TANGRAM TREEMAPS

An enclosure geometrical partitioning method with various shapes

University of Technology, Sydney

Tangram Treemaps

An enclosure geometrical partitioning method with various shapes

By

Jie Liang

Supervisor: A/Prof. Mao Lin Huang

Co-supervisor: Dr. Quang Vinh Nguyen

A thesis submitted in fulfilment for the

Degree of Doctor of Philosophy

In the

Faculty of Engineering and Information Technology

DEC 2012

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

UNIVERSITY OF TECHNOLOGY SYDNEY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

SIGNATURE OF STUDENT

Production Note:
Signature removed prior to publication.

ACKNOWLEDGEMENTS

I would like to gratefully acknowledge the enthusiastic supervision of A/Prof Mao Lin Huang, during this research. He brought me closer to the reality I had initially perceived, eventually enabling me to grasp its rich complexity. This thesis grew out of a series of dialogues with him.

The guidance, motivation and friendship of my co-supervisor, Dr. Quang Vinh Nguyen, has been invaluable on both an academic and a personal level, for which I am extremely grateful.

Furthermore, I owe sincere thankfulness to the members of Visualization Team and fellow researchers and the professors of iNext Research centre.

This research also benefited tremendously from many researchers and staffs in The University of Technology, Sydney. In addition, thank you to the participants for the cooperation and valuable feedbacks in the usability study.

Last but not least, I owe my deepest gratitude to my family and extended family for their continuous encouragement and support to make this PhD thesis possible.

CONTENTS

CONTENTS IN DETAILS

FIGURE LIST

TABLE LIST

EQUATION LIST

SYMBOL LIST

R^2: represents a two-dimensional plane in Euclidean geometry;

S: represents a subset of Euclidean space R^2 is compact if and only if it is closed and bounded.

N: indicates a node is the fundamental unit of which graphs are formed in graph theory. A subset of Nodes are presented by $\mathbf{n_1, n_2, \ldots, n_m}$, such as $\mathbf{N= \{ n_1, n_2, \ldots, n_i, \ldots, n_j, \ldots, n_m\}}$. **m** indicates the Number of Values; **i ,j** indicates the Number of Values.

P: a Polygon bounded by a closed path in a geometry shape. We map Node in tree structure into Polygon representation, e.g. for example, For the Node **N** is transferred as a Polygon **P (N)**

$\boldsymbol{\ell}$: Straight line segments which the polygon composed of. For example, a finite sequence of $\mathbf{L: = \{ \ell (v_1, v_2), \ldots \ell (v_{n-1}, v_n)\}}$. $\mathbf{\ell (v_{e-1}, v_e)}$**:** present The longest side.

V: In the polygon represents the points where two edges meet are the polygon's vertices. A set of vertices which a polygon composed of, are presented in a set of $\mathbf{V: = \{v_1, v_2, \ldots, v_I, \ldots, v_j, \ldots v_n\}}$,

$\mathbf{v_s}$ Initial vertex and $\mathbf{v_{s'}}$ which is the pointv_s transferred to the side after partitioning happened; $\mathbf{v_c}$ which is cutting vertex and $v_{c'}$ which is the point $\mathbf{v_s}$ transferred to the side after partitioning happened,

A: The Area size of polygon. The area size of a polygon equals the area size of a set of sub-polygons $\mathbf{A: = \{a_1, a_2, \ldots, a_i, \ldots a_j, \ldots, a_n\}}$.

W: A weight of a value associated with the property of a vertex. e.g., $\mathbf{W: = \{w_1, w_2, \ldots, w_i, \ldots w_j, \ldots w_n\}}$. $\mathbf{W_{g1}}$ $\mathbf{W_{g2}}$ present subgroups of **W**, e.g., $\mathbf{W: = \{W_{g1}, W_{g2}\}}$;

Θ: an interior angle formed by two sides of a polygon that share an endpoint. **θ:=** **{ θ_1 ,θ_2, …, θ_i,…,θ_j,…,θ_n }** ; θ_{min} defines Minimum Angular resolution Constraint; **α**: partition angle

ALGORITHMS LIST

ABSTRACT

In practices, analysts need to monitor multiple views and real time processes in one physical screen simultaneously regularly, due to the time demands or multi-task requirements. More often the visualization tool shares the screen space with other concurrent projects or process sessions. Although the traditional enclosure (or space-filling) tree approach can guarantee the maximization of space utilization in an isolated session display (that commonly occupies a single rectangular geometrical area), they however do not consider the maximization of display utilization of the whole computer screen, where a number of concurrent sessions are running in one screen.

This thesis proposes a new enclosure visualization method, named Tangram Treemaps that achieves the maximization of the computer screen utilization through the flexibility of display (or container) shapes. Breaking through the limitation of rectangular constraint, the new approach is able to partition various polygonal shapes. Furthermore, our algorithms also improve the efficiency of interactive tree visualization significantly, through the reduction of the computational cost.

Finally, we provide three case studies to demonstrate the commercial value of our method by using different datasets; we evaluate the method according to graph drawing and perceptual guidelines to show the advantage in scientific measurements; we conduct three user studies to compare the performance of our method with the traditional treemaps. Research results have proven that Tangram Treemaps could be adopted into a wider range of applications, taken in account its real-time performance and the quality of the visualization layouts.

CHAPTER 1. INTRODUCTION

We are living in a world, where everyone perceives and collects a certain amount of data every day. However, the raw data itself doesn't contain much value for us to obtain knowledge. Making matters worse, the amount of data we receive is rapidly increasing at a rate faster than our ability of reading (processing) and understanding for making decisions. Data visualization is the channel to speed up the transformation from raw data to right decision, which turns the information overload into an opportunity.

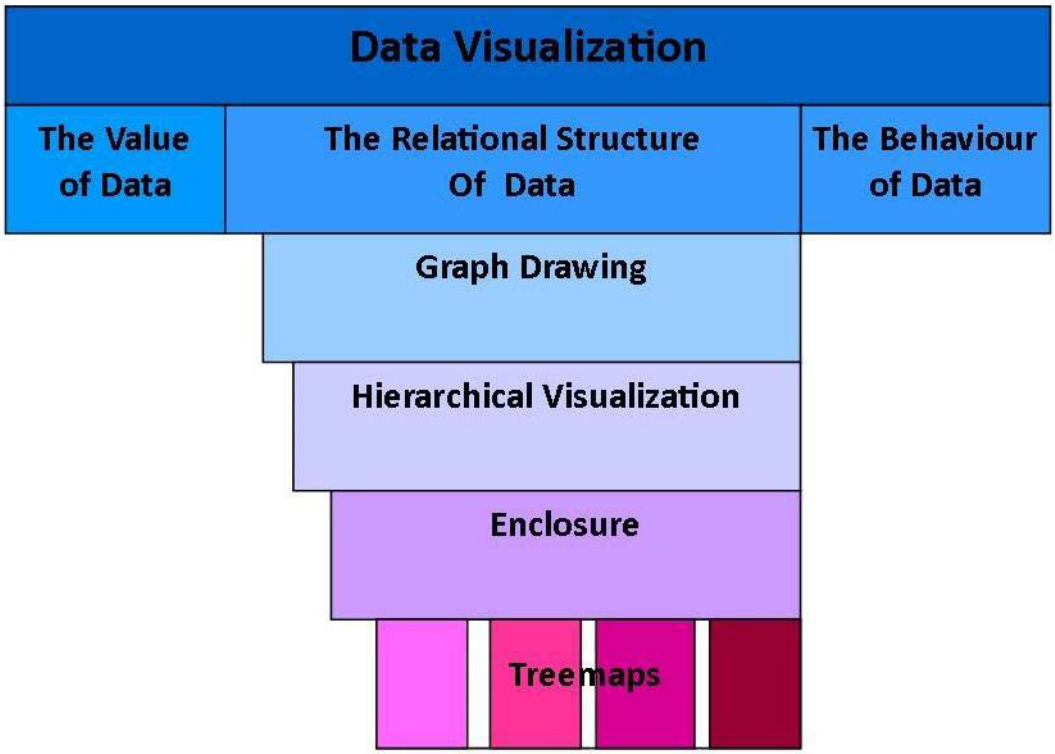


Figure 1-1 Data Visualization Research Scope & Structure

Following the structure of data visualization research (Figure 1-1), This Chapter opens the research scope in Section 1.1 and draws users' attention to Graph visualization, as one of classifications in Section 1.2. Then Section 1.3 moves to Hierarchical visualization

as one stream of Graphic visualization. Furthermore, Section 1.4 narrows down from Hierarchical visualization to enclosure approach. Within enclosure approach, Section 1.5 focuses on treemaps as the main topic of the thesis by introducing the background of treemaps in terms of research and commercial value, and critically reviewing the existing treemaps with advantages and disadvantages. Grounded on the background and related work in treemaps, Section 1.6 addresses the research challenges and specifies the objectives of this thesis research. To meet the challenges, Section 1.7 briefly presents the overview of our new approach. Then Section 1.8 elaborates the original contributions of this research. Finally, the thesis organization is given in section 1.9.

SECTION 1.1 DATA VISUALIZATION

Data visualization is the study of the visual representation of data, and "information that has been abstracted in some schematic form, including attributes or variables for the units of information (Friendly, 2008)".

According to Friedman (2008), the "main goal of data visualization is to communicate information clearly and effectively through graphical means …. To convey ideas effectively, both aesthetic form and functionality need to go hand in hand, providing insights into a rather sparse and complex data set by communicating its key-aspects in a more intuitive way. Yet designers often fail to achieve a balance between form and function, creating gorgeous data visualizations which fail to serve their main purpose — to communicate information".

Most of the visual representations are created to assist viewers to communicate with information in order to understand the data in the following three aspects: Section 1.1.1

the value of data, Section 1.1.2 the relational structure of data and Section 1.1.3 the behaviour of data.

1.1.1 THE VALUE OF DATA

The value of data includes its attributes and elements. We could view value distribution or density in 2D, 3D or high dimensional visual spaces.

For example, Figure 1-2 is a visualization of sun spots from 1850 to 1993. The dots in X and Y coordinate present the data records of the number of aggregated sun spots. Figure 1-3 visualizes a database of films plotted by year, length of film, type ect. In order to express these multiple data values, visualization in Figure 1-2 extends graphical attributes, from position only (Figure 1-1) to combination of size, position, colour and shape. The scatter plots can also be enhanced into 3D visualization. Figure 1-4 is 3 d visualization DNA Replication Inhibition Control. Figure 1-5 represents car data collected by the consumer reports magazine between year 1970 and 1982. It is the visualization of data values presented in a High Dimensional space using Parallel Coordinates, which consists of 406 cars defined by eight attributes.

Figure 1-2 visualization example 1 of data value presented in 2D space: The little image dots represent data records of the number of sun spots, from 1850 to 1993, zoomed in on a small area. (Sourced from GVU Center, Georgia I. T. http://gvu.gatech.edu/).

Figure 1-3 visualization example 2 of data value presented in 2D space: Using Spotfire to visualize a database of films, filtered by category using Action, Comedy, Drama, Music and Science Fiction, plotted by year and length of film with the movie (Golub & Shneiderman,**2003**)

Figure 1-4 visualization example of data value presented in 3D space: Using Spotfire scatter plot for the three-dimensional principal component analysis (sourced from http://www.biomedcentral.com/1471-2105/5/195/figure/F8)

Figure 1-5 The visualization example of data values presented in a High Dimensional space using Parallel Coordinates: It is a set of car data collected from the 1983 ASA Data Exposition (Ramos and Donoho, 1983). The data is about cars tested by the Consumer Reports magazine between the years 1970 and 1982 and consists of 406 cars described by eight attributes.

1.1.2 THE RELATIONAL STRUCTURE OF DATA

The relational structure of data includes the hyperlink structure of websites, and the connection structures of network. Typical example is the node-link diagram. For example, Figure 1-6 is the visualization of data relational structure. It shows the hyperlink structure of Wikipedia website as part of the World Wide Web.

Most visualization of data relationship focuses on the network structures among data items. Figure 1-7 represents the social networks in dimensional visualization, which includes space and time aspects. (Shekhar & Oliver, 2010)

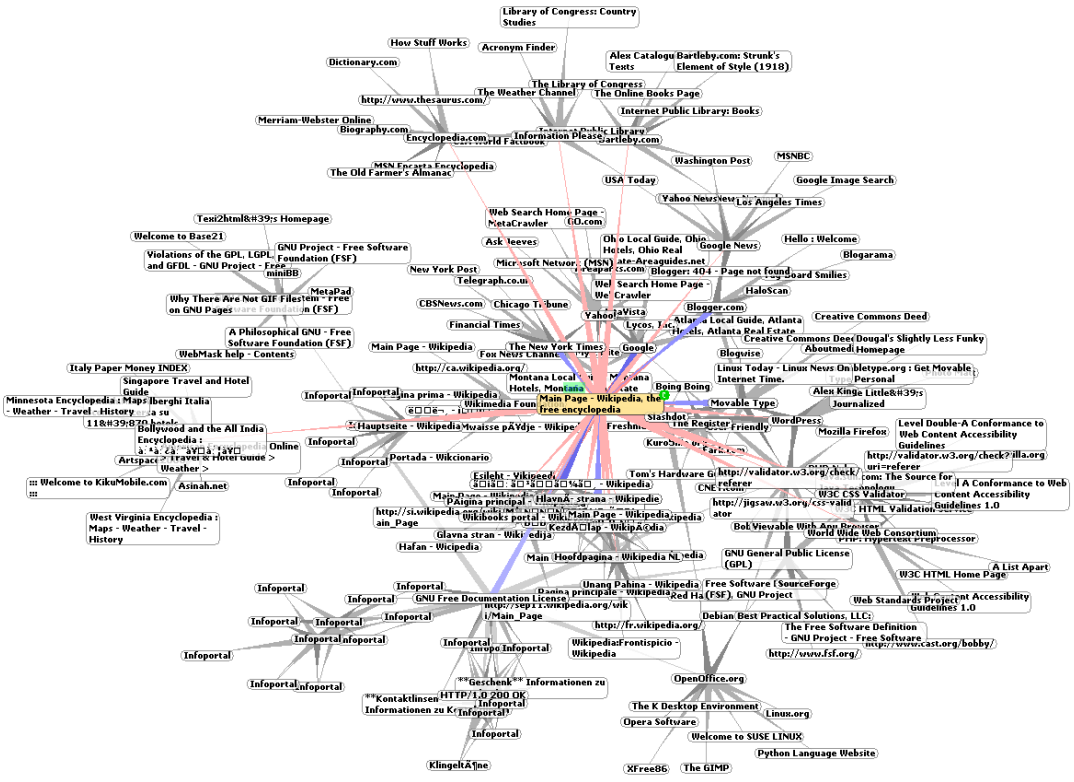


Figure 1-6 The visualization example of data relational structure in 2D: It shows the hyperlink structure of Wikipedia website as part of the World Wide Web. (http://en.wikipedia.org/wiki/File:WorldWideWebAroundWikipedia.png).

Figure 1-7 The visualization example of data relational structure in 3D: A new and efficient way to represent the space and time aspects of social networks. (Shekhar & Oliver, 2010)

In the data visualization of relationships, the main category of visualization is graph visualization, which is built on the base of graph drawing. (Section 1.2) Among data relationships, the hierarchical data is one of common form.(Section 1.3)Treemap enclosure approach is one of visualization technique to visualize hierarchical data. (Section 1.4)

1.1.3 THE BEHAVIOUR OF DATA

The behaviour of data, particularly the behaviour of data movement, is discussed in this section. Data is travelling or transforming from time to time through a variety of data communication and transformation channels, such as telephone network, email network, business sales transaction or finance transaction. We can observe the flow of data volumes to identify abnormal patterns of data movement in a certain time period. This type of the visualization may be used for visual pattern recognition in following typical areas: **Section 1.1.3.1 Data communication patterns, and Section 1.1.3.2 data transaction patterns.**

1.1.3.1 DATA COMMUNICATION PATTERNS

Figure 1-8 display an example of the visualization of data communication patterns. (Lu et al., 2010) It shows the pattern of a DDoS (Distributed Denial of Service) networking attack displayed in the concentric-circle visualization. It updates the dynamic data communication every 10 minutes observation of network.

Figure 1-8 The visualization example of data behaviour: It shows the pattern of a DDoS (Distributed Denial of Service) networking attack displayed in the concentric-circle visualization. It is based on 10 minutes observation of network data communication. (Lu et al., 2010)

1.1.3.2 DATA TRANSACTION PATTERNS

Some visualization techniques can be used for understanding more than one aspect of the data. For example, SeeNet as shown in Figure 1-9 (ref) can be used to view email data volumes, and the behaviour of data communication, generated by AT&T long distance network traffic, and as well as the email network (relational structure). In the visualization, edges represent email connections and weight of edges represents volumes of email data. Visualization of Data transaction patterns also is useful for fraud detection in financial market.

Figure 1-9 The visualization example of data transaction patterns: It uses SeeNet to view email data volumes generated by AT&T long distance network traffic. Edges represent email connections. Weigh and colours of edges represent volumes of email data. (Richard et al., 1995)

This thesis focuses on investigation of novel techniques to visually represent the relational structures among data items. This type of Data visualization is called Graph

Visualization, in which the fundamental theories behind are Graph Drawing and Computational Geometry.

SECTION 1.2 GRAPH VISUALIZATION

Graph drawing is an area of mathematics and computer science combining methods from geometric graph theory and information visualization to derive two-dimensional depictions of graphs arising from applications such as social network analysis, cartography, and bioinformatics.(Di Battista et al., 1994; Herman, 2000) Pictorial representation of the vertices and edges forms a drawing of a graph. Same graph can be represented by different layouts. (Di Battista et al., 1998)

In the abstract of the graph, how vertices are connected by edges is significant. However, In the concrete, the arrangement of these vertices and edges within a drawing is even more important, as the representation layouts directly affects graphical aesthetics and readability and correspondingly affects user's understand ability, and its usability. (Di Battista et al., 1994) The impact of layouts will be upgraded, if the data is dynamic changing and the representation is updating over time. Hence, the general goal of graph drawing is able to preserve the user's mental map (Misue et al., 1995).

Graphs are frequently drawn as node-link diagrams in which the vertices are represented as disks or boxes and the edges are represented as line segments, poly-lines, or curves in the Euclidean plane.(Di Battista et al., 1994) Examples include *force-directed graph drawing(Fruchterman & Reingold, 1991) , Sugiyama graph drawing (Sugiyama & Misue, 1995), orthogonal graph drawing (Eiglsperger et al., 2001) , symmetric graph drawing(Eades & Hong, 2005) and radial graph drawing(Yee et al., 2001) ,* see Figure 1-10 to Figure 1-14.

Commonly-used graphical convention includes arrowheads, which is used to direct their orientation in directed graphs. (Di Battista et al., 1994) However, user studies have shown that other conventions such as tapering provide this information more effectively. (Holten et al., 2009; Holten et al., 2011)

Figure 1-10 An example of Force-Directed drawing of graphs

Figure 1-11 An example of the Sugiyama drawing of graphs

Figure 1-12 An example of orthogonal drawing of graphs

Figure 1-13 An example of symmetric drawing of graphs

Figure 1-14 An example of radial drawing of graphs

Alternative conventions to node-link diagrams include adjacency representations, intersection representations, visibility representations, confluent drawings and visualizations of the adjacency matrix of the graph. Firstly , in adjacency representations such as circle packing, vertices are represented by disjoint regions in the plane and edges are represented by adjacencies between regions; Secondly, in intersection representations, vertices are represented by non-disjoint geometric objects and edges are represented by their intersections; Thirdly, in visibility representations, vertices are represented by regions in the plane and edges are represented by regions that have an

unobstructed line of sight to each other; Fourth, in *confluent drawings*, edges are represented as smooth curves within mathematical train tracks.

Graphs visualization is also widely used to model dependency relationships, which include hierarchical structures. The technique presented in this thesis is focused on hierarchical visualization.

