the Table Lens system[11].

Aligning a set of solarPlots would correspond to nesting one within the other, generating a series of concentric plots (similar to Figure 9). Unlike rectilinear alignment however, the series of concentric circles has an implicit importance ordering, with the outer circles being the most important because it is shown at the highest level of detail. The innermost circle is deemed least important and is highly aggregated. Such a representation is appropriate for data sets that have an importance ordering corresponding to the ordering implicit within the concentric graphical representation. An example is time-based data, e.g. stock price data, where the later time values are of more interest than the earlier time values. By using the concentric circles representation we can compare more recent time trends with older trends for similarities and differences. At the same time more detail is shown on the newer data values.

The solarPlot system can be extended to support concentric alignment and to allow aggregation to occur not just within each circle plot but also between circles. When too little space is allocated between two concentric circles we can combine them by aggregating together the values of the two circles. It is also interesting to note that the solarPlot display can be extended to encode two attributes instead of just one as in the case of Figure 1, Figure 2, Figure 3, and Figure 4. I.e. the solarPlot can be used as a bar chart instead of as a histogram. This can be achieved by encoding the second attribute using spoke lengths and using color to encode the level of aggregation. The spoke length of an aggregate object would then be the average value of all the elements it contains. In this extended system aggregation would still occur in the same way (i.e. when two values fall on the same circumference pixel) and the interactive controls can also remain the same.

3 Aggregate TreeMap

The aggregate tree is a spatially contiguous method for displaying hierarchies. This visual display is similar to the Polar TreeMap visualization presented by Johnson[6]. In this visualization we represent a hierarchy as a triangle. Each node fills a space within that triangle. The space allotted to a particular node is based upon its total number of descendants in the hierarchy. Subtrees that have greater numbers of nodes will be allotted more space than those with fewer nodes.

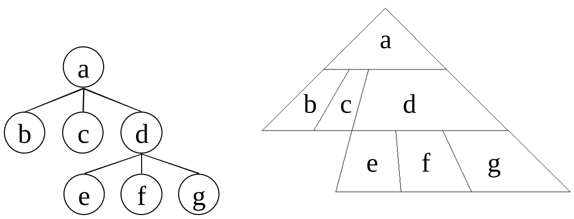

Figure 7: Node-link Tree Representation and its corresponding Triangular Aggregate Tree

The triangle is vertically divided into as many levels as are in the hierarchy. For example Figure 7-right shows an aggregate tree of a 3 level hierarchy (shown to the left). The topmost level is given to the root of the tree. The second level is divided among the root's children based on their total number of descendants. Child *d* gets two-thirds (four-sixths) of the space because there are 4 nodes within that subtree while there is only one node in the *b* and *c* subtrees.

Figure 8: Home directory with last modification date encoded by color saturation.

To better utilize the display space, we can use a partial circular representation instead of a triangular representation. The circular representation provides a greater angle range at the apex of the tree and this trickles down to the nodes on the lower levels of the tree. For example Figure 8 (shown in color in Plate 1) shows a circular aggregate tree representation of the author's home directory. Note that node color may be used to encode an attribute of the hierarchy nodes (e.g. *file-permissions*, *file-size*, *modification-date*). The color saturation of the nodes in Figure 8 are used to represent *date-of-last-modification.* A triangular aggregate tree of the same hierarchy as Figure 8 (Plate 1) is shown in Plate 2.

We do not use a full circle in Figure 8 because we want a separation between the right and left sides of the hierarchy. This is important when the nodes within the hierarchy are ordered from left to right based on an attribute. In this case the nodes are ordered alphabetically by *filename.*

The aggregate treeMap is an effective interactive display for manipulating large hierarchical data sets for the following reasons:

1. *Automatic Aggregation*

Aggregation is performed automatically on a level by level basis. For each level we consider every node within the level and determine whether it fills a perceivable area of space. If so, we let it be. It not we aggregate it with its nearest, smallest neighbor. This is in contrast to certain tree techniques that collapse entire subtrees to avoid occlusion. In this case, some nodes may be collapsed although they are not contributing to the occlusion problem. Because of aggregation, this treeMap representation can encode very large trees (the tree shown in Figure 8 has 5842 nodes).

When multiple nodes are aggregated, the resulting aggregate node will fill the entire space allotted to its encapsulated nodes. Each node within the aggregate will contribute to a portion of the aggregate node's color. The percentage contribution of each node is dependent on its number of descendants. Larger nodes (i.e. nodes with more descendants) will contribute a greater percentage than smaller nodes. I.e. when nodes of different color are aggregated the aggregate object will be a combination of all their colors. It should be pointed out, though, that combining too many different hues into an aggregate may make it difficult to separate out all the constituent colors from the resulting aggregate color.