SECTION 1.3 HIERARCHICAL (TREE) VISUALIZATION

In reality, there are many information sources that are organized in hierarchical forms. For example, the organizational structure of a file system, the structure of a classification system, the taxonomy of objects, such as animals, plants, airplanes, etc. They all can be represented in hierarchical structures. Such structures not only play significant roles in their own right, but also provide means for representing the structure of a complex domain in a manageable form. Practically, these hierarchical structures are often very large with thousands or even millions of elements and relationships. As a result, providing an interactive visualization of the entire structure, with capability for deep exploration at different levels of granularity is crucial for the analysts in the knowledge discovery process. For instance, in computer forensics, visualization of file systems can assist in identifying the suspected regions for deeper investigation. Interactive visualization of large decision trees, produced by automatic classifiers from data sets with hundreds and thousands of attributes can assist in better understanding the structure of the classifier and a more efficient visual pruning. The overall view could unveil macro patterns and commonalities in the structure, as well as abnormal substructures in it, which can be further delved into at a lower level of granularity.

The hierarchical approaches originally were proposed in (Warfield, 1977; Carpano, 1980; Sugiyama et al., 1981). In early stage of data visualization research, hierarchical

approach was not a main stream. Techniques in hierarchy visualization have been developed since 1990. Techniques in the visualization of hierarchical structures have been classified into two main categories: the connection and the enclosure (Nguyen & Huang, 2005). Both approaches provide effective visualization of hierarchies. The use of each approach is selected primarily according to the properties of the data in a particular application domain.

The connection approach, such as Classical Hierarchical drawing (Eades et al.,1997), Radial tree drawing (Eades, 1992), Balloon tree drawing (Jeong & Pang, 1998)and Hyperbolic Tree (Lamping et al., 1996), uses a node-link diagram that displays the relationships in information explicitly. This approach generally gives users an immediate perception of the relationships. However, as most of the available display pixels are used as background, the connection approach may become inefficient in certain cases in terms of utilization of display space. Examples are shown in Figure 1-15 to Figure 1-18.

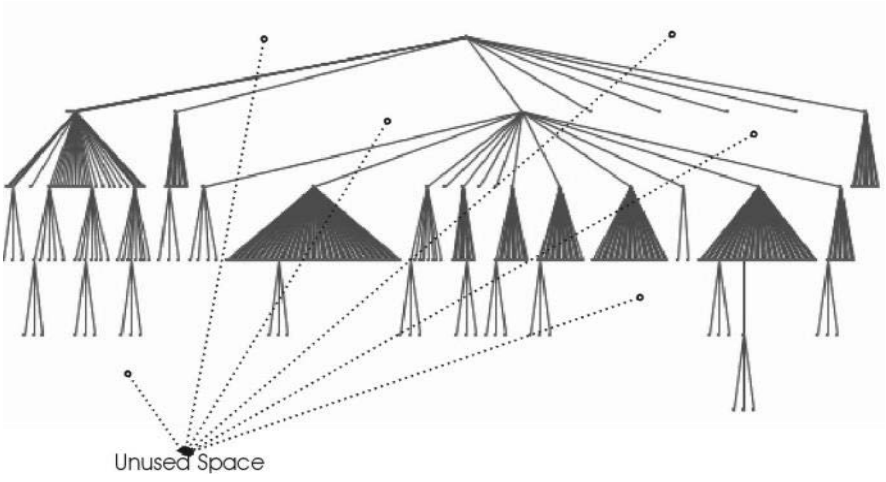


Figure 1-15 An example of classical hierarchical drawing

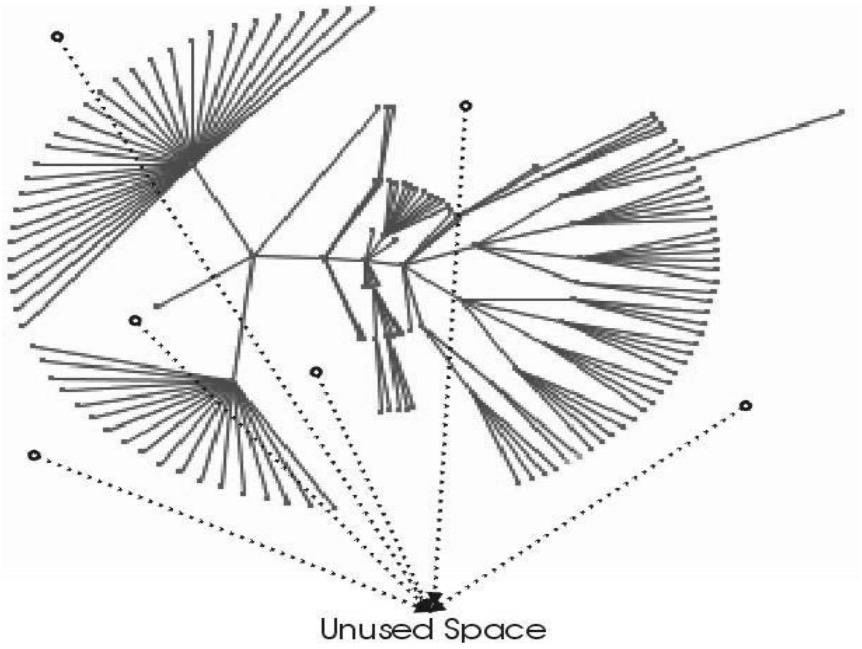


Figure 1-16 An example of the radial tree drawing

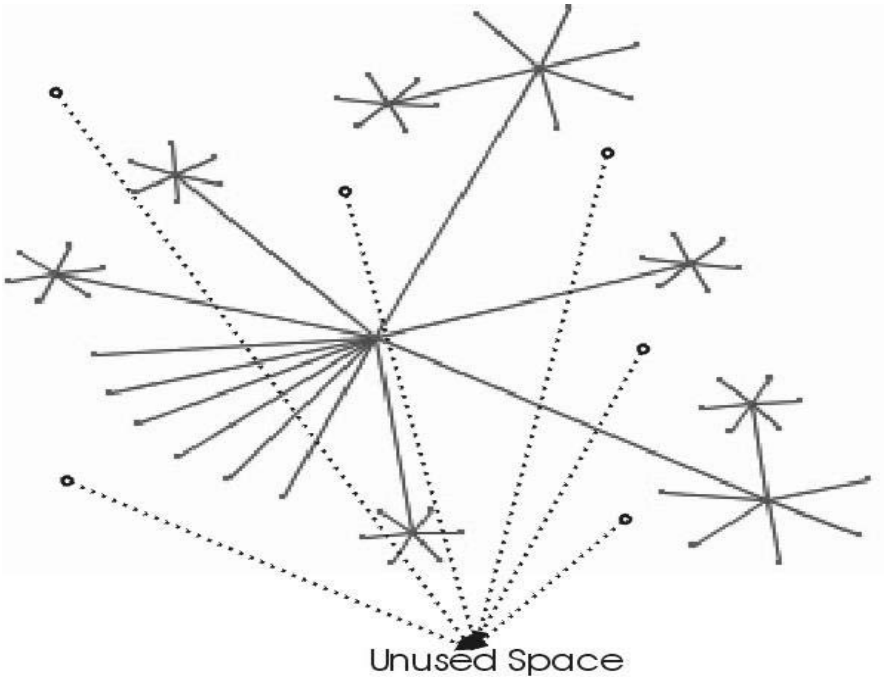


Figure 1-17 An example of balloon tree drawing

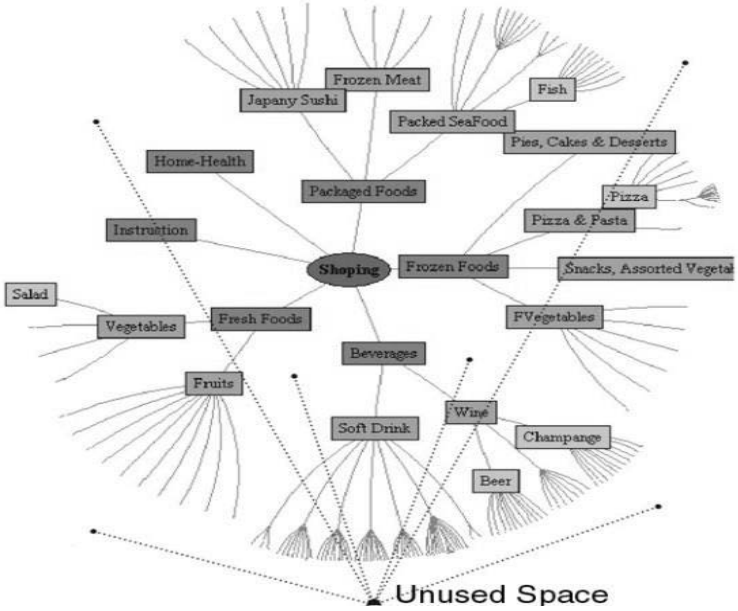


Figure 1-18 An example of Hyperbolic Tree drawing

SECTION 1.4 ENCLOSURE APPROACH

On the other hand, *enclose or space-filling approach* is considered to be a successful method for visualizing large hierarchical data sets with attributed properties. This partitioning method uses *enclosure* to represent the tree structures, ensuring that all nodes and their sub-hierarchies are located inside their "father's" display region. It can provide a visual presentation of global patterns of the overall data structure in a compact display. This technique ensures space efficiency by dividing the display area into nested sub-areas and assigning them as geometrical regions to represent subsets of the entire dataset in display. This is also referred to the term of " containment".

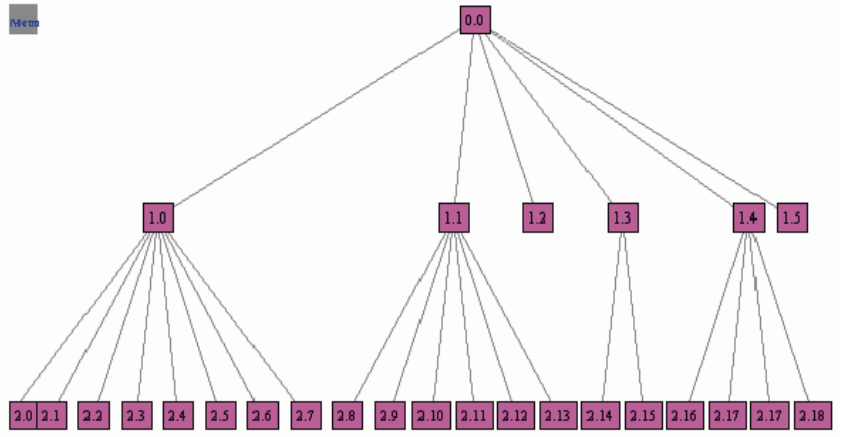
Space-filling techniques, especially Treemaps (Johnson & Shneiderman, 1991; Van wijk &Van de, 1999; Bruls et al., 2000; Bederson et al., 2002), have also shown high applicability and commercial value in many areas, such as finance analysis (Wattenberg 1999), sport reporting (Jin & Banks, 1997), image browsing (Bederson, 2001) and software and file system analysis (Baker & Eick, 1995).

It is important to note that, in tree-maps, the size of the individual rectangles is significant. For example, if the tree represents a file system hierarchy, this size may be proportional to the size of the respective file. This is why tree-maps enjoy popularity in information visualization in spite of the fact that it is difficult to perceive the structure in the representation. This thesis is an attempt to advance representation of structured data in tree map context.

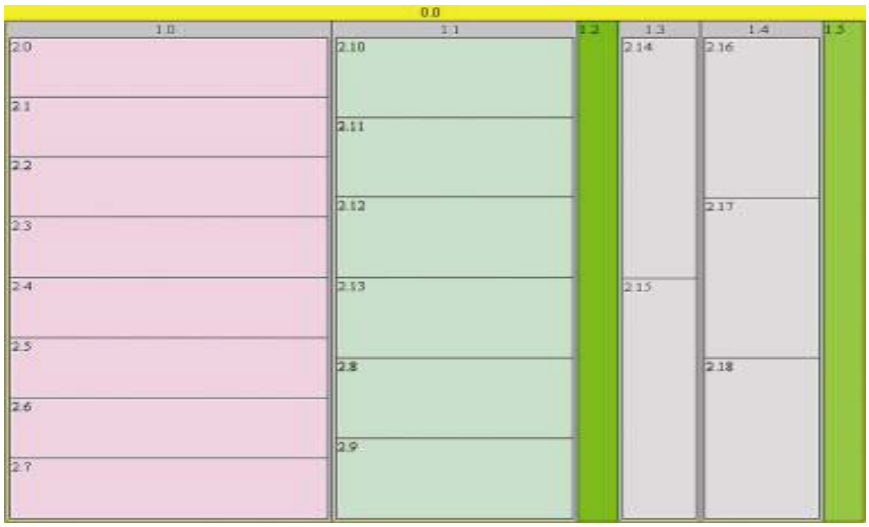
Following sections discuss Treemaps techniques in details, including Section 1.4.1 Slice and Dice Treemaps, Section 1.4.2 squarified treemaps, Section 1.4.3 Voronoi Treemaps, Section 1.4.4 Space-Optimized Treemaps, Section 1.4.5 Radial Edgeless Tree and Section 1.4.6 TreemapBar.

1.4.1 SLICE & DICE TREEMAP

Treemaps, first proposed by Johnson & Shneiderman in 1991, utilize the enclosure partitioning concept to represent hierarchical structures within rectangular display space. The general algorithm is to recursively enclose nodes of the tree in rectangular areas. The sizes of the rectangles are dependent on the number of children of the node to be partitioned. This process is repeated until all the leaf nodes are reached. This original layout of treemap is called Slice and Dice (Johnson & Shneiderman, 1991. Figure 1-19 illustrates a classical node-link diagram view of the tree and its corresponding Slice and Dice Treemaps. Node 0.0, as the root node in Figure 1-19a, is mapped to the maximum rectangular area in Figure 1-19b. These treemaps are then constructed by recursive subdivision of parent node in vertical direction for one layer and horizontal direction for next layer. The result of treemaps reflects the hierarchal information of the tree. However, the disadvantage of this initial layout algorithm (Figure 1-19b) is that the restriction of division directions causes density of rectangles in high aspect ratio, even for small data sets.



(a)



(b)

Figure 1-19 Slice and Dice illustration: Visualization of a small data set using: a) classical tree view, and b) Slice and Dice treemaps technique.

1.4.2 SQUARIFIED TREEMAP

Original tree maps following slice and dice partitioning directions create rectangles nodes with a wide distribution of aspect ratios. The visual representation generated affect users' ability to recognize nodes, especially when data size increases. To assess the effects of aspect ratios on readability, Experimental evaluation have been carried out. The results prove that extreme aspect ratios have significant negative impact on rectangular area judgments and comparisons. To improve the readability, researchers developed algorithms which attempted to optimize rectangle aspect ratios of squares.

They developed qualified layouts based on three assumptions: 1) The border of nodes is reduced, as squares minimize rectangular perimeter; 2) Squares are easier to select with a mouse cursor; 3) Similar aspect ratios closing to one, ease area comparisons. The first and second assertions are both supported by theoretical and empirical evidence (Fitts' law). However, the empirical perception results (Hong et al.,2010) proved square aspect ratios are not optimal for area comparisons, which declaimed third assumption.

Figure 1-20 Squarified treemaps illustration: an example of using squarified treemaps to map the news room (http://www.marcosweskamp.com/blog/archives/000105.html September 16, 2004)

Squarified treemaps have tidy layout with roughly square data elements. The algorithm is built based on aspect ratio aesthetics rule, which was not rooted in empirical perception data. However, the other aspects of graph visualization were not yet taken account in this algorithm. Figure 1-20 shows Squarified treemaps application in news map.

1.4.3 VORONOI TREEMAP

Up to date, most treemaps layout algorithms are limited to rectangular shapes of containers. Balzer and Deussen proposed a polygonal treemaps, which relaxes rectangular constrains, by utilizing centroidal Voronoi tessellations (Aurenhammer, 1999). This method is widely-used for energy minimizations in many domains of application, for example data compression, image processing, and resource planning ect (Du, Q. & Wang, X. 2004).

Figure 1-21 Voronoi Treemap illustration: (a) Enhanced AW Voronoi Treemap layout of 4075 nodes at 10 hierarchy levels (b) Enhanced PW Voronoi Treemap layout of 16288 nodes at (a brighter colour indicates a lower hierarchy level) 7 hierarchy levels (a brighter colour indicates a lower hierarchy level) (Balzer & Deussen, 2005)

Contrary to existent layout algorithms that are based on the subdivision in rectangles, this new layout algorithm enables the subdivision in arbitrary polygons. Voronoi treemaps enables a polygon-based two-dimensional partitioning following the Treemap paradigm. The output is a set of polygons representing the nodes of the top hierarchy level. For the next hierarchy level, this procedure is performed recursively for all top level nodes within the respective polygons. When the recursion ends, the treemap layout is completed. (Figure 1-21)

Voronoi Treemaps have three advantages. Firstly, it offers low aspect ratios as sub-areas have overall aspect ratio between width and height that converges to one. Secondly, It provides better interpretability of hierarchical structures, as siblings are identified clearly without being grouped during the layout process. Thirdly, Voronoi tessellations enable the partitioning of an m-dimensional space without producing holes or

overlapping. It provides flexible adaptability regarding the enclosing shape, as sub-areas are non-regular shape.

Unfortunately, this method has a very high computational cost. The reason is that the computation of the approximation of the layout involves large number of iterations to reach an error below the desirable threshold. As a consequence, it is nearly impossible to process data sets of thousands of nodes or more on a personal computer.

To summarize, voronoi treemaps use the planar voronoi tessellations (Aurenhammer & Klein, 1999) to enable partitioning. On one hand, it gives good aesthetics value according to graphical design based criteria and on the other hand it provides flexibility for visualization in different shapes of containers, like polygons and circles. Thereby the algorithm provides a more flexible adaptation of Treemaps for a wider range of applications. However, the time complexity caused by Voronoi Tree algorithm makes it inappropriate for real time calculation.

1.4.4 SPACE-OPTIMIZED TREEMAP

While most tree maps process partitioning starting on the side of rectangular container, Huang and Nguyen in 2002 created a new tree map concept, which starts partitioning from the centre of area and provides 360 degree freedom to divide the sub-areas. The new partitioning algorithm generates polygonal shaped nodes instead of rectangles. Space-optimized tree takes a different approach, which combines connection and enclosure visualization. To overcome the drawback of unclear visual hierarchal structure, it on one hand maximizes the utilization of display area and on the other hand optimizes the trees into a geometrical area with node and link diagram to be display, so it is named Space-optimized tree.

Figure 1-22 Space –Optimized treemaps illustration: An example of applying Space-Optimized tree to visualize a large data set of approximately 50 000 nodes. (Nguyen & Huang, 2003)

The process is similar to the radial drawing which uses wedge for positioning sub-tree. However, constrain of radial drawing is angle dividing by clock-wisely only so that the further partitioning is not restricted in area. The difference with space-optimized tree is the layout generated restricted in particular local region. It not only follow angle dividing but also recursively positions children of a sub-tree into confined polygon areas. (Figure 1-22)

The Space-optimized Tree takes the advantages of both approaches, which are the enclosure and the connection. Space-optimized Tree adopts tree map paradigm in a new way, while uses a node–link diagram to present the entire hierarchical structure. In order to help users to explore large hierarchical data visualization, Space-optimized treemap adopts a new hybrid viewing technique that combines two viewing methods, the modified semantic zooming and a focus + context technique.

Experiments in (Nguyen & Huang, 2003) show that this approach has full capacity to geometrically partition very large hierarchical structures in a short time. However, the area inward the centre point easily gets crowded when the layers of structure increase. Finally, the algorithm itself does not include any constraint to meet aesthetics and perceptual rules.

1.4.5 RADIAL EDGELESS TREE

To apply visualization techniques to mobile device interfaces, Hao in 2007 proposed an approach, called Radial EdgeLess Tree (RELT), for visualizing and navigating hierarchical information on mobile devices such as a PDA and cellular phones.

Same as Space-optimized tree, it combines the existing connection tree drawing with the space-filling approach to achieve the efficient display of trees in a small geometrical area, such as the screen that are commonly used in mobile devices.

Figure 1-23 Radial Edgeless Tree illustration: An example of Sprint PCS Vision Phone®(left) and RELT emulator (right) (Hao et al., 2007)

Radial edgeless algorithm traverses the tree with depth first, recursively partitions the remaining area, and allocates each partitioned area for a node. By arranging a set of tree nodes as no overlapped polygons adjacent in a radial manner, RELT maximally utilizes the display area while maintaining the structure orientation. Furthermore, the hierarchy generated by this algorithm follows human natural perception direction from north-west to south-east in a top-down manner. For small datasets displayed in screen size, it provides more structural clarity for end users. The simplicity of algorithm minimizes the computational cost, which makes appropriate for mobile device.

This approach also has been extended to adapt for stock market visualization(Hao et al, 2007) and further applied on the device with small screen (J.Hao et al., 2009) and handheld touch screen (Chhetri & Zhang, 2012). However, current algorithm without further improvement is not yet suitable for larger data visualization.

1.4.6 TREEMAP BAR

Treemaps can also be combined into fundamental visualization form. These fundamental graphs can be classified into three major categories: line, bar and pie. The bar is commonly used for visualizing volume data, count data and simple statistics. Nevertheless, bar chart is still useful for numerical comparing categorical data values. However, this basic visualization is not sufficient any more to deal with the complexity of business data with multi-dimensional attributes.

To extend the capability of original bar chart to visualize the dataset with multiple dimensions, Huang in 2009 proposed TreemapBar visualization technique which embeds treemaps inside bar areas in the chart. This approach overcomes the shortcoming of Bar Chart by adding more dimensions in bar Chat. Figure 1-24 illustrates a case study of TreemapBar in stock market analysis. Colour is coded for different category, which provides additional information about the stock performance. It also combines treemap bar chart + table lens interaction technique in order to allow users to view the details of a particular bar when the density of bars increases. However, this approach doesn't use space efficiently, and is not capable for large scale data visualization.

Figure 1-24 TreemapBar illustration: a) Stock analysis with the normal view of TreemapBar, b) Stock analysis with a focused view of TreemapBar, where X axis represents the industry sectors and Y represents the index. (Huang et al, 2009)

SECTION 1.5 RESEARCH CHALLENGES

While most of *enclosure approaches* described above have achieved the maximization of space utilization of a single geometrical area that is allocated only for displaying the visualization, they do not consider the maximization of space utilization of the entire Computer Screen that are commonly shared by multiple sessions, see Figure 1-25. The major problems are outlined as below.

1.5.1 RESEARCH CHALLENGE ONE

Figure 1-25 Research Challenge illustration: An example of a modern computer screen shared by displaying of two parallel sessions: 1) Treemaps Visualization of Newsroom, and 2) an Accounting System. We can see that the screen left two large unused display areas: one in the bottom left corner and another in the right top corner. There is also a large overlapped area between two session displays.

Multiple display-sessions on computer screen cause either unused space or overlapping on screen space. Treemaps and other enclosure methods of hierarchical data visualization guarantee only the 100% utilization of display space that is allocated for displaying the visualization itself. However, in modern computer systems, including

desktops, laptops, iPads and iPhones, the display screen is commonly shared by multiple sessions. For example, it is very common that a Visual Analytics GUI requires the opening multiple windows (or sessions) concurrently, in which one could display an Excel File (the original data), one could display a Parallel Coordinate Visualization (the raw data visualization), and another could display a Disk Diagram (the visual data mining result). Hence, the question is **how we could maximize the utilization of computer screen for displaying three windows and minimize the overlaps among these windows.** So far, none of the existing enclosure methods, including Treemaps, have considered this issue yet. In fact, most of the existing Computer Systems create **either unused spaces or display overlaps among the parallel sessions in the screen,** see Figure 1-25. Hence, it is crucial for visualization designer to consider how to maximize the utilization of whole computer screen. The concern of screen space utilization is the first challenge of the research.

1.5.2 RESEARCH CHALLENGE TWO

Existing treemap algorithms containing rectangular constrain, which create rigidly axis-aligned layout, could not leverage fully the ability of human perception. Most of the existent treemap layout algorithms have one property in common: they are based on and thereby restricted to rectangles. The Gestalt research and Geon Theory has shown that humans have a tendency to seek out whole shapes and they can very quickly detect when one shape is different from another (Ware, 2004). This constraint not only limits degree of freedom drastically, but also restricts the space of layout variability. The issues of high aspect ratios and misinterpretations concerning the hierarchical structure are consequential symptoms. Additionally, the restriction to rectangular shape implies that the layout of treemaps can only take place within rectangular display areas. More

complex shapes like circles, triangles, and arbitrary polygons are not possible. Although these shapes may not be necessary, treemaps visualizations are used independently. However, by embedding treemap layouts within more complex display environment, such as in a Visual Analytics process with multiple windowing, a better adaptability is quite useful or even essential. Therefore, the concern of **visualization layout variability** is the second challenge of the research.

1.5.3 RESEARCH CHALLENGE THREE

It is inappropriate for complicated algorithms to be applied into real time applications. In last decades, a variety of alternative Treemaps have been proposed, such as Clustered Treemaps (Wattenberg, 1999), Cushion Treemaps (Van wijk &Van de, 1999), Squarified Treemaps (Van wijk &Van de, 1999), Ordered and Quantum Treemaps (Bruls et al., 2000). Wijk and Wetering (Bederson et al., 2002) identified that although these further developments of Treemaps significantly improve the readability, some algorithms can only effectively display static visualization. However, when data in most cases is dynamically updating, they are not be able to *maintain stability over time in the display of changing data* (Bruls et al., 2000; Bederson et al., 2002; Onak & Sidiropoulos, 2008) . Therefore, it is vital to develop appropriate treemaps algorithms to deal with dynamic data. The concern of low computational complexity is the third challenge of the research.

SECTION 1.6 RESEARCH OBJECTIVITIES

The overall objective of this research is to investigate *enclosure data visualization* approaches which address all of the challenges in Section 1.5. More specifically the research objectives are described below:

- **Objective 1**: To investigate *enclosure data visualization* techniques that can maximize the overall utilization of computer screens, which have multiple session displays.
- **Objective 2:** To investigate *enclosure data visualization* techniques that can minimize the overlapping among multiple session displays in computer screens.
- **Objective 3:** To investigate alternative Treemaps techniques that can partition hierarchical data structures in a variety of shapes. This objective is to achieve the layout variability in enclosure data visualization.
- **Objective 4:** To investigate optimized enclosure partitioning algorithms that can reduce the time complexity to quickly produce layouts. The objective is to meet the requirements for displaying of dynamical data.
- **Objective 5:** To conduct an experimental (or scientific) evaluation of techniques produced in objectives 1 and 2.
- **Objective 6:** To conduct a user-centred evaluation (or usability study) of techniques produced in objectives 1 and 3.

SECTION 1.7 OUR NEW APPROACH

This thesis proposes a new enclosure data visualization method - Tangram Treemaps. It allows users to create hierarchical data visualization within the area of arbitrary shapes, such as polygons with various angles. At the beginning of area partitioning, a geometrical region of user-defined shape *R* is chosen and a data set with its own specified tree structure *T* is also confirmed. We then apply the Tangram partitioning algorithm to draw the tree *T* within the region *R*. The partitioning (or drawing) outcome can be expressed as $D = T \rightarrow R$

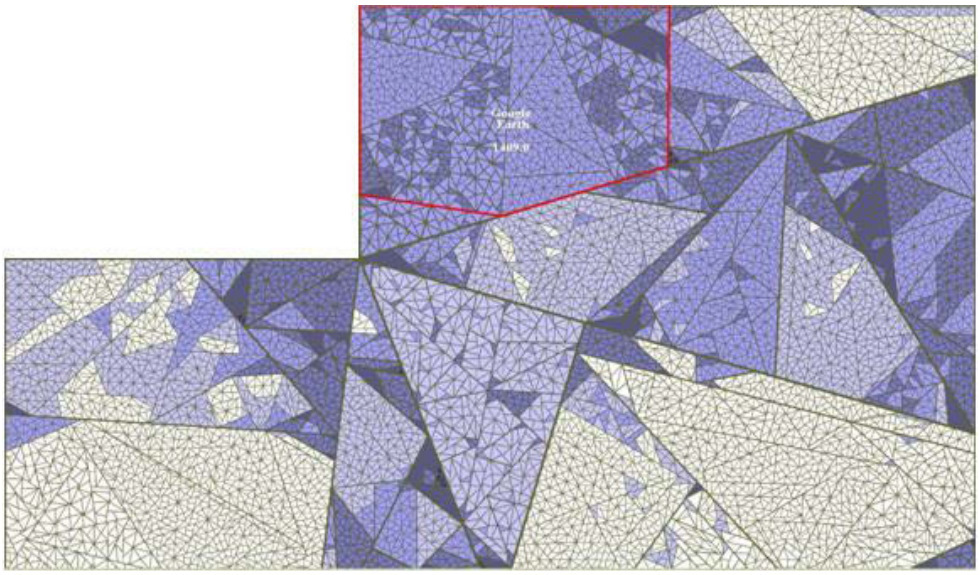


Figure 1-26 Our new approach illustration 1: An example of our new Tangram Treemaps that partition a hierarchical data structure in a polygon shape.

Tangram Treemaps provide a major departure from the traditional enclosure tree visualization methods. It does not require the area partitioning to be limited with the use of a set of rectangular containers. The previous constraint, affected to all existing enclosure tree visualizations, limits the potential freedom of development in different geometrical shapes of containers, other than regular rectangles.

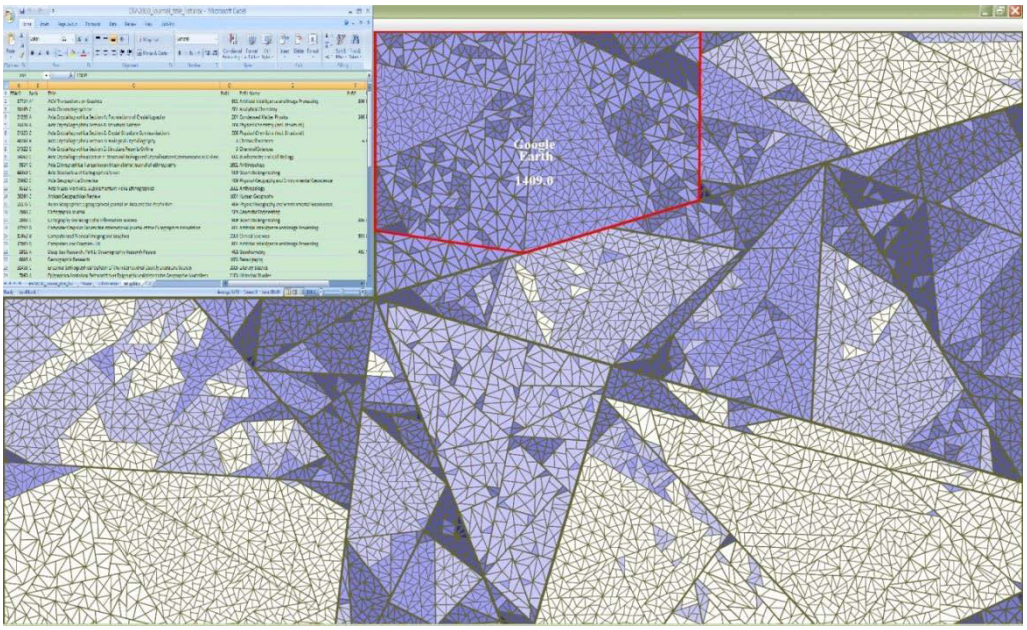


Figure 1-27 Our new approach illustration 2: An example of computer screen that achieves the maximization (100%) of space utilization and the minimization (0%) of the overlap among two session displays by using the new Tangram Treemaps as shown in Figure 1-26.

Tangram Treemaps aim to breakthrough limitation of flexibility whilst fulfils the enclosure space utilization. It has the capability to effectively relax rectangular constraint and functionally provides flexible adaptability into any enclosure shapes, including convex and concave polygon shapes. Most importantly, the simple and flexible algorithm can produce a layout for a very large data set in real time visualization.

Because our new approach gives a great flexibility in the design of geometrical shapes, the new approach allows users to achieve both the **maximization** (100%) of space utilization and the **minimization** (0%) of the overlapping among multiple session displays in a computer screen (See Figure 1-27).

This thesis raises a new objective in the design of containers for multiple session displays, including visualization display. It argues the importance of using various shaped geometrical regions for the visualization of trees. It proposes an enclosure tree visualization technique named Tangram Treemaps that can be used to achieve the new objective; that is the maximization of the computer screen utilization, while multiple sessions are running concurrently. We discuss the technical detail of our new approach and provide three case studies of domain-specific tree partitioning.

SECTION 1.8 CONTRIBUTIONS

Up to date, most of existing enclosure tree visualization techniques only concern the maximization of space utilization in a single session display medium. They declare 100 % space utilization in comparison with the node-link diagram approach. However, this declaration is made under the assumption that there is only one session running and displaying in the physical display medium (the entire computer screen). However, in many cases this assumption may not true as most of the modern computer system are capable to support multiple sessions and windows running on one screen simultaneously. Therefore, the major significance of the thesis is first time raising the issue of maximizing the multi-session display space utilization. The major contribution of the thesis is providing appropriate solution to this issue for the first time. The proposed technique has been evaluated with scientific experiments and user studies and tested in three case studies.

Specific contributions of this thesis are:

- It proposes a flexible enclosure tree layout method for partitioning trees with various polygonal shapes that breaking the limitation of rectangular constraint. Moving beyond layout with rectangles, the flexibility of our Tangram Treemaps can be adopted into a wider range of applications, within different boundary of polygonal shape of containers.

- It proposes a flexible method for generating different layouts in the visualization including vertical-horizontal rectangular, angular rectangles and polygonal layouts. Additionally, with a library of different shape containers, users can make own preference over layouts and change any time during the analysis process, for different scenario-based tasks.

- It proposes a method that combines different types of layouts in the visualization to emphasize the importance.
- It proposes a method that achieves the simplicity of algorithms. Simplicity in the algorithm allows Tangram Treemaps to work efficiently for static visualization. Importantly, the low computational time makes it appropriate to be applied to real time applications.

Several sections of this thesis have been submitted and published (refer to Publication List in appendix) as below:

- Algorithms and experimental results **[IV12a] [IVTUE13]**
- Interaction mechanisms **[CGIV08] [IV10]**
- Cases studies **[IV09]**
- Evaluation and User studies **[IV12b] [OzCHI2012]**
- Perceptual Guidelines **[ICMLA 2012] [DVVA13] [IVMLA13]**

SECTION 1.9 THESIS ORGANIZATION

In this thesis, we present the methodology of the new approach, discuss the implementation algorithms and demonstrate its capability in cases studies. We evaluate the approach based on optimization criteria and graph drawing aesthetics. Finally, we conduct three formal user studies in typical tasks of visual data analysis and assess the performance of Tangram Treemaps in comparison with traditional treemaps.

The thesis is organized as follows: Following the introduction and background introduced in Chapter 1, Chapter 2 explains our approach. Section 2.1 first introduces the original idea and Section 2.2 describes the philosophy of Tangram Treemaps and outlines the pipeline of visualization process for Tangram Treemaps.

Chapter 3 explains how the theory is developed into our work, and discusses the technique and algorithms of our work in details. Following the pipeline of visualization outlined in Section 2.2, Section 3.1 defines technical specifications; Section 3.2 explains the algorithms of the approach with illustrations, examples and experimental results.

Chapter 4 describes the interaction mechanism which integrates our work into interactive visualization.

Chapter 5 examines the computational complexity of the algorithms and evaluates our approach according to a set of design guidelines, including Aspect Ratio of Rectangular Areas and Proximity of Nodes positions, in comparison with other standard treemaps techniques.

Chapter 6 further investigates how well Tangram Treemaps work in the scenario based tasks during the visual analysis process. We conduct three formal user studies to compare Tangram Treemaps with other typical rectangular treemaps. First study is to conduct controlled experiments in locating the object(s) in the structure as a typical task of hierarchical exploration; Second study combines hierarchical exploration and value

comparison; Third study assesses the effect of rectangular area size judgment, as one common task of visual comparison.

Chapter 7 synthesizes algorithm and interaction into the application and presents a number of case studies and demonstrates the capabilities of Tangram Treemaps in interactive visualization environment.

Finally, the work is concluded in Chapter 8, which recaps the research strength and connects our findings with research challenges in Section 8.1 and original contributions in Section 8.2. Section 8.3 discusses the limitations of the research, which opens up opportunities for future work. The areas for further development and researches include Technical Improvements, Alignment with Industry, Treemap Design Guidelines and Systematic Treemap Evaluation Principles. Last, 4 summarize the significance of the research in the field of data visualization.

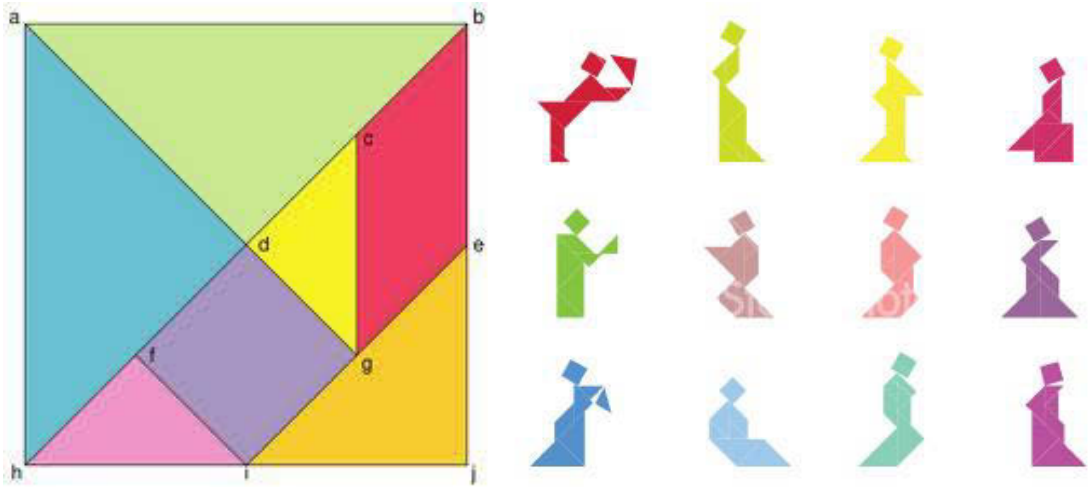
CHAPTER 2. TANGRAM TREEMAPS

The new approach originates from the new objective of maximizing the overall utilization of computer screens. By taking advantages of various existing enclosure methods, it should further improve the quality of enclosure approach in reduction of computational complexity, increase of readability and visual clutter reduction.

In brief, Section 2.1 first introduces the original idea coming from ancient game. Section 2.2 presents the overall framework of methodology.

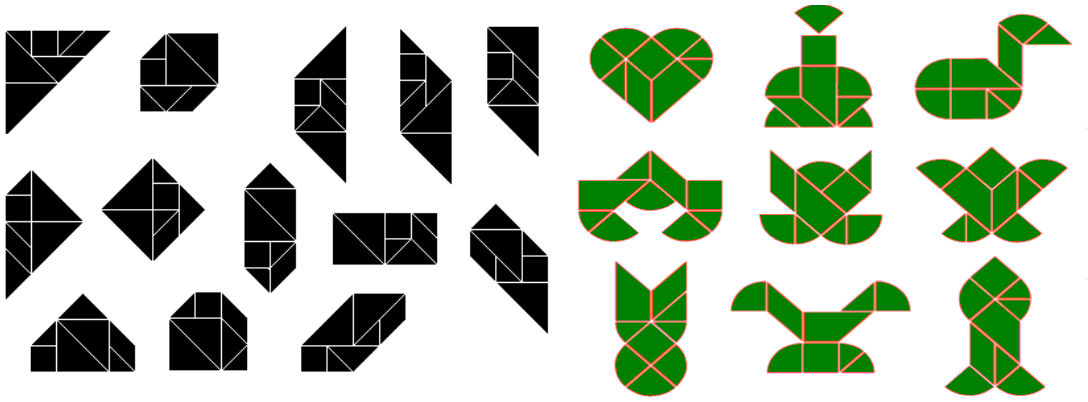
SECTION 2.1 ORIGINAL IDEA

The original idea is inspired from an ancient puzzle game named "Tangram" (Chinese: 七巧板; pinyin: qī qiǎo bǎn), which first was invented in China and then carried over to Europe by trading ships in the early 19th century. (Slocum & Jerry, 2001) **Tangram,** the keyword of our **approach** is one of most popular puzzles game, which uses 7 different flat pieces cutting out from a square (See Figure 2-1 a) to make up different kinds of shapes, e.g. human(See Figure 2-1 b), animal(See Figure 2-2b) and etc. The rule of the puzzle is to form a specific shape (given only an outline or silhouette) using all seven pieces, which should not overlap (see Figure 2-2a). Below shows illustrations of convex tangram configurations and the variation called Cardio Tangram.