Currently we also use color to encode the level of aggregation (i.e. the number of values within the aggregate). If saturation is used to encode a data attribute (as in Figure 8 or Plate 1 and Plate 2) then hue is used to encode the aggregate level. In this example, the more purple a node is, the greater the aggregate level. The saturation of the nodes in Figure 8 is not changed by the aggregate level. On the other hand, when hue is used to encode a data attribute (e.g. *file permissions*) then saturation is used to encode the aggregate level. An example aggregate tree map where hue is used to encode *file permissions* is shown in Plate 3.

By using both hue and saturation to encode two different types of information we recognize that accurate data readings can no longer be extracted from the display. This is because hue and saturation are integral properties, and such properties are difficult for our perceptual system to separate[12]. However, our intention with the color coding is not to provide accurate measurements on the level of aggregation or on the encoded data values but rather to bring the user's attention to the highly aggregated areas which are visually more salient than the other areas in the display. For example in Plate 1 (Figure 8), our eye is drawn to the large magenta area, which is a highly aggregated area. This section corresponds to different folders in the *Mail* directory. Our eye is also drawn to the highly saturated areas which corresponds to files with more recent *modification dates*. The brighter magenta colors in the *Mail* directory correspond to new entries in the *inbox* folder. Another highly saturated area is the subtree indicated by the red circle, which shows recently modified program files for generating the displays used within this paper.

2. Interactive control of the aggregation level

Users may interactively control the level of aggregation by allocating more or less space to the entire tree. This is done by expanding the height or width of the tree. When more space is allocated, aggregate nodes will automatically be de-aggregated if there is sufficient space. Similarly, when less space is allocated, nodes with insufficient space will be automatically aggregated. The process occurs continuously, as with the solarPlot display, thus users can view any level of aggregation. Continuous change also enables users to better maintain context across aggregate levels because changes can occur in small steps.

We have also considered focus + context techniques like the one shown in Figure 4, which allows part of the data set to be aggregated at a different level of detail than the other data values. However, we have not integrated such techniques into the aggregate tree because there is no apparent way to achieve this without introducing false implications into the display - expanding a subtree height may falsely imply a greater number of levels, while expanding a subtree width may falsely imply a greater number of descendants.

3. Aggregation of spatial neighbors

In this case spatial neighbors correspond to siblings within the hierarchical structure represented by the aggregate treeMap. Since siblings within a hierarchical structure usually correspond to related user defined categorizations of data concepts, aggregating them will hopefully not hide too much information from the user. The siblings can also be ordered based on another data attribute, e.g. order the sibling concepts alphabetically (based on their name) so that users can easily identify elements within a particular aggregate group.

More accurate data on the individual nodes may also be obtained by moving the mouse over the chosen nodes (this technique was used by Eick et al.[3]). Moving the mouse over a node will cause information on that node to be displayed as text on the bottom left corner of the display. In the case of aggregate nodes, the names of all member elements will be displayed.

4. Context maintenance across aggregation levels

Subtrees are mapped to angles based on their total number of descendants. For the circular aggregate trees, this mapping remains constant no matter the aggregation level. This ensures that a particular subtree will always be rooted at the same angular value and will always have the same angular spread.

5. Feedback on level of aggregation

The level of aggregation is based on the size of the circle or triangle. A larger aggregate tree-map indicates a lower aggregation level and a smaller tree-map indicates a higher aggregation level.

6. No total loss of data values

All nodes mapped onto the aggregate treeMap are represented to some degree. Nodes that are too small to be perceived are summarized with neighboring nodes, and their appearance changes the appearance of the resulting aggregate. This is in contrast to tree manipulation techniques that miniaturize the data elements without aggregation. In this case some nodes may become too small to see, and thus be culled from the display.

The aggregate treeMap is more effective than node-link diagrams for certain tasks. We extracted some common tasks performed on tree representations[13] and examined which were appropriate for the aggregate treeMap. Some of the tasks in which it excels include: 1) obtaining global relationships and structure from the entire hierarchy, 2) finding clusters, duplicate relationships, and inheritance properties from the structure of the hierarchy, and 3) discovering attributes of entire subtrees. Tasks such as the following are less appropriate for the aggregate treeMap: 1) finding the most recent common ancestor between two nodes, 2) finding the path to a particular node from the root of the hierarchy, 3) getting accurate information on individual nodes, and 4) tree navigation. For the second set of tasks, it is perhaps more appropriate to use a node-link diagram together with some interactive technique such as fisheye-lens[4], or the hyperbolic tree[7].