(a) **(b)**

Figure 2-1 Tangram illustration 1: (a) Tangram is an ancient Chinese puzzle consisting of seven shapes(Sourced from http://www.walkingrandomly.com/?p=65) (b)Seven pieces of puzzles can be put together and build different figures. (Sourced from http://www.istockphoto.com/stock-illustration-3652877-tangram-people-set-003.php)



(a) **(b)**

Figure 2-2 Tangram illustration 2: (a) Thirteen convex tangram configurations (Sourced from http://en.wikipedia.org/wiki/File:13convexesTangram.png) (Fu Traing Wang and Chuan-Chin Hsiung proved it in 1942) (b) Cardio Tangram Variations (Sourced from http://www.juegotangram.com.ar/IDS/EN/tipostangram/CardioTangram/CardioTangram.htm)

SECTION 2.2 FRAMEWORK

We present the proposed Tangram Treemaps in an overall view with a framework, including evolution of idea and visualization pipelines.

2.2.1 IDEA EVOLUTION

Prior work in visualization limits the adaptability to various shapes in the design of display containers, besides of rectangles. There shapes are sometimes more closing to the physical or natural shapes, for example, like fluctuation, evolution, landscape, biological object etc. The methodology proposed here begins from the requirement of visualizing data structure in the shape other than rectangular.

In computer graphics, all the shapes could be simulated by multiple pointed polygons with infinity or finite sides. Hence, in order to be employed in polygonal shaped container, we aim to invent a new polygonal partition method.

In context of enclosure approach, the new approach shall utilize the treemap paradigm, by adopting containment which enclosure child node in parent node and encode value using area. The new approach is able to use containment arrangement in three ways: the shape of container, the shape of child or the change of both. Therefore, the new method offers flexibility in these three ways to represent and emphasis structure data visualization. With modification or combination of the algorithms, the new approach should be capable of producing multiple layouts for various polygonal shapes.

The evolution of research approach is abstracted in the concept map (Figure 2-3). In the first stage, we relax rectangular constraint completely and add freedom for partition direction rather than horizontal and vertical directions only. The new tessellation process creates visualization mainly with triangular tilting. D&C Triangular Approach is created in this stage. In order to increase visibility, we add one condition with minimum angular resolution. D&C Triangular approach is improved with the angular resolution

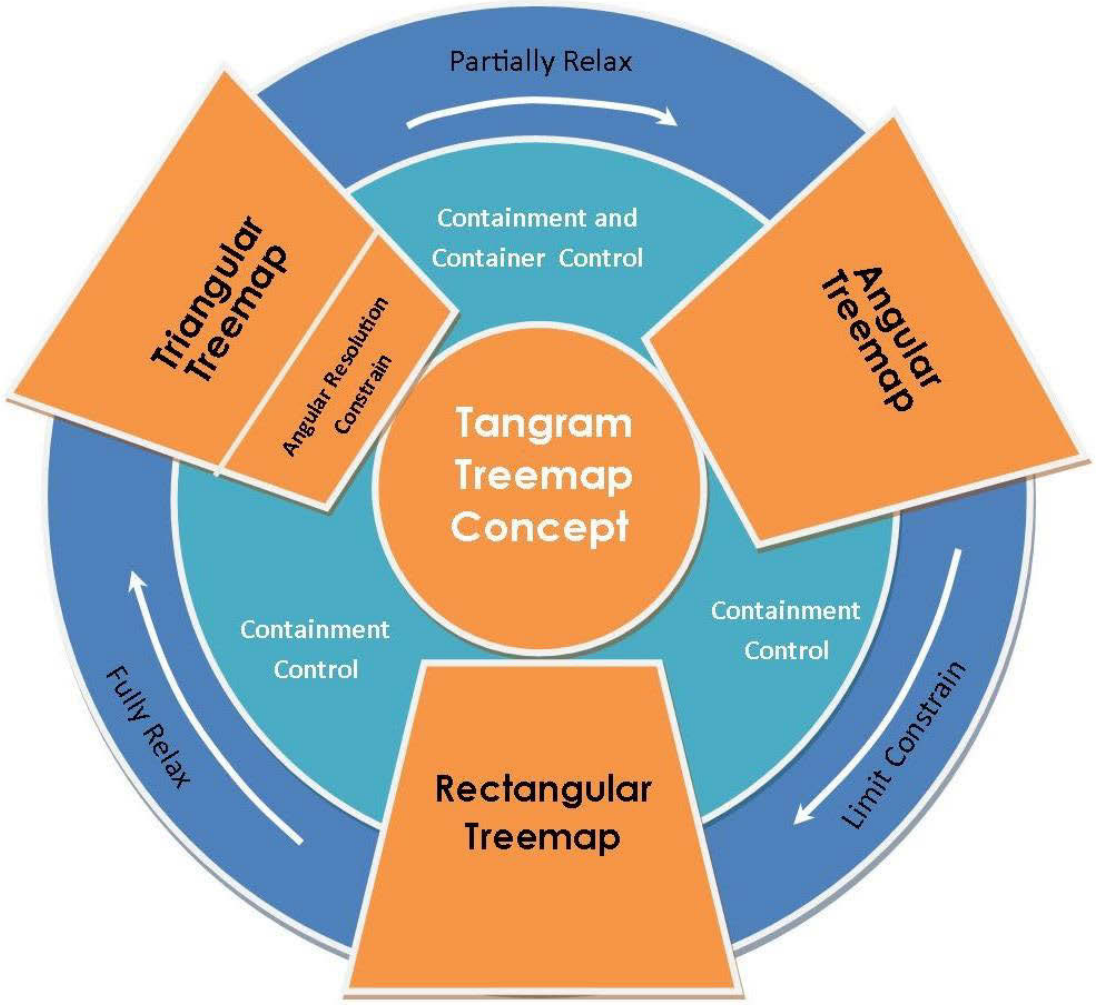


Figure 2-3 The concept map of Tangram Treemap's idea evolution

constraint. In the second stage, we maintain orientation in the same direction, by shrinking the relaxation of constrain. In this way, we preserve the partition directions with fixed angle condition. This condition generates layouts mainly with polygonal titling. This stage creates Angular Polygonal approach. In third stage, we limit fixed angle condition to 90 degrees. This constraint generates D&C rectangular approach.

In order to fit into user tasks and scenarios, we employ these approaches in three environments mentioned above: Containment control (the shape of child), Container control(the shape of container) and both of Containment and container control. So that application designers can utilize it with other visualization methods for diverse domain

purpose and users can make own preference over layouts and change any time during the analysis process.

2.2.2 VISUALIZATION PROCESS PIPELINE

The principle of *Tangram Treemaps* is to produce quickly layouts for large hierarchical data in two-dimensional space. The efficiency of the proposed method is based on the combination of the simple and effective Divide-And-Conquer approach with the Treemap's paradigm. Particularly, it inherits the *area partitioning* technique from the *enclosure* approach to ensure the maximum utilization of geometrical space for displaying trees, while it can also emphases and highlight hierarchical structure. The series of procedures for constructing such visualization can be described as below:

Procedure 1 – Weight Calculation: This procedure calculates weight values for every node in the hierarchy. A weight of a node is defined as a value associated with the property of a node. For example, the weight of a vertex might depend on the domain-specific attributes of the node such as the size of a file or an object, the role of a person in an organisation, and etc. In this thesis, we simplify the weight of a node as the value associate with the numbers of its descendant nodes.

Procedure 2 – Area Partitioning: In a bounded area, procedure 2 recursively partition the entire display area into a set of sub-display areas called local region, according to the nodes in the hierarchy. After partitioning, each sub area is then assigned to each node in the hierarchical structure. The local region of a particular node is the sum of the areas assigned to the children of the node. The area of local region proportionally depends on the value of weight associated with the node.

Procedure 3 – Node positioning: After partitioning, procedure 3 computes the positions of all nodes and their sub-nodes inside their display local regions. Each node is bounded by a local region and the drawing of the sub-hierarchy is restricted to inside of the geometrical area. In most cases, the position of a vertex is allocated at the centre of its local region.

Procedure 4 –Attributes assignment: For individual node, Procedure 4 assigns the graphical attributes, including the *size*, *label*, *shape*, and *colour* based on their levelling property in the

hierarchy. According to the *enclosure* visualisation scheme, the local region assigned to a child vertex is always smaller than the one assigned to its parent. Procedure 4 uses colour code to assign alternative values for individual node. To emphase and distinguish hierarchical structure, procedures utilizes visual cues to clarify the presentation, including boundary, gap, colour saturation and etc. To interact with visual representation, procedure 4 employs the interaction scheme along with animation at this stage as well.

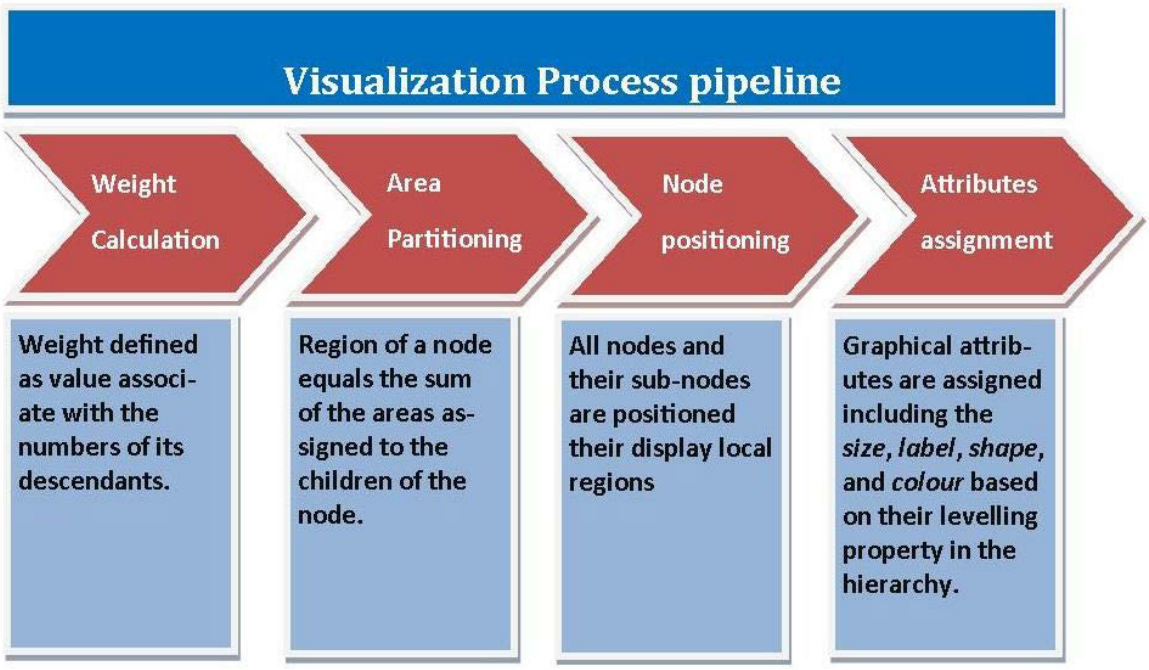


Figure 2-4: Flow chart of visualization process pipeline for Tangram Treemaps

CHAPTER 3. TECHNIQUES AND ALGORITHMS

Before explaining the methodology, Section 3.1 provides technical specifications covering the topic of Basic Properties in 3.1.2, Weight Calculation in 3.1.3, Position of Vertices in 3.1.4 and Tessellation Methods in 3.1.5, and D&C Tessellation Properties in 3.1.5.4. After deciding the Tessellation Method, Section 3.4 discusses The Tangram Treemaps in details. In Section 3.4, it deliberates each approach in the timeline of approach evolution, D&C Triangular Approach in section 3.4.2, Angular Polygonal Treemap in Section 2.4.3, and D&C Rectangular Approach in Section 3.4.4. Each section of these three approaches includes the illustration of partitioning process with examples and explanation of the algorithm and the experimental results. After introducing three approaches, Section 3.5 Tangram Treemaps demonstrate how to take advantages of all these approaches in three ways, control of containment, container and both. Finally Section 3.4 summarizes the chapter.

SECTION 3.1 TECHNIQUE SPECIFICATION

3.1.1 TECHNICAL CONVENTION

To streamline the graphics notation and convention in technical sections, all the symbols and notation are defined in this section.

In the geometry space, $\mathbf{R^2}$ represents a two-dimensional plane in Euclidean geometry; **S** represents a subset of Euclidean space R^2 is compact if and only if it is closed and bounded.

In the graphics within the geometry space $\mathbf{R^2}$, **N** indicates a node is the fundamental unit of which graphs are formed in graph theory. A subset of Nodes are presented by $\mathbf{n_1, n_2,\ldots, n_m}$, such as $\mathbf{N= \{ n_1, n_2,\ldots, n_i,\ldots,n_j,\ldots, n_m\}}$ **m** indicates the Number of Values; **i ,j** indicates the Number of Values.

In a geometry shape, **P** is a Polygon bounded by a closed path. We map Node in tree structure into Polygon representation, for example, For the Node **N** is transferred as a Polygon **P(N)**; ℓ is Straight line segments, which the polygon composed of . For example, a finite sequence of $\mathbf{L: = \{\ell\ (v_1, v_2), .. \ \ell\ (v_{n-1}, v_n)\ \}}$. We use $\mathbf{\ell\ (v_{e-1}, v_e)}$ to present The longest side.

In the polygon, **V** represents the points where two edges meet are the polygon's vertices. A set of vertices which a polygon composed of, are presented in a set of $\mathbf{V: = \{\ v_1, v_2, \ldots, v_I,\ldots, v_j,\ldots v_n\}}$, including v_s which is Initial vertex and $v_{s'}$ which is the pointv_s transferred to the opposite side after partitioning happened; v_c which is cutting vertex and $v_{c'}$ which is the point v_s transferred to the opposite side after partitioning happened, About the properties of polygon, we have **A** which is The Area size of polygon. The area size of a polygon equals the area size of a set of sub-polygons $\mathbf{A : = \{a_1,a_2,\ldots,a_i,\ldots a_j,\ldots,a_n\}}$.

W presents a weight of a value associated with the property of a vertex. The weight of a vertex might be dependent to the domain-specific attributes of that vertex such as the size of a file or an object, the role of a person in an organisation, etc. In this thesis, the weight of

a vertex is simply defined based on the numbers of its descendants, e.g. **W: = { w_1,w_2,..,w_i,..w_j,...w_n}**. **W_{g1} W_{g2}** present subgroups of **W**, e.g. **W: = { W_{g1} , W_{g2}}** ;

θ is an **interior angle** formed by two sides of a **polygon** that share an endpoint. For a simple, convex or concave of the polygon, this angle will be an angle on the 'inner side' of the polygon. A polygon has exactly one internal angle per **vertex**. If every internal angle of a simple, closed polygon is less than 180°, the polygon is called **convex**. **θ:= { θ_1 ,θ_2, …, θ_i,...,θ_j,...,θ_n }** ; **θ_{min}** defines Minimum Angular resolution Constraint; **α** is partition angle.

3.1.2 BASIC PROPERTIES

Tangram Treemaps are applicable to all kinds of hierarchical graphs, focusing on the rooted tree structures. Before explanation of algorithms, basic properties are defined here. Terminologically, a rooted tree contains a tree ***T*** and a distinguished and only one vertex ***r*** of ***T***. The node ***r*** is defined as the root of ***T***. In other words, ***T*** can be viewed as a directed acyclic graph with all edges oriented away from the root ***r***. To avoid the confusion with the **side vertex** of a polygon used in our partitioning process, we call a node ***N*** to represent a vertex in ***T*** and a side vertex of a polygon ***P*** is represented by ***v.*** If ***T*** contains a node ***n***, then the sub-tree ***T(N)*** rooted at ***n*** is the sub-graph induced by all vertices on paths originating from ***n***. Each node ***n*** has an associated weight value ***w(n)***. The weight can be computed based on the selected properties of the data. For example, in file directory, the weights of vertices representing file systems are conveying the actual sizes of the corresponding files and folders. The local region ***P(n)*** of node ***n*** is a polygon shaped container. ***P(n)*** contains the drawing of a sub-tree ***T(n)***. The area of ***P(n)*** is calculated proportionally to ***w(n)***.

3.1.3 WEIGHT CALCULATION

Unlike point-based rendering paradigm (Schulz et al., 2011), we still follow the polygonal partitioning approach in the algorithm. The D&C partitioning produces nested polygons whose sizes are proportional to the weight of vertices. The calculation of nodes' weights is independent to the layout algorithms and this process starts before the geometrical partition. We assign a weight *w(v)* to each vertex *v* for the calculation of the local region *R(v)* that relates to the regions of its father and siblings. There are many ways to define the weight of a vertex, and one way to do it is based on the vertex property. This thesis, however, defines the weight of a node simply based on its descendants.

Each Node n_i in tree *T* is associated with a weight w_i and thus we have a set of weights $\{w_1, w_2, \ldots, w_m\}$ associated with the vertex set $\{n_1, n_2, \ldots, n_m\}$ in *T*. The set of nodes weights can be calculated recursively from leaves in the following rules:

If a node n_i is a leaf, its weight is defaulted as $w_i = 1$. If the node n_i has *k* children $\{n_{i,1}, n_{i,2}, \ldots n_{i,j}, \ldots n_{i,k}\}$, its weight is calculated by using formula:

$$w(n_i) = 1 + C\sum_{j=1}^{k} w(n_{i,j})$$

Equation 1: Node weight calculation formula

In the equation1, C is a constant ($0 < C < 1$), and $w_{i,j}$ is the weight assigned to the j^{th} child of vertex n_i. Constant C is a scalar that determines the difference of local regions' sizes based on the number of descendants of these vertexes. In other words, the larger the C's value is, the bigger the difference of local regions *P(n)* of vertices with more descendants and vertices with fewer descendants.

3.1.4 Position of Nodes

After the calculation of local region, we calculate the position of nodes. This section describes the technical detail of how to define the node position inside its polygonal local area.

When within non-self-intersecting closed polygon, we position the label as the centroid of polygon. The vertices are initially numbered in order of their occurrence along the polygon's perimeter by clockwise, and the vertex (x_n, y_n) is assumed to be the same as (x_0, y_0). The centroid of a non-self-intersecting closed polygon defined by n vertices (x_0,y_0), (x_1,y_1), ..., (x_{n-1},y_{n-1}) is the point (C_x, C_y), where

$$C_{\mathrm{x}} = \frac{1}{6A}\sum_{i=0}^{n-1}(x_i + x_{i+1})(x_i\ y_{i+1} - x_{i+1}\ y_i)$$

$$C_{\mathrm{y}} = \frac{1}{6A}\sum_{i=0}^{n-1}(y_i + y_{i+1})(x_i\ y_{i+1} - x_{i+1}\ y_i)$$

Equation 2: Position of nodes calculation formula

and where A is the polygon's signed area,

$$A = \frac{1}{2}\sum_{i=0}^{n-1}(x_i\ y_{i+1} - x_{i+1}\ y_i)$$

Equation 3: Polygon's signed area calculation formula

When within other shaped polygon, we use other solution as below. An example of nodes positioning is illustrated in Figure 3-1. Suppose that the local region $P(n_{i,j})$ of the j^{th} child vertex of n_i is already defined, We calculate the position of the child node $n_{i,j}$. Firstly, the algorithm finds a point Q' in the polygon $P(n_{i,j})$ of node $n_{i,j}$ that the straight line connecting Q and its starting vertex v_s divides the polygon $P(n_{i,j})$ into two halves of the same area.