4 SolarPlot + Aggregate TreeMap

The solarPlot and the aggregate treeMap displays can be integrated to form a hybrid visualization system for viewing data attributes (e.g. e-mail arrival frequency) within the confines of a structured hierarchy (e.g. an organizational hierarchy). Figure 9 (Plate 4) shows a hand-generated organizational hierarchy for a hypothetical company. The apex of the hierarchy represents the company CEO and the first level of the hierarchy shows the vice presidents within the company and so on. The histograms within each node in the hierarchy represents the amount of electronic mail received by each person in the organization throughout the period of a day. The color of each node in the tree encodes the time in which a person joined the organization. From Figure 9 we see that a new division was just added to the company very recently (i.e. the more saturated area, or lightly colored area).

In this hybrid display, when nodes are aggregated the histograms within them would get aggregated as well, showing the electronic mail distributions for several people. Each node or aggregate node is represented by a sector in Figure 9. As before the level of detail should be controllable by changing the global size of the structure.

Based on the visualization in Figure 9 we can begin to explore how the data attributes of objects are related to the hierarchical relationships among the objects. For example in Figure 9 we see that most people in the organization have an approximately flat email traffic distribution (i.e. email arrives at all times in the day). The two main groups of people falling into this distribution category are the vice-presidents, and CEO, as well as the programming staff. Exceptions to this distribution are at the new division and the sales division. People in the new division have a great deal of email coming in at the start, originating from introductions within the company. This traffic, however, disappears after the first few hours and they receive virtually no email thereafter because they have yet to announce their new email addresses to their friends and external colleagues. People in the sales department, however, have a more cyclic email trend, peaking out in the middle of the work day and virtually disappearing after the work day. This indicates that most of the email activity is work related.

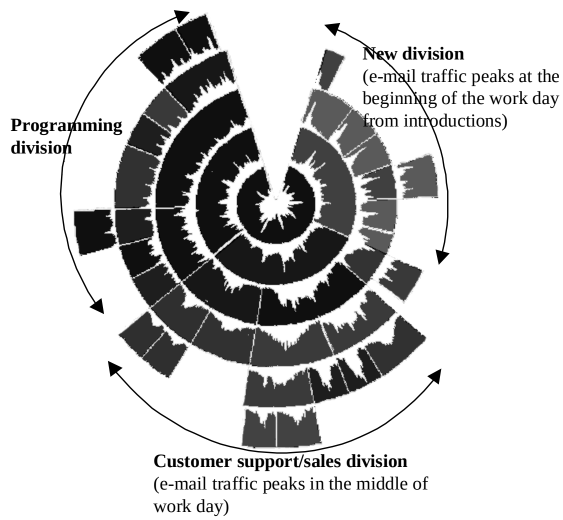

Figure 9: SolarPlot + Aggregate TreeMap hybrid

A problem with the display is that the degree of email aggregation for each person is different. The level of aggregation is based on 1) the level of a person in the hierarchy and 2) the importance of a person in the organization (measured by the number of subordinates in the hierarchy). A more important person in the hierarchy has a greater angular spread to represent the histogram data. However, by the same token, a level higher up in the hierarchy has less total circumference. Another problem with the hybrid display is that each different person has a different angle-to-attribute-value encoding. For example, the CEO has the entire day mapped to 330 degrees; this is different from the number of degrees used for each of the vice presidents, etc. Thus trying to compare particular email time-intervals across people in the hierarchy can be a difficult task.

5 Other Interactive Metaphors

In this paper, we dynamically scale the size of our circular representations to change the level of data detail. Other interactive metaphors for controlling level of detail include:

1. Zoom

Zooming is used in the PAD++ system[2] to control level of detail. When a user zooms in on an object, more information is shown about that object (i.e. a lower level of aggregation is used). Since this interactive method is already a very common navigation technique, it can be easily learned and remembered by end users. One weakness, however, is that there can usually only be one focus area (i.e. two spatially distinct regions cannot be viewed in detail simultaneously). This interactive method can be easily integrated into our visual displays. Zooming into a particular portion of the display will enlarge the focus region and cause the elements within to be de-aggregated according to the larger area available.

2. Spatial distortion

Spatial distortion, like zooming, changes the amount of space allocated to showing portions of the data set. Spatial distortion techniques enlarge particular regions of the visual display while contracting surrounding areas so that the entire visualization still takes up the same amount of space. Unlike zooming, spatial distortion techniques allow multiple focal points.