Specifically, from the first vertex v_1 of *P*, we divide the polygon *P* into several triangles namely $\{v_1v_2v_3, v_1v_3v_4, \ldots, v_1v_{k-1}v_k, \ldots, v_1v_{n-1}v_n\}$. The area of polygon *p* is the sum of total areas of triangles $v_1v_2v_3, v_1v_3v_4, \ldots, v_1v_{k-1}v_k, \ldots, v_1v_{n-1}v_n$. The next step is to find the side-edge in *P* that point *Q* lays on, called v_jv_{j+1}. This process can be executes linearly via the summation of area of these triangles and finishes when the sum of areas of triangles $v_1v_2v_3, v_1v_3v_4, \ldots, v_1v_{j-1}v$, is less than half of the area of *P* and sum of areas of triangles $v_1v_2v_3, v_1v_3v_4, \ldots, v_1v_{j-1}v_j, v_1v_jv_{j+1}$ is greater or equal to half of the area of *P*.

When the side-edge v_jv_{j+1} has been found, the next step is to compute the position of *Q* on v_jv_{j+1}.This position can be anywhere on the segment from the father vertex v_i to point *Q* and it can be adjusted in order to optimize the visualization for each type of applications. We, however, decide to position the node simply at the midpoint of the segment. In the special case, when node $n_{i,j}$ has only one child, it means its local region and the child's local region are same. The system defines the position of the child node $n_{i,j}$ in the segment from v_i to point *Q*.

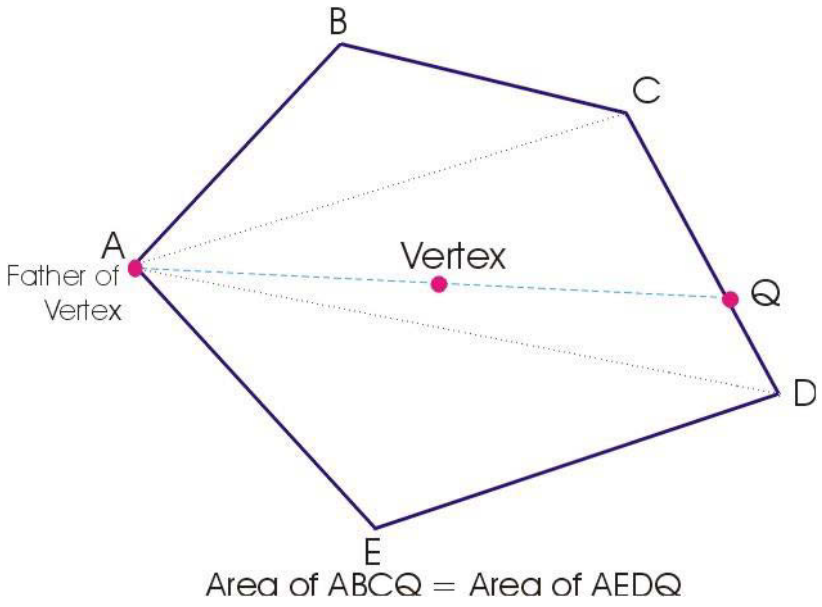


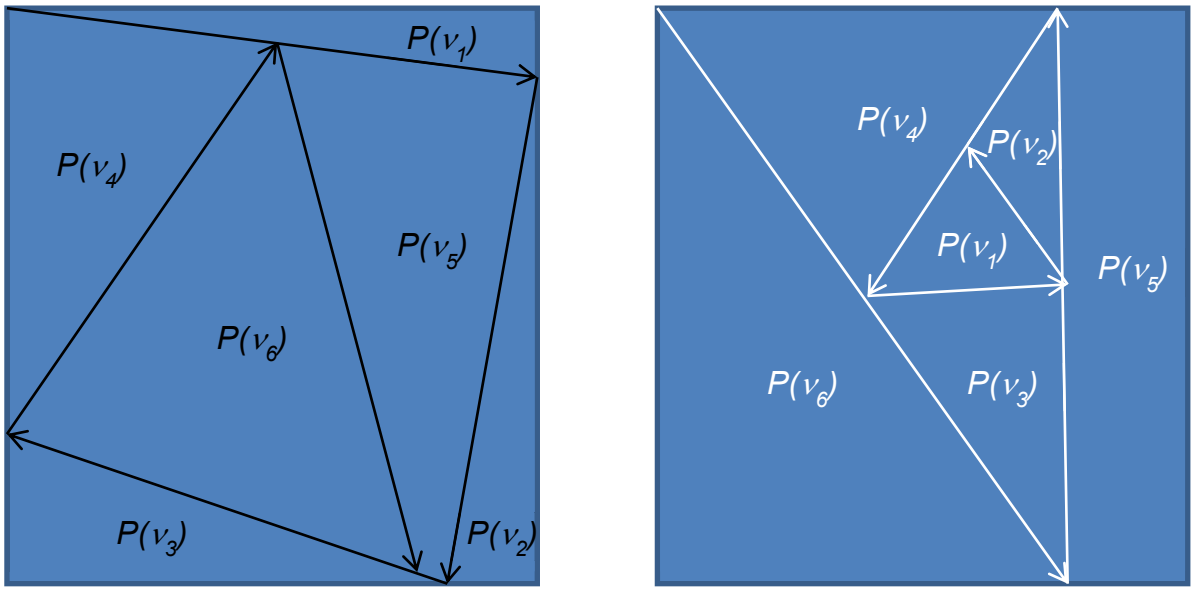
Figure 3-1 Position of nodes illustration: An example of positioning a vertex in its local region

3.1.5 TESSELLATION METHODS

Each node *n* is bounded by a polygon *P(n)*. The partitioning and drawing of a sub-tree *T(n)* is restricted within the area of *P(n)*. A drawing of sub-tree *T(n)* rooted at *n* is calculated based on the properties of *n* and its local region *P(n)*. In partitioning process, the local region *P(r)* for root *r* is defined as the entire polygonal container. From the given geometry, local regions $P(n_1)$, $P(n_2)$, …, $P(n_k)$ of all child nodes $n_1, n_2, \ldots, n_k$ are then defined. Using the enclosure partitioning mantra, the procedure is repeated to all descendant vertices based on the defined local regions of the sub-rooted vertex until leaf vertices are reached. For a node *n*, its region *P(n)* be partitioned for the sub-tree *T(n)*. *T(n)* has *k* children $\{v_1, v_2, \ldots, v_k\}$. The partitioning process is as follows:

- Partition *P(n)* into polygonal local regions $\{P(n_1), P(n_2), \ldots, P(n_k)\}$ for *k* children $\{n_1, n_2, \ldots, n_k\}$.
- Calculate positions of $\{n_1, n_2, \ldots, n_k\}$ as the centre position of $\{P(n_1), P(n_2), \ldots, P(n_k)\}$.
- Repeat the process to all sub-trees $T(n_k)$ in top-down direction
- Stop when all leaf vertices of the *T* are reached

There are two algorithms to partition the $P(v_i)$ into multiple polygonal local regions $P(v_{i+1})$, $P(v_{i+2})$, …, $P(v_{i+n})$ for the children $\{v_{i+1}, v_{i+2}, \ldots, v_{i+n}\}$. Figure 3-2 demonstrates two tessellation approaches, using same data set used in the Figure 1-19. The first level of data set is shown on a given square, which includes 6 sorted vertices $\{v_1, v_2, v_3, v_4, v_5, v_6\}$, whose weights are *{1, 1, 2, 4, 6, 8}* respectively. Figure 3-2 a uses *Linear Partitioning* algorithm and Figure 3-2b uses *Divide and Conquer (D&C)* algorithm in its tessellation process. The details of two methods are explained in Section 3.1.5.1 and Section 3.1.5.2. The two tessellation algorithms are outlined in ***LinearPartition()*** in section 3.1.5.1 and ***D&C Partition(),*** in combination with ***Algorithm Ini FirstPoint(vs, P(N)), Algorithm: Divide(), Algorithm: Conquer(), and Algorithm: AngularResolutionConstraint()*** *in Section 3.2.*



(a) **(b)**

Figure 3-2 Tessellation Illustration of linear and Divide and Conquer methods: algorithms for the first level of the dataset in **Figure 1-19** using a) linear division algorithm (left) and b) divide and conquer algorithm (right)

3.1.5.1 Linear Tessellation process

The first algorithm we employ on polygon tessellation is linear approach. This tessellation process follows the sequence of nodes according to the hierarchy of nodes in the tree structure.

In details, the algorithm constructs node N into a polygon P(N). For N has a list of children with number of k $\{n_1, n_2, \ldots, n_k\}$, the algorithm partitions Polygon P(N) into sub-polygons with number of k, according to their corresponding weight $\{w_1, w_2, \ldots, w_k\}$. The polygon P(N) has n number of side vertices called $\{v_1, v_2, \ldots, v_n\}$ in $\mathbf{R}^2$. The Linear treemap algorithm selects the first point for subdivision and then linearly divide P(N) into $P(n_1)$, $P(n_2)$,…, and $P(n_k)$ according to their hierarchical order. If Let P(N) be the polygon for partitioning, $\{n_1, n_2, \ldots, n_k\}$ is a list of k child nodes of node N with corresponding polygonal boundary $\{P(n_1), P(n_2), \ldots, P(n_k)\}$ and with weight $\{w_1, w_2, \ldots, w_k\}$ respectively. The polygon P(N) has n number of side vertices called $\{v_1, v_2, \ldots, v_n\}$ in $\mathbf{R}^2$. The partitioning process linearly divide P(n) into $P(n_1)$,

$P(n_2)$,…, and $P(n_k)$ according to their hierarchical order. The algorithm has been outlined in **Algorithm- LinearPartition()** below .

Algorithm- LinearPartition()

Input: a polygon P(N) of node N with n side vertices $\{V_1, V_2,\ldots, V_n\}$ in R2 to be partitioned

Output: a list of sub-polygons $\{P(N_1), P(N_2),\ldots P(N_i), \ldots,P(N_k)\}$ corresponding to the child nodes $\{N_1,N_2, N_i,\ldots,N_k\}$ with weights $\{w_1,w_2, \ldots w_i,\ldots,w_k\}$ respectively

1: if k = 1 then

Note: Node N is the leaf node; When Node of N does not have any child

2: $P(N_i) = P(N1) = P(N)$

3: else

4: V_s = **Ini FirstPoint**(P(N))

5: re-arrange the order of side vertices of P(N) starting from V_s

6: end if

7: for each N_i do

8: $$A_i = A \times \frac{w_i}{w}$$

9: $P(N_i)$ = Conquer(P(N), A_i)

10: end for

Note: Refer Ini First Point algorithm in Section 3.2.1.2.

3.1.5.2 D&C TESSELLATION PROCESS

The basic concept for generating polygonal subdivision is to utilize the divide and conquer approach in sub division process. Original divide and conquer is important algorithm design paradigm based on multi-branched recursion. Divide and conquer paradigm decomposes a problem into two or more sub-problems of same or related type, until the sub-problem becomes simple to be solved directly. Similarly, we break down the nodes into two groups according to the original sequence of their orders, with similar weight for each group. In partitioning process, our algorithm divides the polygon into two sub-polygons, representing two sub-groups of nodes. This procedure is executed recursively for all top level nodes within the respective sub-polygons. The same process will follow for each subgroup of nodes. When the recursion ends, a complete partitioning is obtained. The algorithms are detailed in Section 3.2

3.1.5.3 OPTIMAL TESSELLATION APPROACH

The results of our experiments show that the layouts generated by *D&C Partition* algorithm exceeds *Linear Partitioning* algorithm in terms of readability, nodes separations, and angular aspect ratio. Angle aspect ratio adopts the aspect ratio into polygonal and mostly triangular shape graph. Within polygon, angle aspect ratio is defined as the ratio of the largest angle to the smaller angle. When the ratio in triangle is larger or smaller than one in great degree, it means the triangle is very tinny. In this case, we use “low aspect angles”. Most results for large data sets, layouts produced by *Linear Partitioning* algorithm contain a large number of nodes with very low aspect angles. For instance, as result of Linear partitioning algorithms showing in Figure 3-3, a great number of the triangles have narrow angles and densely pack together. It provides an artistic look of the image, but ambiguity of nodes distinctiveness may also mislead visual analysts in task-based scenarios. Using the same data set with 2,400 vertices, the visualization generated by *D&C* partition algorithm in

Figure 3-4 provides a much better quality in legibility for human perception. Therefore, we adopt the *D&C* approach in our development.

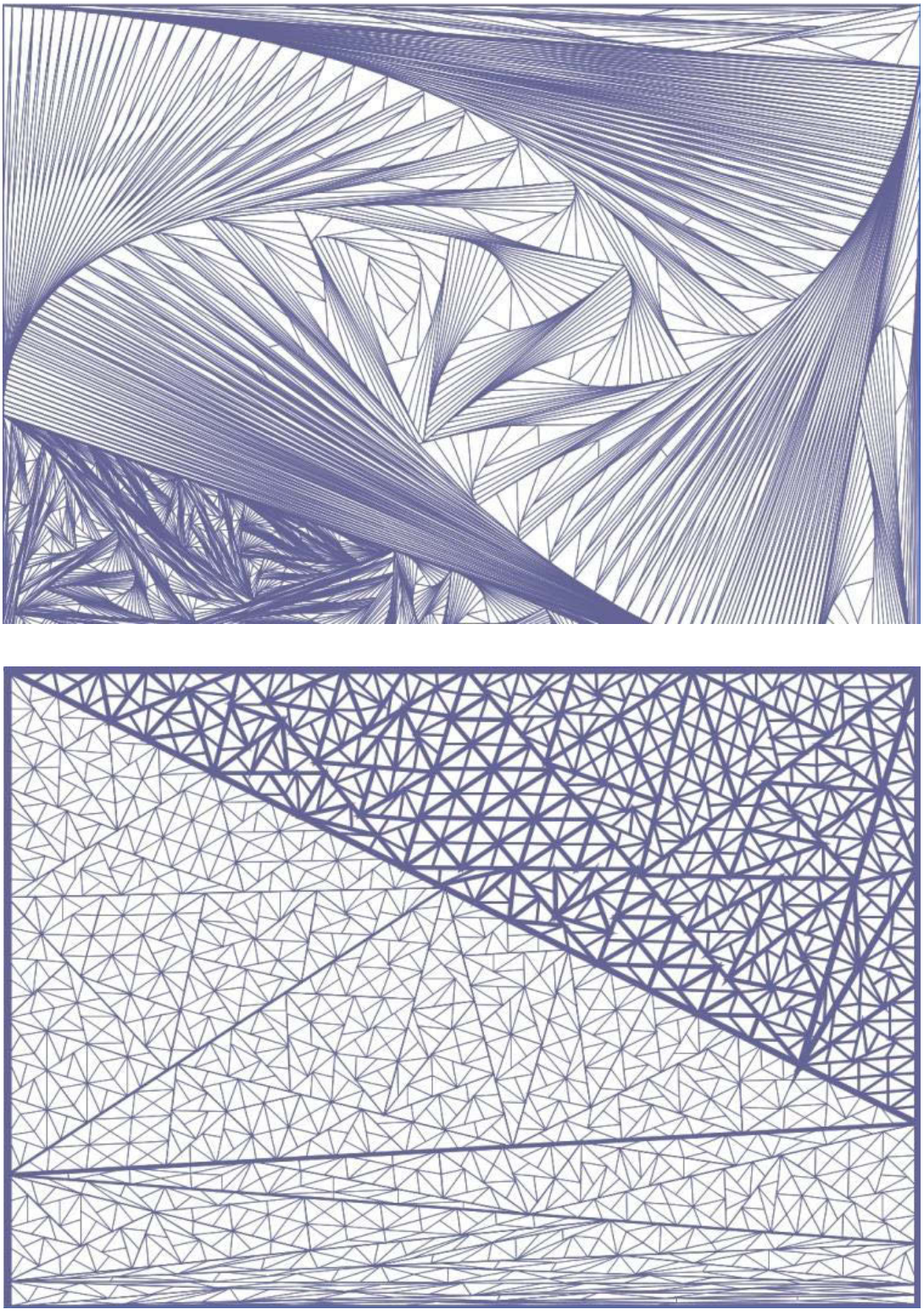


Figure 3-3 Tessellation Methods Comparison 1: visualization of a large data set with 2,400 vertices using the linear partition algorithm (up) and the D&C Partition algorithm at the vertex (down)

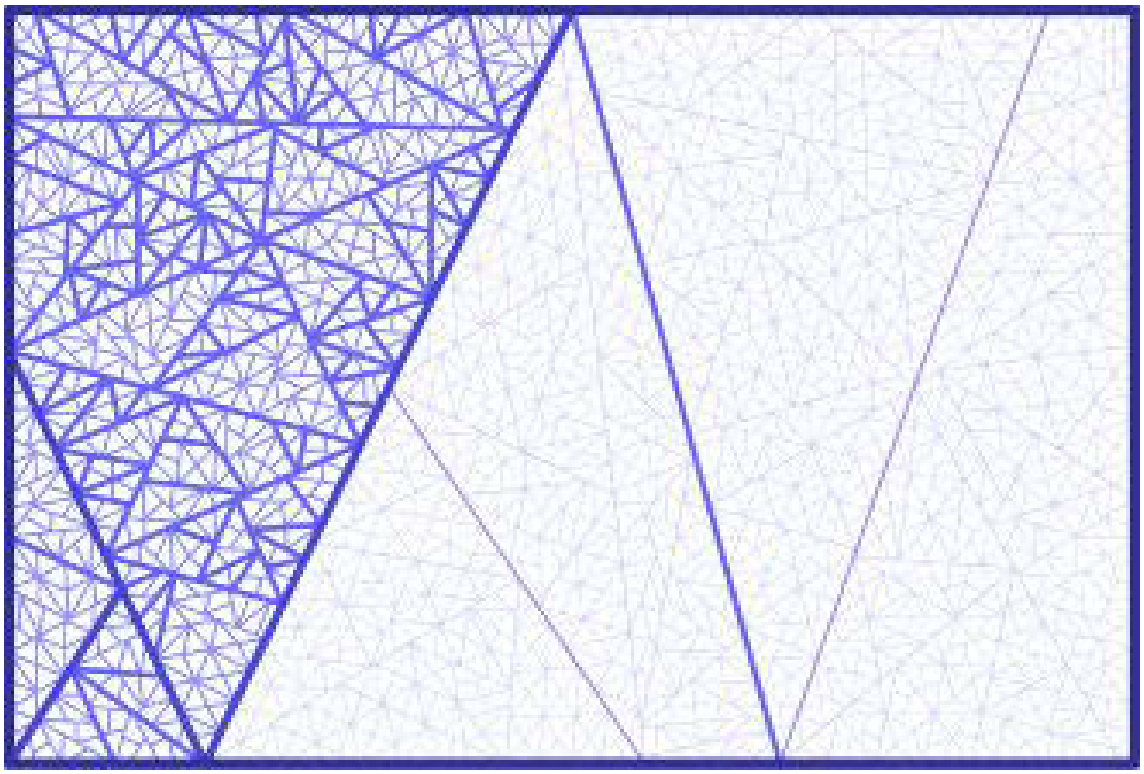


Figure 3-4 Tessellation Methods Comparison 2: A visualization of a large data set with 2,400 vertices same as Figure 3-3 with D&C partitioning at the side.

3.1.5.4 D&C TESSELLATION PROPERTY

Although the boundary of a node is theoretically a polygon, the results from our experiments have shown that most of boundaries after the partitioning process are triangles. In other words, this partitioning process could also be considered as *triangular tessellation*. This section explains D&C partitioning property in lemmas 1, 2 and 3 below.

Lemma 1: The partitioning process of a convex polygon will produce a list of convex polygons.

***Proof*:** suppose that at a particular stage, the *Divide and Conquer (D&C)* partition algorithm recursively divides a convex polygon P into two smaller polygons P_1 and P_2 where $P = P_1 \cup P_2$. P has m side vertices $\{v_1, v_2, .., v_m\}$. Suppose that s_1 has the largest angle and the partitioning start from v_1. Point v_c is the cutting point on a side v_iv_{i+1} where it divides P into P_1 and P_2. The vertices for the polygons P_1 and P_2 are $\{v_1, v_2, .., v_i, v_c\}$ and $\{v_1, v_c, v_{i+1}, .., v_m\}$ respectively. The corresponding angles of the vertices in P, P_1 and P_2 are: $\boldsymbol{\{P.v_1(\vartheta), P.v_2(\vartheta), \ldots, P.v_m(\vartheta)\}}$, $\boldsymbol{\{P_1.v_1(\vartheta), P_1.v_2(\vartheta), \ldots, P_1.v_i(\vartheta), P_1.v_c(\vartheta)\}}$ and $\boldsymbol{\{P_2.v_1(\vartheta), P_2.v_c(\vartheta), \ldots, P_2.v_{i+1}(\vartheta)\}}$. Because P is a convex polygon the angles at all vertices $\boldsymbol{\{P.v_1(\vartheta), P.v_2(\vartheta), \ldots, P.v_m(\vartheta)\}} \leq 180^o$. The vertices $v_2, .., v_m$ are untouched during the partitioning process, thus the angles $\boldsymbol{\{P_1.v_2(\vartheta),\ldots, P_1.v_i(\vartheta)\}}$ in P_1 and $\boldsymbol{\{P_2.v_{i+1}(\vartheta),\ldots, P_2.v_m(\vartheta)\}}$ in P_2 are equalled to $\boldsymbol{\{P.v_2(\vartheta),\ldots, P.v_i(\vartheta)\}}$ and $\boldsymbol{\{P.v_{i+1}(\vartheta),\ldots, P.v_m(\vartheta)\}}$, which are $\leq 180^o$. We now examine the side vertex v_1 and the cutting side vertex v_c. Since v_c is a point on the polygon, (i.e. the segment v_1v_c is inside the polygon), the division on v_1v_c always produces $\boldsymbol{P_1.v_1(\vartheta)}$ and $\boldsymbol{P_2.v_1(\vartheta)}$ that $\boldsymbol{P_1.v_1(\vartheta)} + \boldsymbol{P_2.v_1(\vartheta)} = \boldsymbol{P.v_1(\vartheta)}$. This means both $\boldsymbol{P_1.v_1(\vartheta)}$ and $\boldsymbol{P_2.v_1(\vartheta)}$ are $\leq 180^o$. In addition, because the cutting point v_c lies on the side s_is_{i+1}, $\boldsymbol{P_1.v_c(\vartheta)} + \boldsymbol{P_1.v_c(\vartheta)} = 180^o$, and thus $\boldsymbol{P_1.v_c(\vartheta)} \leq 180^o$ and $\boldsymbol{P_1.v_c(\vartheta)} \leq 180^o$. As a result, all angles in P_1 and P_2 are $\leq 180^o$ or P_1 and P_2 are convex polygons.

Lemma 2: The partitioning process of a concave polygon will produce a list of mixed convex and concave polygons.

Proof: suppose that at a particular stage, D&C partition algorithm recursively divides a convex polygon P into two smaller polygons P_1 and P_2 where $P = P_1 \cup P_2$. P has m vertices $\{v_1, v_2, .., v_m\}$. Suppose that v_1 is a concave whose angle $\geq 180^o$ and the partitioning starts from v_1. The vertices v_2, .., v_m are untouched during the partitioning process, thus the angles $\{\boldsymbol{P_1.v_2(\vartheta)}, .., \boldsymbol{P_1.v_i(\vartheta)}\}$ in P_1 and $\{\boldsymbol{P_2.v_{i+1}(\vartheta)}, .., \boldsymbol{P_2.v_m(\vartheta)}\}$ in P_2 are equalled to $\{\boldsymbol{P.v_2(\vartheta)}, .., \boldsymbol{P.v_i(\vartheta)}\}$ and $\{\boldsymbol{P.v_{i+1}(\vartheta)}, .., \boldsymbol{P.v_m(\vartheta)}\}$. If one or more of the angle $\{\boldsymbol{P_1.v_2(\vartheta)}, .., \boldsymbol{P_1.v_i(\vartheta)}\} \geq 180^o$, than P_1 and P_2 are concave polygons. If none of angle $\{\boldsymbol{P_1.v_2(\vartheta)}, .., \boldsymbol{P_1.v_i(\vartheta)}\} \geq 180^o$, since v_c is a point on the polygon, (i.e. the segment v_1v_c is inside the polygon), the division of this angle always produces $\boldsymbol{P_1.v_1(\vartheta)}$ and $\boldsymbol{P_1.v_1(\vartheta)}$ that $\boldsymbol{P_1.v_1(\vartheta)} + \boldsymbol{P_1.v_1(\vartheta)} = \boldsymbol{P.v_1(\vartheta)}$. This means both $\boldsymbol{P_1.v_1(\vartheta)}$ and $\boldsymbol{P_2.v_1(\vartheta)} \leq 180^o$, or either $\boldsymbol{P_1.v_1(\vartheta)}$ or $\boldsymbol{P_2.v_1(\vartheta)} \leq 180^o$. Finally, because the cutting point v_c lies on the side v_iv_{i+1}, $\boldsymbol{P_1.v_c(\vartheta)} + \boldsymbol{P_2.v_c(\vartheta)} = 180^o$, and thus $\boldsymbol{P_1.v_c(\vartheta)} \leq 180^o$ and $\boldsymbol{P_2.v_c(\vartheta)} \leq 180^o$. This has proved that P_1 and P_2 can be either convex or concave.

Lemma 3: The partitioning process of a convex polygon will produce polygons with fewer vertices than the partitioned polygon.

Proof: Suppose that at a particular stage, *D&C partition* algorithm recursively divides a convex polygon P into two smaller polygons P_1 and P_2 where $P = P_1 \cup P_2$. P has m vertices $\{v_1, v_2, .., v_m\}$. Call v_c is the cutting point on a side v_iv_{i+1} where it divides P into P_1 and P_2. As a result, the vertices for the polygons P_1 and P_2 are $\{v_1, v_2, .., v_i, v_c\}$ and $\{v_1, v_c, v_{i+1}, .., v_m\}$ respectively. The total number of vertices in both P_1 and P_2 are $m + 3$ and the numbers of vertices in P_1 and P_2 are respectively: $i + 1$ and $m - i + 2$. Because the cutting point v_c lies on a side of the polygon, it has to lie on one of the segments from second vertex to the last vertex, i.e. $i \geq 2$ and $i \leq m - 1$. Therefore, $m - i + 2 \leq m$ and $i + 1 \leq m$, corresponding to P_1 and P_2 have fewer vertices than P.

Remark: Because the property of Lemma 3, the partitioning process divides polygons into multiple polygons with fewer number of vertices and that eventually it produces triangles. The partitioning of a triangle also produces multiple triangles. These properties are illustrated in the Figure 3-3.

SECTION 3.2 IMPLEMENTATION ALGORITHMS

The Tangram tessellation follows Divide and conquers partitioning method descried in Section 3.1.5.2. According to the graphical properties of Divide and conquer partitioning method, the partitioning results generated mainly includes triangular titling. This method in the first stage creates D&C Triangular approach, which relax the partitioning rules. To achieve optimal aesthetics value of representation, we add minimum angular resolution constraint in D&C Triangular approach. In the second stage, we further attach angular condition to D&C Triangular Approach. The representation generates mainly polygons rather than triangular. This approach is called Angular polygon approach, as we still do not follow horizontal and vertical partitioning direction. In third stage, D&C Rectangular Approach is created, when we restrict Angular Polygon approach with 90 degree.

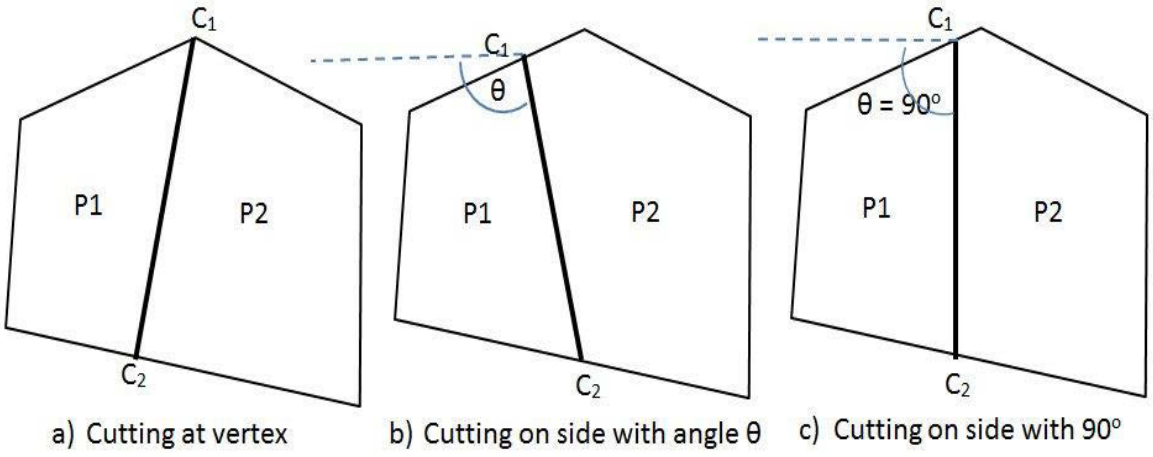


Figure 3-5 First Cutting Illustration of implementation algorithms: the algorithms for cutting a polygon into two sub polygons.

The approaches are explained with partitioning process illustration here. Technically, to partition *P(n)* into local regions $P(n_1)$, $P(n_2)$, …,$P(n_k)$ for *k* children $\{n_1, n_2, \ldots, n_k\}$, we first divide $\{n_1, n_2, \ldots, n_k\}$ into two sub-lists of nodes with similar total weights. Then we select the initial point. The division can either starts at the vertex V_s or at a point V_c on the side starting at the vertex Vx. We then divide *P(n)* into two sub-regions for the two list of vertices with algorithms. There are three ways to divide a polygon into two small polygons at each step for the *D&C* partitioning. Following starting point, we can divide P(n) at initial point without partition angle constrain (Figure 3-5a). Alternatively, we partition P(n) on a side with a

specified angle (Figure 3-5b). For a special case, we restrict the partition angle to 90 degree only (Figure 3-5c). Figure 3-5 illustrates the division of a polygon into two polygons P_1 and P_2 for the two lists of nodes whose weights are 3 and 5 respectively, following three ways. After dividing, the following partitioning executes the dividing ways in each step. These partitioning approaches create D&C triangular algorithm, Angular Polygonal algorithm and D&C Rectangular algorithm. Figure 3-6 shows the final layouts after partitioning using three approach for the small data set, including 6 vertices *{1.0, 1.1, 1.2, 1.3, 1.4, 1.5},* whose weights are *{*3, 4, 1, 2, 3, 1*}* respectively.

Specifically we introduce D&C Triangular Approach in Section 3.2.1, Angular Polygonal Treemap in Section 3.2.3 and D&C Rectangular Approach in Section 3.2.4. Each section of these three approaches includes the illustration of partitioning process with examples and explanation of algorithm and experimental results.

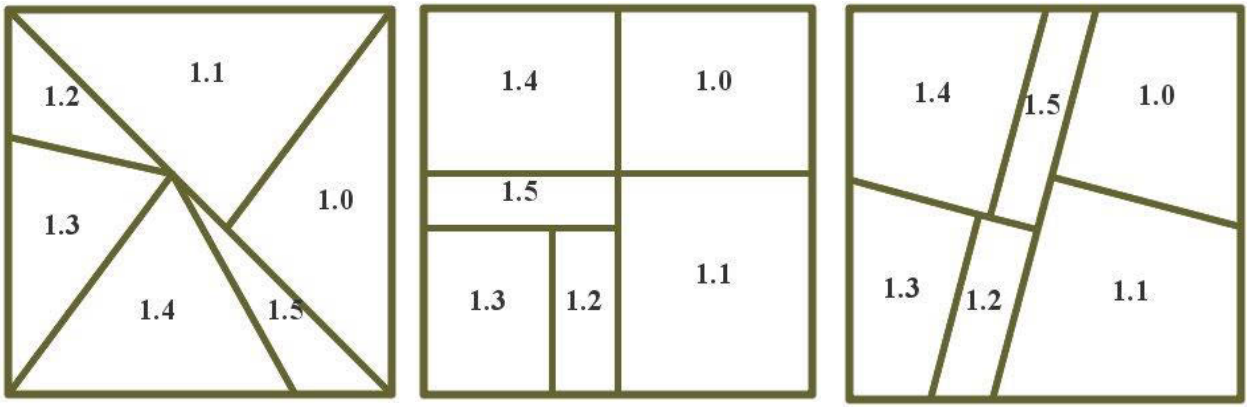


Figure 3-6 An illustration for a small data using implementation algorithms: Three partitioning output are for 6 child nodes {1.0, 1.1, 1.2, 1.3, 1.4, 1.5} with weights {3, 4, 1, 2, 3, 1} using a) partition at a vertex point, b) partition on a side with 750 and c) partition on a side with 900.

3.2.1 D&C Triangular Approach

D&C Triangular Approach illustrates the partitioning process with an example in Section 3.2.1.1 and explains the algorithms and sub-algorithms in Section 3.2.1.2 and demonstrates the experimental results in Section 3.2.1.3.

3.2.1.1 PARTITIONING PROCESS ILLUSTRATION

We first describe Divide and Conquer Tessellation using an example. Suppose that we have a square and we need to divide a square into 6 sub-regions for 6 nodes *{1.0, 1.1, 1.2, 1.3, 1.4, 1.5}*, whose weights are *{3, 4, 1, 2, 3, 1}* respectively. Starting vertex for partitioning is at top-left corner of the square. The first step is to divide the list of nodes into two sub-lists of nodes with similar total weights, i.e. *{1.0, 1.1}* and *{1.2, 1.3, 1.4, 1.5}* whose weights are 7 and 7 respectively. We next divide the square into two polygons at the vertex or on the side respectively. The area of a divided polygon is calculated in corresponding to the ratio of weight of the sub-list of vertices and the total weight of all vertices. Next, for each sub-list of vertices, we repeat the same process with the partitioned polygon. The process finishes when all the leaf vertices are reached. Figure 3-7 illustrates the specific partitioning process of the example.

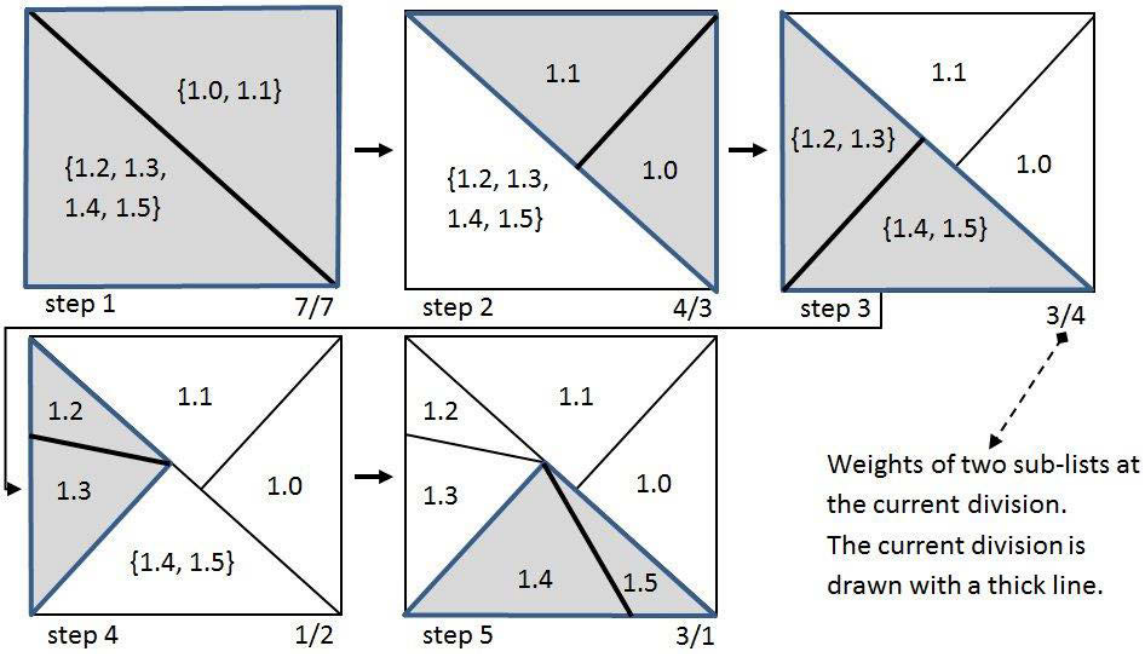


Figure 3-7 Subdivision partitioning process using D&C Triangular algorithm

3.2.1.2 D&C TRIANGULAR ALGORITHMS

D&C Partition algorithm follows recursive structure of the original tree. The algorithm first divides the set of data in same level into two sets of data. The two separate groups of data nodes share the similar weight to each other. The tessellation enables Sub-area sizes corresponding to the size of the nodes in the hierarchy. This algorithm is applied to the subdivision in each recursion step. It contains three sub-algorithms, ***Initialize first point, divide and conquer***. To initialize first side vertex for the starting point, we need to confirm the property of the polygon. If polygon is convex, we select the vertex whose angle is the largest among the polygon. If polygon is concave, we start with the first concave angle whose value is negative. Divide follows recursive structure of the original tree map layout. The Divide Algorithm divides the nodes into two sets of nodes, whose desired weights are close to the half of total weight. The algorithm calculates the weight of the first set Wg1 and the second set Wg2 of data nodes. ***Conquer*** constructs the polygon into two polygons proportionally according the Weight Wg1 & Wg2. To optimize the criteria, the algorithm can be modified in reference to the given constraint, if suitable. The tessellation algorithm is outlined by means of ***Algorithm D&C Partition***, along with sub-algorithm of Initialize first point ***Algorithm Ini FirstPoint(vs, P(N))***, sub-algorithm of Divide ***Divide()***,sub-algorithm of Conquer, ***Conquer()***. To achieve optimal results, the algorithm should also include angular resolution constraint in Section 3.2.2.

Algorithm: Divide and Conquer Triangular Approach

Input: Bounded plane S in R^2; a polygon $P(N)$ with n side vertices $\{v_1, v_2, \ldots, v_n\}$ in S and with weights $\{w_1, w_2, \ldots, w_n\}$

Output: subdivision of P(N) into sub-polygons $\{P(v_1), P(v_2), \ldots, P(v_n)\}$ in S

Algorithm ***D&C Partition ()***

1: Initialize a set of n vertex $V := \{ v_1, v_2, \ldots, v_i, \ldots, v_j, \ldots v_n \}$ With $v_i \in S$, $v_i \neq v_j$

2: Initialize a set of n weights $W := \{w_1, w_2, \ldots, w_i, \ldots w_j, \ldots w_n\}$

3: Initialize a data structure for the D&C Partition

4: **Repeat**

4: *ini First Point* (v_s, $P(N)$)

5: Construct polygons with side vertex

Note: arrange side vertices of P(v) with $\{v_s, v_{s+1}, ,..,v_i,..v_j,\ldots v_{n,\ldots} v_{s-1}\}$

6: *Divide* ($P(N)$)

Note: divide $\{P(v_1),P(v_2)\ldots,P(v_n)\}$ into two groups of polygons $P_1 = \{P(v_1)\ U\ P(v_2)\ U\ \ldots\ U\ P(v_s)\}$ & $P_2 = \{P(v_{s+1})\ U\ P(v_{s+2})\ U\ \ldots\ U\ P(v_n)\}$ with similar weight, $w_1(P_1) \approx w_2(P_2)$

8: *Conquer* ($P(N)$)

9: Option: Angular Resolution Constraint ()

10: *D& C Partition* (P_s)

11: *D&C Partition* (P_r)

Note: if with option of Angular Resolution Constraint (), replace P_s with P_c

Algorithm - Initialize first point

Description: To initialize first side vertex for the starting point, we need to confirm the property of the polygon. If polygon is convex, we select the vertex whose angle is the largest among the polygon. If polygon is concave, we start with the first concave angle whose value is negative.