Distortion techniques are less appropriate for our representations because they may cause our representations to lose their circular property. As a result we may no longer have *constancy in data encoding* (i.e. where a given angle always represents a particular data value). An alternative is to distort the space based on the radial length rather than based on x and y dimensions as is usually done. This distortion method is especially useful for the concentric representation shown in Figure 9. By using radial distortion we can give particular circles more radial space but still show the contextual information at a higher level of aggregation.

3. Magic Lens

Another possibility is to use Magic Lenses[10] to control aggregation for a chosen spatial area. A slider attached to the lens can be used to manipulate the level of detail. This method may also cause us to lose the nice angle encoding constancy property especially if we magnify our representations in-place.

6 Conclusion & Future Work

Aggregation is a useful technique for manipulating large data sets. This paper explores two interactive displays that are tightly coupled with the aggregation technique. These techniques require little user input but at the same time allow users to dynamically control the aggregation level in a continuous and intuitive fashion.

When aggregating a set of data values, we are simplifying the data set. We are therefore bound to lose some information. The goal is to keep the amount of information lost to a minimum with respect to the types of tasks that the user wants to perform. Interactivity helps because users may rapidly view the data at different levels of aggregation. We also provide users with visual feedback so that the different aggregation levels are clearly shown, and so that context can be easily maintained across aggregation levels.

In the future we plan to explore uses of the hybrid technique and how to overcome some of its limitations. We also hope to explore different interactive metaphors and different aggregation schemes[5].

References

1. Becker, A., and Cleveland, W. S., Brushing Scatterplots. *Technometrics*, May 1987, vol. 29, no. 2, p. 127-142.
2. Bederson, B.B. and Hollan, J.D. PAD++:A Zooming Graphical Interface for Exploring Alternate Interface Physics, *UIST '94*, November 1994, p17-27.
3. Eick, S. G., and Steffen, J. L. , Visualizing code profiling line oriented statistics. *Proceedings of Visualization '92*, IEEE, October 1992.
4. Furnas, G.W. Generalized fisheye views. *Proceedings CHI'91 Human Factors in Computing Systems*, ACM, April 1991, p. 16-23.
5. Goldstein, J., Roth, S.F., Kolojejchick, J., and Mattis, J., A framework for knowledge-based interactive data exploration, *Journal of Visual Languages and Computing*, December 1994, p. 339-363.
6. Johnson, B.S., *Treemaps: Visualizing Hierarchical and Categorical Data*, PhD dissertation, Univ. of Maryland, 1993.
7. Lamping, J., Rao, R., and Pirolli, P., A focus + context technique based on hyperbolic geometry for visualizing large hierarchies, *Porceedings CHI'95 Human Factors in Computing Systems*, ACM, 1995, p. 401-408.
8. Leung, Y.K., and Apperley, M.D. A Review and Taxonomy of Distortion-Oriented Presentation Techniques. *ACM Transactions on Computer-Human Interaction*, Vol. 1, No. 2, June 1994, p.126-160.
9. Mackinlay, J.D., Robertson, G.G., and Card, S.K., Perspective wall: Detail and context smoothly integrated. *Proceedings CHI'91 Conference on Human Factors in Computing Systems*, ACM, April 1991, p. 173-179.
10. Stone, M.C., Fishkin, K., and Bier, E.A. The Movable Filter as a User Interface Tool. *Proceedings CHI'94 Human Factors in Computing Systems*, ACM, April 1994, p.306-312.
11. Tenev, T. and Rao, R., Research Report: Managing Multiple Focal Levels in Table Lens, *IEEE Symposium on Information Visualization*, 1997, p. 59-65.
12. Treisman, Anne, Features and Objects: The Fourteenth Bartlett Memorial Lecture, *The Quarterly Journal of Experimental Psychology*, Lawrence Erlbaum Associates Ltd, The Distribution Center, Blackrose Road, Letchworth Herts SG6 IHN, U.K., p. 201-237
13. UMCP OLIVE: Tree, *http://www.otal.umd.edy/Olive/Text/Tree.html*

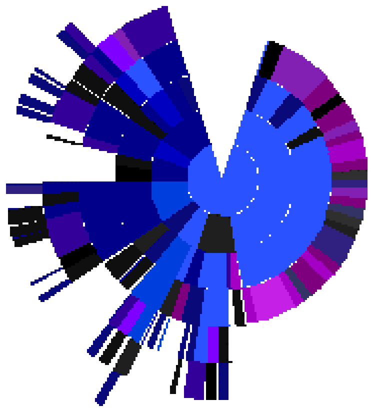

Plate 1: Aggregate tree map with saturation encoding *date-last-modified* [hue encodes level of aggregation]