Summary:

if P(N) is convex, return the largest angle in P(N)

if P(N) is concave, return the first concave angle in P(N)

Algorithm *Ini FirstPoint*(v_s $P(N)$)

Input: a polygon P(N) with n side vertices $\{v_1, v_2, ..., v_i, ... v_j, ..., v_n\}$ in R^2 and with angles $\{\theta_1, \theta_2, \ldots, \theta_i, \ldots, \theta_j, \ldots, \theta_n\}$

Output: Initialize first point v_s

1: Assign $v_s = v_1$

2: Initialize $\theta_s = \theta_1$

2: crossProd = crossProduct(v_n, v_1, v_2)

3: if crossProd >= 0 then

4: numPositive++;

5: else

6: numNegative++;

7: end if

8: **while** i ∈ [1,n]

9: if $\theta_i > \theta_s$ then

10: $v_s = v_i$

11: end if

12: crossProd = crossProduct(v_{i-1}, v_i, v_{i+1})

13: if crossProd >= 0 then

14: if numPositive = 0 then

15: $v_{pos} = v_i$;

Note: Assign v_i to the first positive side vertex v_{pos}

16: end if

17: numPositive++;

18: else

19: if numNegative = 0 then

20: $v_{neg} = v_i$;

Note: Assign v_i to the first negative side vertex v_{neg}

21: end if

22: numNegative++;

23: end if

24: if numNegative > 0 && numPositive > 0 then

Note: It is a case of a concave polygon

25: if numPositive >= numNegative then

26: $v_s = v_{pos}$

27: else

28: $v_s = v_{neg}$

29: end if

30: end if

31: end while

32: return v_s

Algorithm: Divide

Description: The Algorithm sorts the nodes on the same level of the data structure in ascending sequence, according to the weight value of the nodes. Then the algorithm divides the nodes into two sets of nodes, whose desired weights are close to the half of total weight. The algorithm calculates the weight of first set and second set of data nodes.

Algorithm: ***Divide()***

Input: a set of data nodes $\{N_1, N_2, .., N_i, \ldots, N_j, \ldots N_n\}$ with Weight value w_i;

Output: two sets of data nodes $\{N_1, N_2, \ldots, N_t\}$ with weight W_{g1}, and $\{N_{t+1}, \ldots, N_{n-1}, \ldots N_n\}$ with weight W_{g2}

1: Initialize $t = 1$

Note: t is number of nodes contained in the first group of data set

2: Initialize W = $w_1+w_2,..,+ w_n$

2: **while** $t \in [1,n]$

3: $W_{g1} = w_1+w_2,..,+ w_t$

4: $W_{g2} = W - W_{g1}$

5: if $W_{g1} < W_{g2}$ then

6: t++

7: **end if**

8: **end while**

9: **return** t

10: **return** W_{g1}

11: **return** W_{g2}

Algorithm: Conquer

Description: The Algorithm constructs the polygon into two polygons proportionally according the Weight W_{g1} & W_{g2}. The algorithm uses the first point as starting and next two vertex after first point to form the first triangular and calculate the area size. Then later on adjust the end point according to the desired area size.

Algorithm: ***Conquer()***

Input: a polygon P(N) of node N with n side vertices $\{v_1, v_2, \ldots, v_t, \ldots, v_j, \ldots v_n\}$ in S to be partitioned ; two sets of data nodes $\{N_1, N_2, \ldots, N_t\}$ with weight W_{g1}, and $\{N_{t+1}, \ldots, N_{n-1}, N_n\}$ with weight W_{g2};

Output: two sub polygons $P_s(N)$, and $P_r(N)$ with area of A_s and A_r and P(v) = $P_s(N)$U $P_r(N)$

1: Initialize v_t as the temporary ending vertex and v_s' as the end point

2: Initialize A as the total area size of Polygon P(N) and $A_t = Area(v, v_t, v_{t+1})$ as current processing polygon area size $P_t(N)$ and A_s as the first group sub-Polygon $P_s(N)$

3: Initialize d as the desirable line distance and $\ell(v_i, v_{i+1})$ as the vertex distance between V_i, V_{i+1}

4: Initialize $t = 1$

5: **while** $t \in [1, n]$

6: $A_t = Area\ (v_s, v_t, v_{t+1})$

7: **if** $A_t/A < W_{g1}/W$ **then**

8: t++

9: **end if**

10: **end while**

11: **return** t

Note: End point lies on Line (V_t, V_{t+1})

12: $v_s' = v_t + X*d$

Note: Assign V_s' on the Line (V_t, V_{t+1}) with distance from V

13: Initialize X = 1

14: **while** $X \in [1, \ell(v_t, v_{t+1})/d]$

15: $A_t = Area\ (v_s, v_t, v_s')$

16: **if** $A_t/A < W_{g1}/W$ **then**

17: X++

18: **end if**

19: **end while**

20: **Return** X

21: **Return** V_s'

22: **Return** A_t

23: $A_s = At$

24: $Ar = A - A_t$

25: Construct polygon with side vertices $(v_s, v_{s+1}, \ldots, v_{s+i}, v_s')$

26: Construct polygon with side vertices $(v_s', v'_{s+1}, \ldots, v_{n-1}, v_n)$

3.2.1.3 D&C Triangular Algorithm Experiments Results

We apply D&C Triangular Algorithm in convex and concave container to demonstrate the experiments results as below. The datasets used are generated randomly with two attributes, which are the total number of hierarchal levels and the total number of nodes. The datasets can be uniform or non-uniform. For experiments on convex polygonal display areas, Figure 3-8 illustrates the partitioning results in a hexagon for a large data set with approximately 16,600 vertices and 10 hierarchical levels. Figure 3-11 is an example of the layout for an octagon container visualizing a large data set with approximately 122,000 vertices and 8 hierarchical levels. Extending visualization into convex containers, a shape of "house look" in Figure 3-9 uses a data set with approximately 12,000 vertices and 15 levels, and a shape of "book look" in Figure 3-10 uses data set with approximately 81,000 vertices and 7 levels.

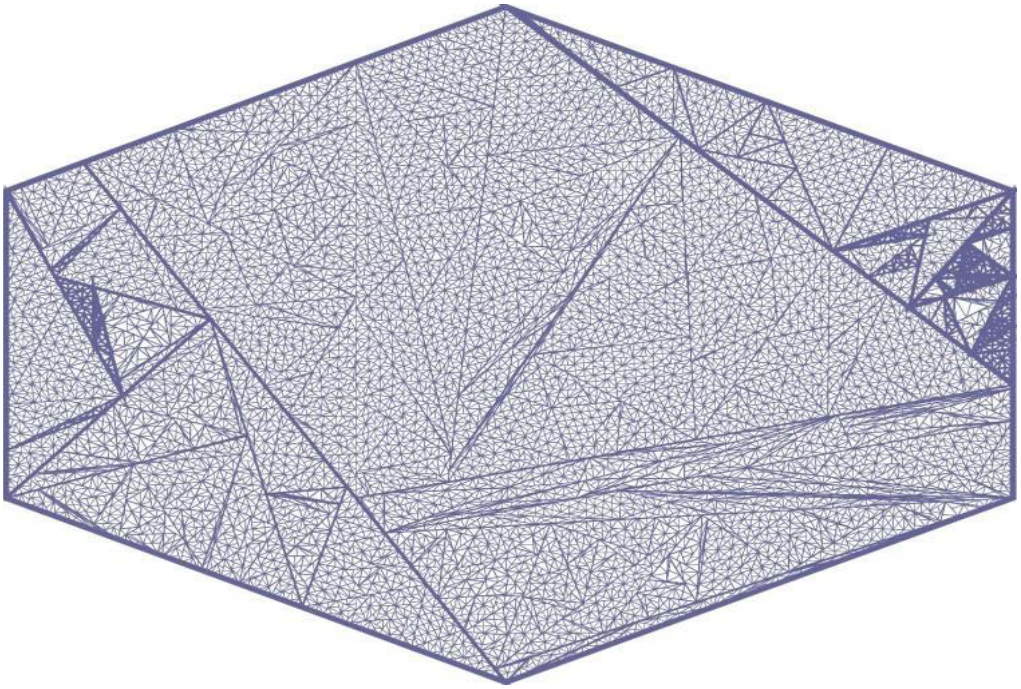


Figure 3-8 A visualization using D&C Triangular algorithm in a hexagon (a data set with approximately 16,600 vertices and 10 levels).

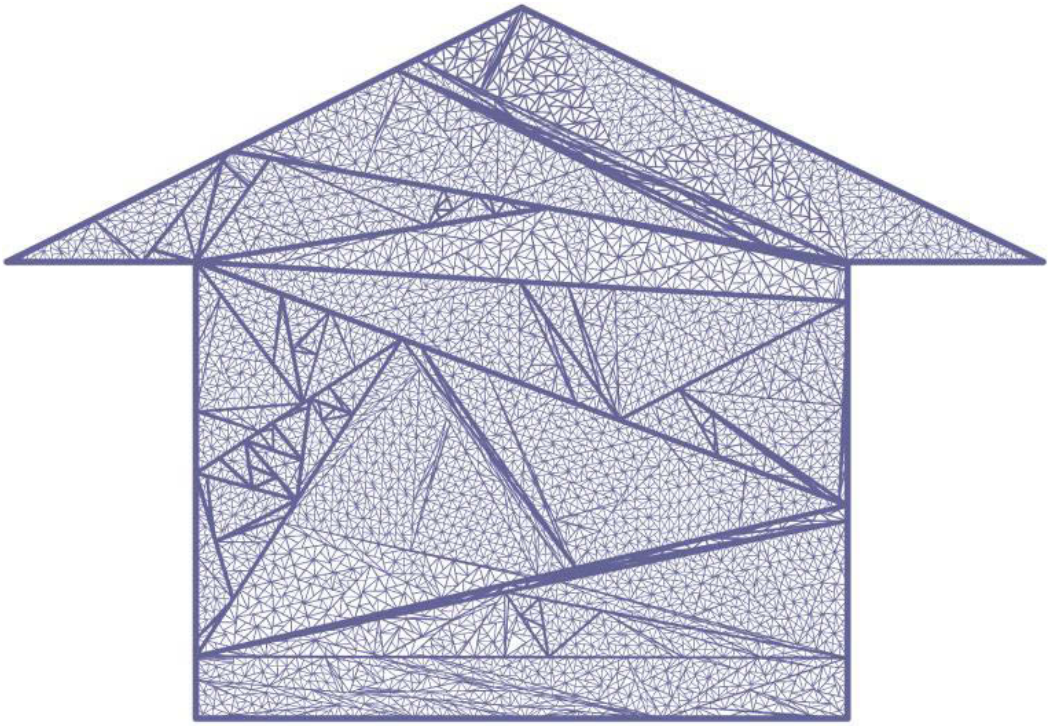


Figure 3-9 A visualization using D&C Triangular algorithm in a concave polygon (data set with approximately 12,000 vertices and 15 levels)

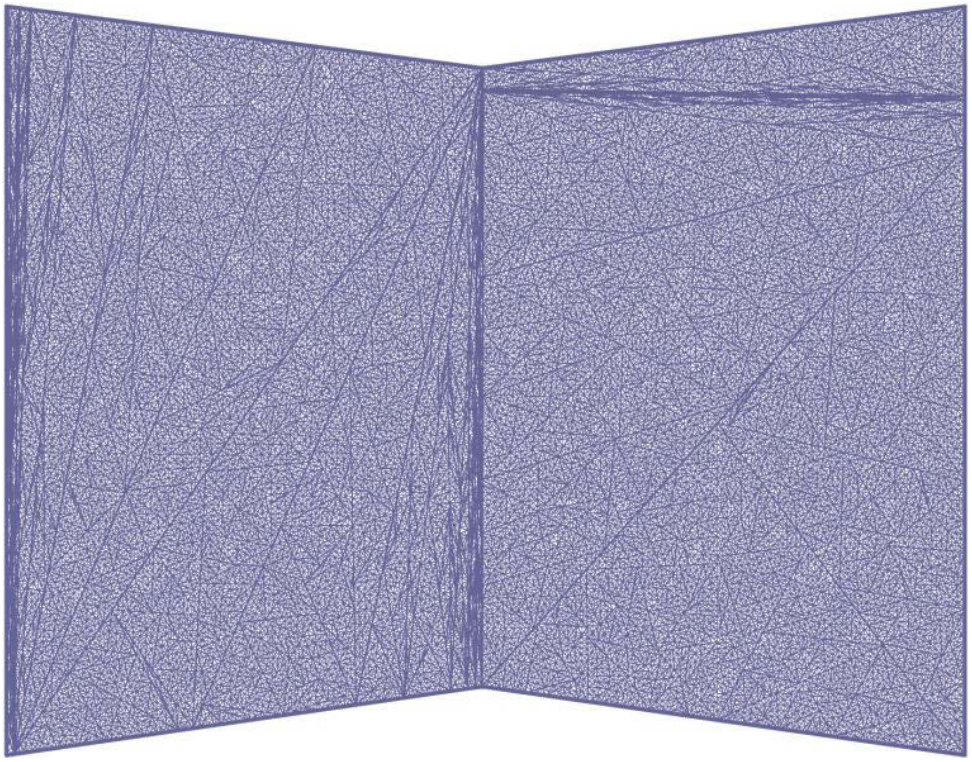


Figure 3-10 A visualization using D&C Triangular algorithm in a concave polygon (a data set with approximately 81,000 vertices and 7 levels).

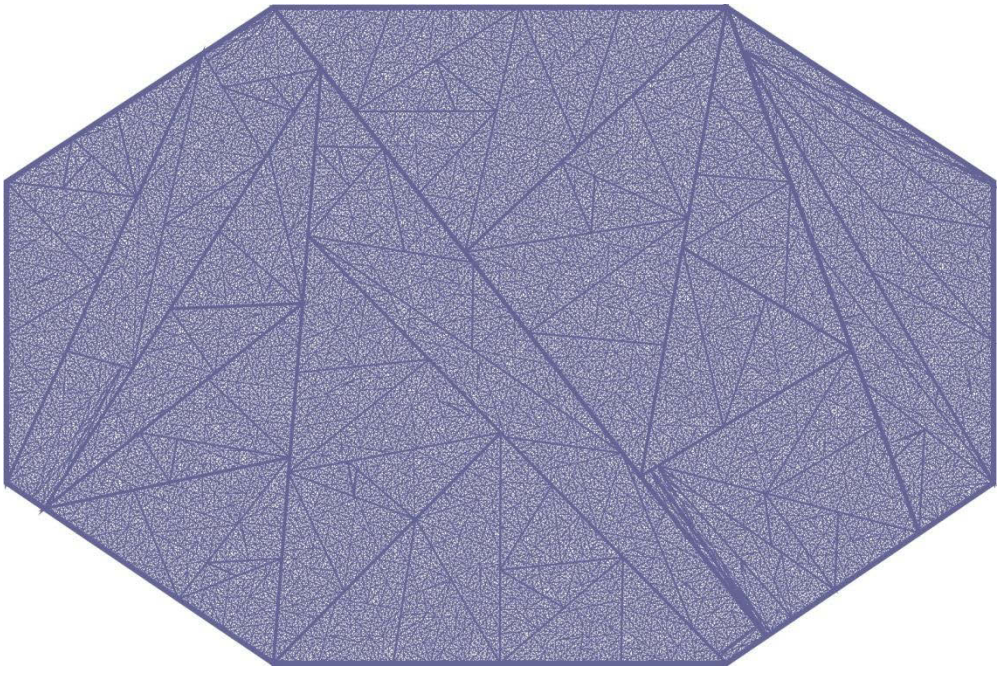


Figure 3-11 A visualization using D&C Triangular algorithm in an octagon (a data set with approximately 122,000 vertices and 8 levels).

3.2.2 D&C Triangular Approach with Angular Resolution Constraint

According to the graph drawing aesthetics rules, angular resolution is one of the significant attribute based on theoretical and empirical evidence. Study has found that the angles smaller than 15 degree make edges much more difficult to distinguish (Battista et al, 1999). Although the above algorithm generally produces good quality of aspect-ratio of angles, it does not guarantee that angles in polygons satisfy the angular Resolution constraint. To improve visibility, the solution is to maximize the angular resolution (Huang et al, 2009). We include the angular Resolution constraint process in algorithm to ensure high quality and aesthetics of representation results. We adopt the study in Huang's work (2009) and set the minimum angular Resolution constraint as 15 degree. This section explains the algorithms and sub-algorithms in Section 3.2.2.1 and demonstrates the experimental results and compare with D&C Triangular Approach in Chapter and evaluates the visualization results in Section 3.2.2.2 and Section 3.2.2.3.

3.2.2.1 Angular Resolution Constraint Algorithms

The algorithm examines the angles of the previous results first. If the angle doesn't satisfy the minimum angle constraint, the algorithm locates the centre point on the longest side of Polygon and replaces the First point to it. The algorithm follows the Conquer algorithm to divide the polygon from the new initial point. After the subdivision at the new initial point, the algorithm re-test the ϑ_c or ϑ_c' If ϑ_c or ϑ_c' is smaller than angle Resolution constraint. The algorithm modifies the replaced first point to increase ϑ_c or ϑ_c' the angular resolution. This testing process repeats till the angular resolution requirement is met.

Algorithm: Angular Resolution Constraint()

Input: a polygon P(N) of node N with n side vertices $\{v_1, v_2, \ldots, v_i, \ldots, v_j, \ldots v_n\}$ in S to be partitioned

Output: two sub polygons $P_c(N)$, and $P_r(N)$ with area of A_c and A_r and P(v) = $P_c(N)$U $P_r(N)$. The angles in $P_c(N)$, and $P_r(N)$ are greater than minimum angle constraint

Algorithm: ***AngularResolutionConstraint()***

1: Initialize θ_{min} = 15 degree

Note: θ_{min} is minimum angle constraint

2: Initialize d as the desirable distance for angular constrain on Line (V_L, V_{L+1}) as the longest side $\ell(v_{L-1}, v_L)$

3: Compute θ_s and θ_s'

Note: θ_s and θ_s' are the angles generated by the first subdivision at v_s

4: If $\theta_s < \theta_{min}$ or $\theta_s' < \theta_{min}$

5: Replace V_s to the centre point V_c at the longest side $\ell(v_{e-1}, v_e)$ on P

6: Divide (P(N))

7. Re-test θ_s and θ_s'

8: **While** $\theta_s < \theta_{min}$ or $\theta_{s'} < \theta_{min}$

9: $V_c' = V_t \pm X * d$

Note: $X \in \lceil 1, L(V_t, V_{t+1})/d \rceil$

Note: To increase θ_s or θ_s' angular resolution , re-assign V_s' on the Line (V_L, V_{L+1}) with distance d from V_L or V_{L+1}

9: **end while**

10: **Repeat**

3.2.2.2 Angular Resolution Constraint Experiment Results

We apply D&C Triangular Algorithm with angular resolution constraint to convex and concave container same as the experiment environment in Section 3.2.1.3 to demonstrate the experiments results as below. Based on the improved algorithms with angular Resolution constraints, further experiments are implemented again in previous containers, Hexagon (Figure 3-8) and Concave-House (Figure 3-9) Concave-Book (Figure 3-10).

During the experiments, we also noticed this process also increases the computational time compared to the original algorithm. The computational time for generating the layout (excluding file reading and drawing processes) is less than a second for even very large data sets with hundred thousands of vertices. The computational times for generating the layouts at Figure 3-8 to Figure 3-11 are 29, 87, 93, 271 and 389 milliseconds respectively. In experiments showing Figure 3-12 to

Figure 3-14, the computational time was 161 milliseconds, 131 milliseconds and 791 milliseconds respectively. Although adding the constraint increases the partitioning process, its computation time is still comparable to the original algorithm (Table 5-1). The computation complexity is discussed in Chapter 5 in details.