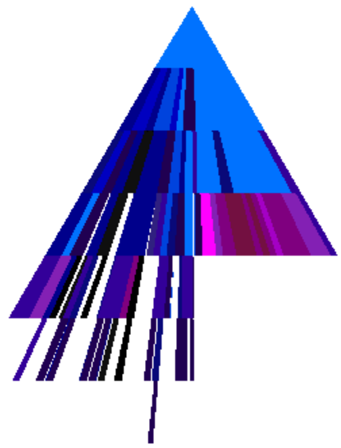

Plate 2: Triangular aggregate tree map with identical data and encodings as Plate 1

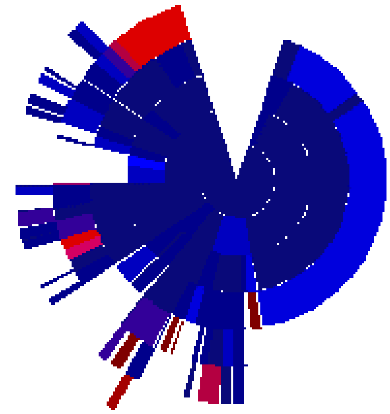

Plate 3: Solar Plot with hue encoding *file-permissions* [saturation encodes level of aggregation]

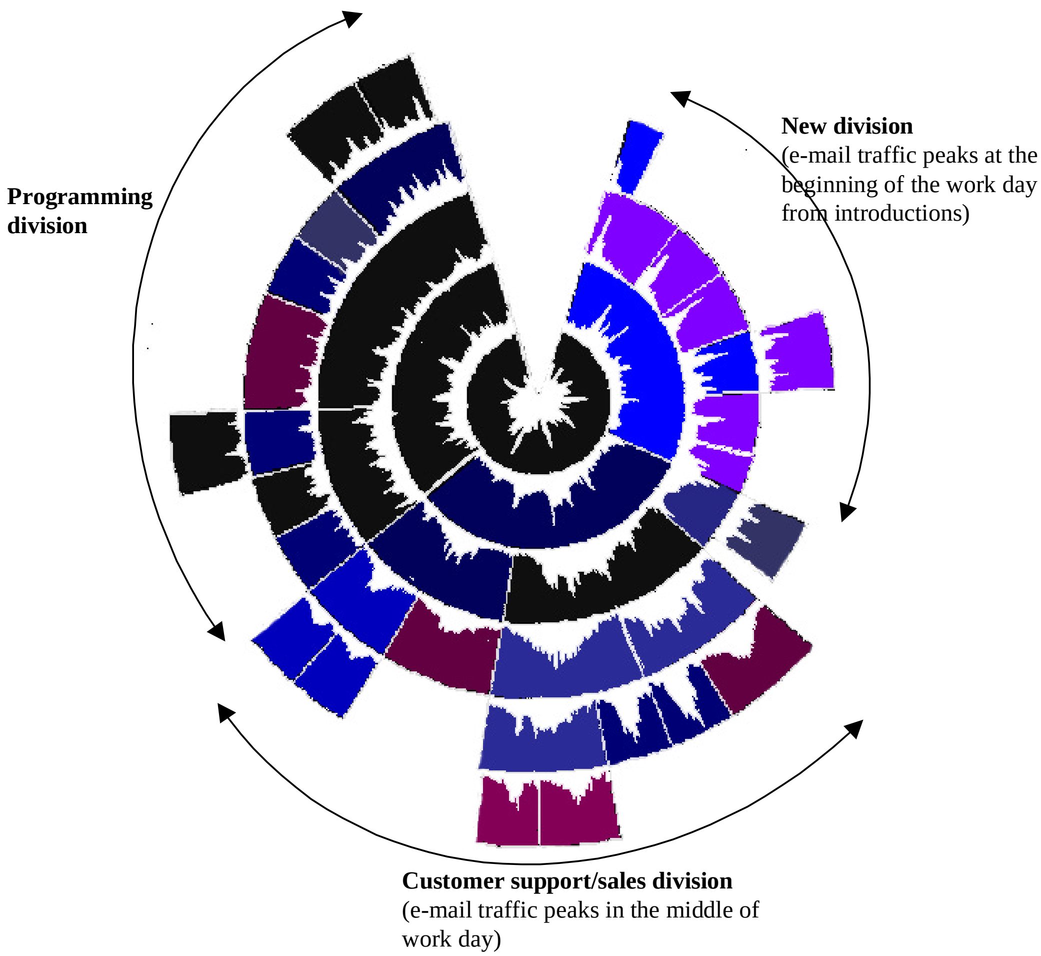

Plate 4: Hybrid display encoding a company employee hierarchy and the *e-mail traffic* of each employee within the period of one day.