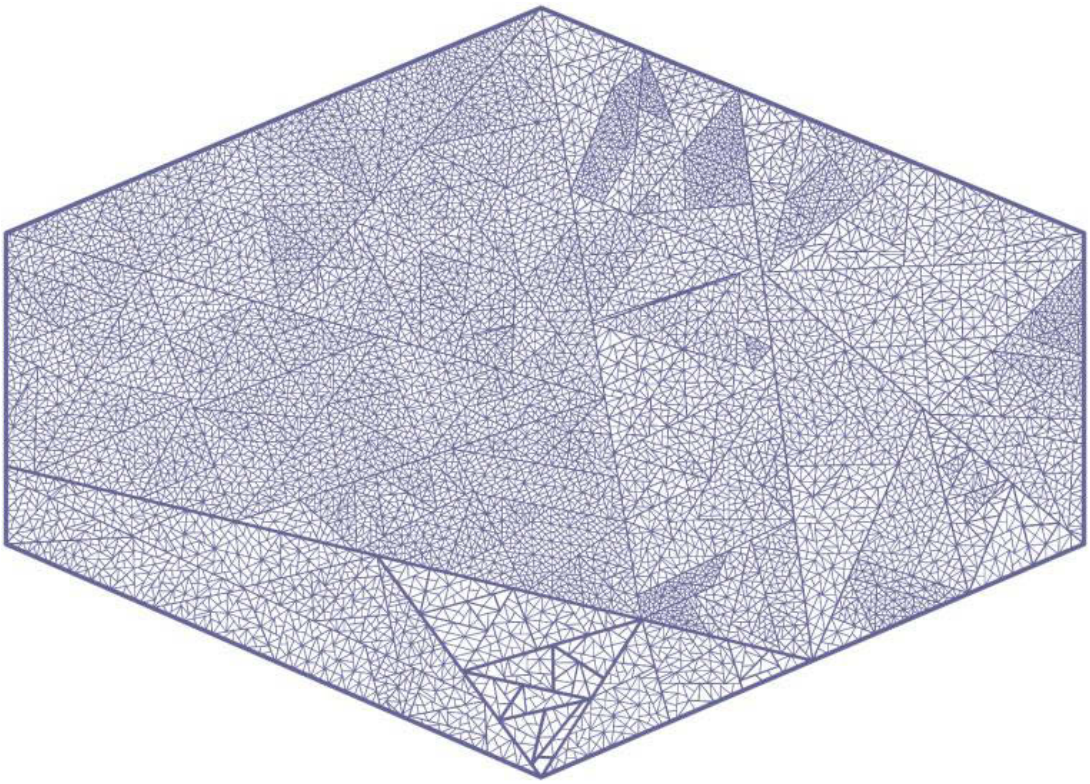


Figure 3-12 A visualization using the D&C Triangular algorithm with angular resolution constraint in a hexagon (the same data set and container as Figure 3-8)

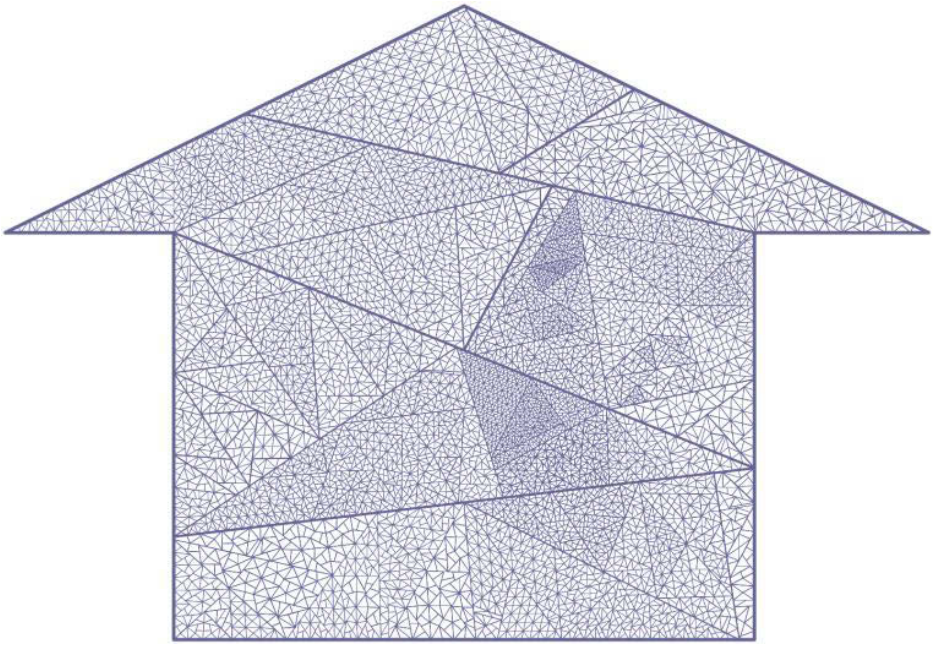


Figure 3-13 A visualization using the D&C Triangular algorithm with angular resolution constraint in a concave polygon (the same data set and container as Figure 3-9).

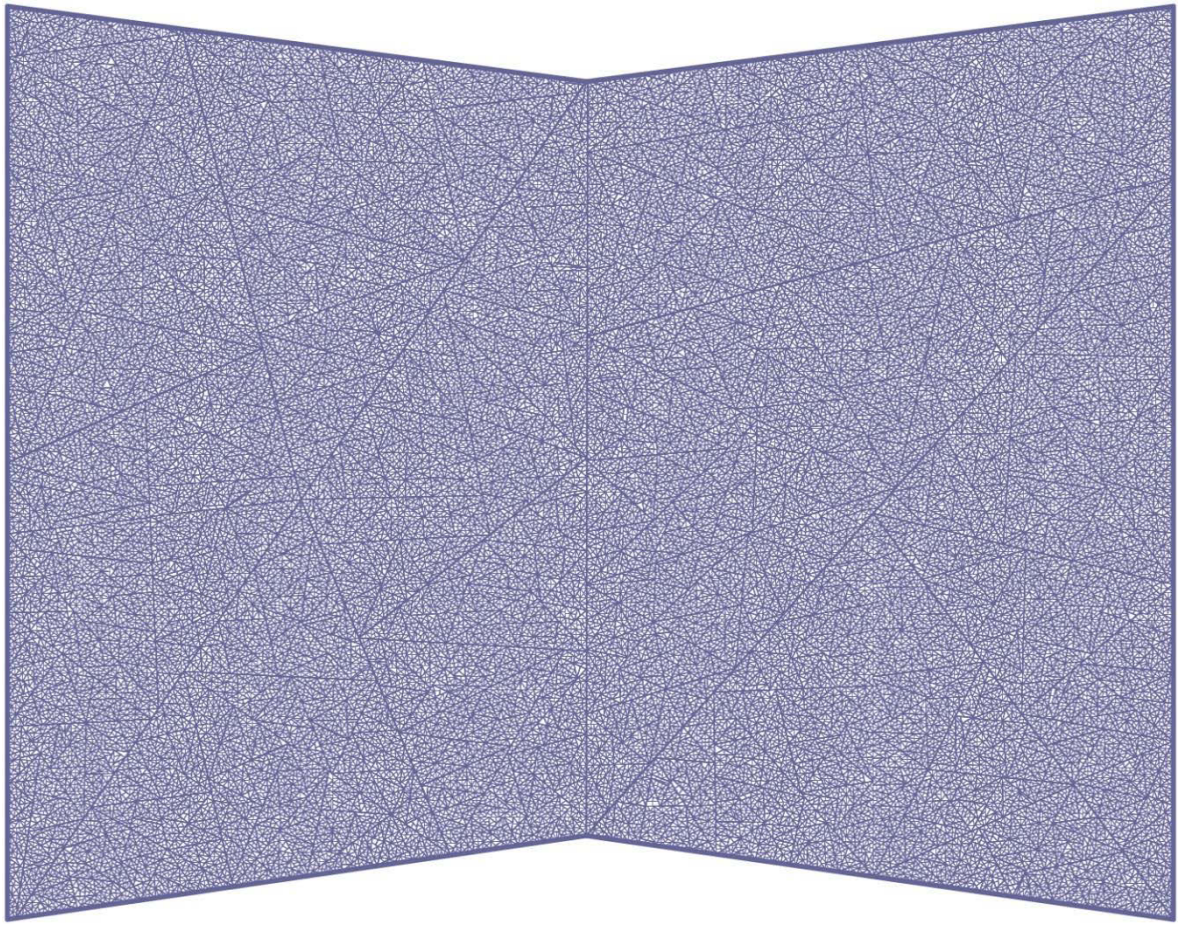
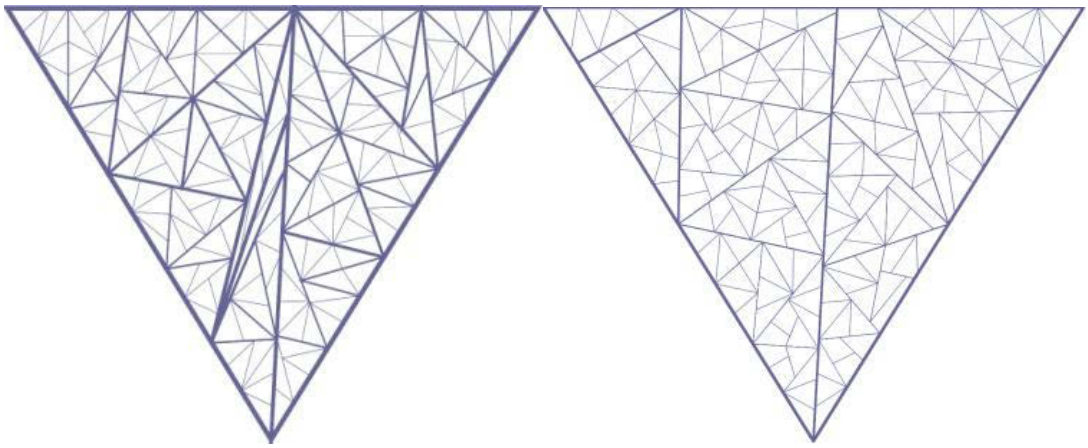


Figure 3-14 A visualization using the D&C Triangular algorithm with angular resolution constraint in a concave polygon (the same data set and container as Figure 3-10).

3.2.2.3 Angular Resolution Constraint Results evaluation

Results of all experiments have been improved significantly in the angle-aspect ratio and perception of data property and hierarchical information. Figure 3-15 demonstrates visually the improvement at Figure 3-15 b with the angular Resolution constraint in comparison with Figure 3-15 a without angular Resolution constraint. It shows that there are no thin angles, and thus it offers a better visual balance of symmetries and densities in the visual representation of a data set. We visualize a larger dataset of 16,600 nodes and 10 levels in a hexagon to compare. From the experimental results, we can also see the improvement visually. To demonstrate the improvements, this section displays a comparison image (Figure 3-17) of the experimental results using *D&C* Triangular algorithm with/without angular Resolution constraint on different data sets and containers, generated in Section 3.2.



a b

Figure 3-15 The partitioning of a data set with 272 vertices using a) D&C Triangular algorithm and b) D&C Triangular algorithm with angular resolution constraint.

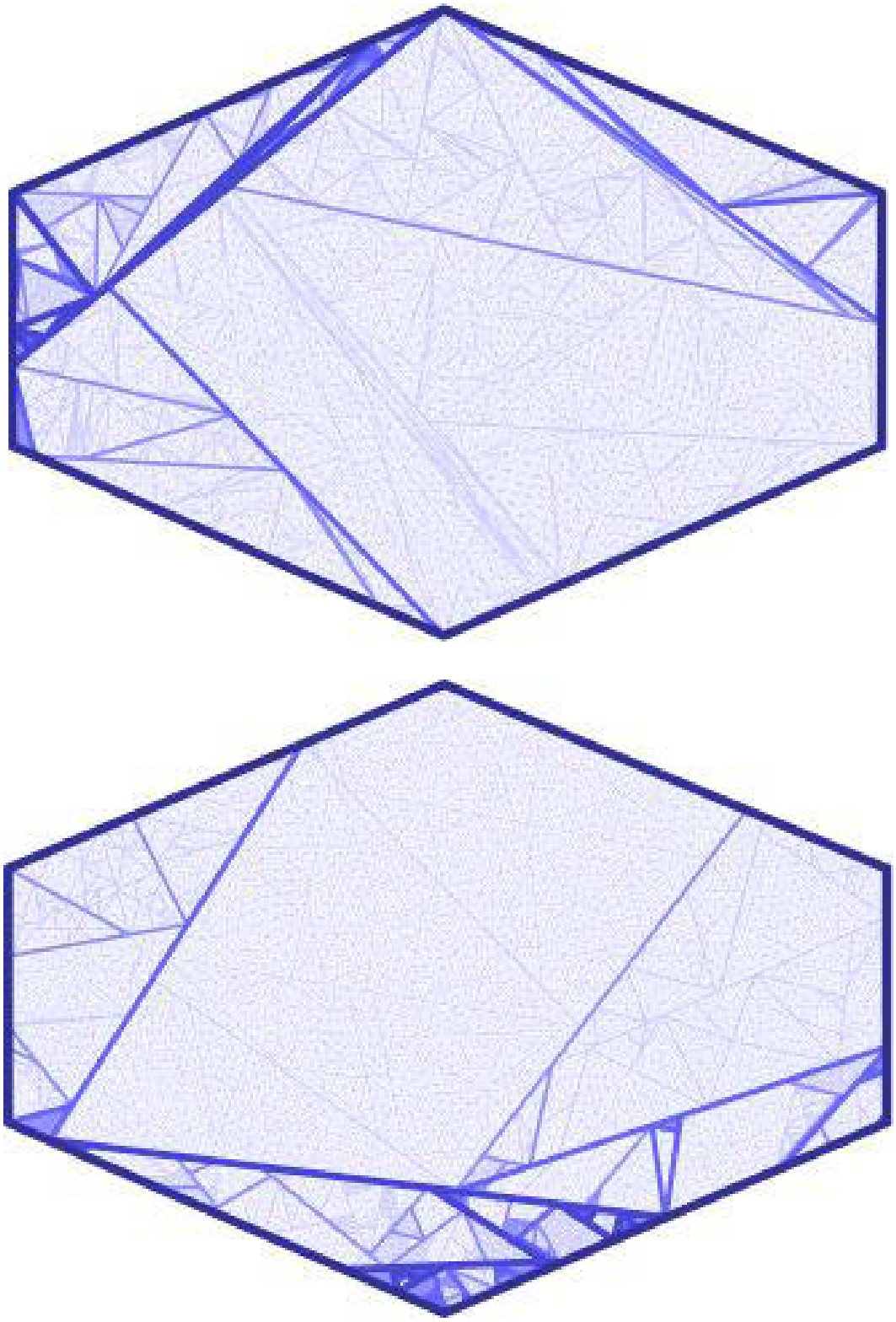


Figure 3-16 a visualization using the D&C Triangular algorithm on a hexagon for a file-system with approximately 16,600 vertices and 10 levels, showing a) without the minimum angular constraint(up) and b) with the minimum angular constraint(down).

In all these figures, the quality of aspect-ratio of angles in the polygons are improved quite significantly compared to those without angular resolution constraint. However, in order to prove the improvements, we conducted a scientific evaluation on the measurement of angle aspect ratios. In the experiments, we performed the datasets on a square container of 1000x1000 pixels. Datasets consists of three sets, varying from 12,000 nodes to 16,600 nodes and 81,000 nodes. Only Triangular polygons were considered in the evaluation. The optimal angle is selected as 60^{o} corresponding to a regular triangle.

Table 3-1 shows the evidence of the improvement in angle-aspect ratios by using angular Resolution constraint in comparison with the original *D&C* Triangular algorithm. The average of difference shows the average of variations of the angles against the optimal angle inside a triangle. Other columns indicate the percentages of the number of angles at each segment, including below 25^{o}, 25^{o} to 45^{o}, 45^{o} to 75^{o}, 75^{o} to 95^{o}, and above 95^{o}.

Table 3-1 shows that, 25% of the total results are outside of the desirable thread-hole, with original algorithm, i.e. below 25^{o} or above 95^{o}. Nevertheless, the angle resolution constraint algorithms improve the outcome significantly. Almost none of the angles are below the constraint of 25^{o} used in this evaluation and also fewer for those above 95^{o}. It also indicates that around 30% improvement of the number of angles at the ideal thread-hole (45^{o} to 75^{o}).

Table 3-1 Angle aspect ratio of polygons (triangles) using D&C Triangular Algorithm and D&C Triangular with angle Resolution constraint.

data	Methods	Avg of Diff	$\theta < 25^o$	$25^o < \theta < 45^o$	$45^o < \theta < 75^o$	$75^o < \theta < 95^o$	$\theta > 95^o$
12,000	D&C	27.2	10.4	30.8	29.5	12.1	17.2
12,000	Angle	20.2	0.2	33.5	40.1	14.8	11.4
16,600	D&C	25.5	8.2	32.0	30.3	12.7	16.8
16,600	Angle	20.2	0.2	33.6	39.6	15.7	11.0
81,000	D&C	25.8	8.8	31.1	30.7	12.6	16.8
81,000	Angle	20.3	0.1	34.0	39.8	14.3	11.7

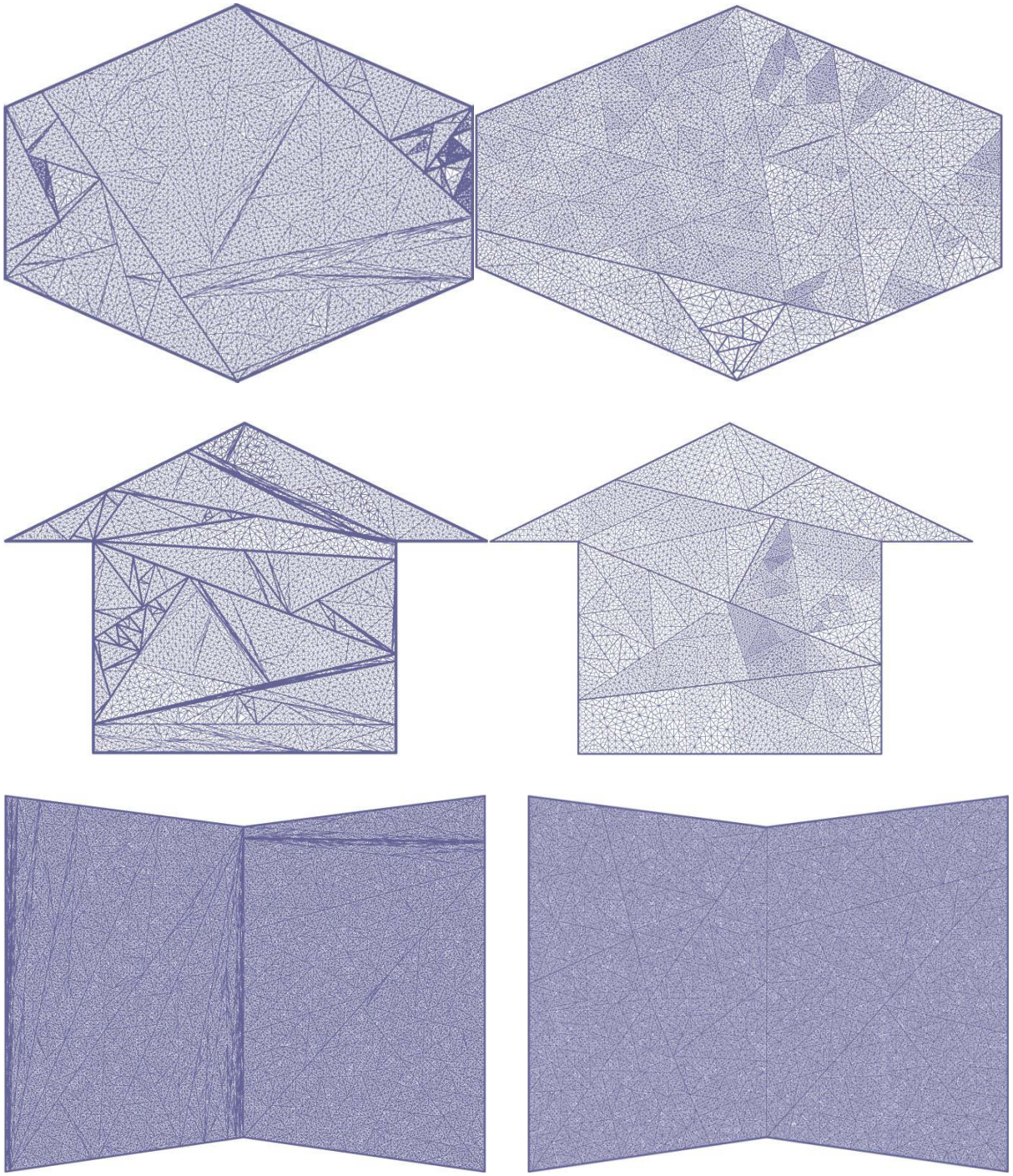


Figure 3-17 Layout results with angular resolution constraint Improvement overview: Left columns are experimental results generated by D&C Rectangular Algorithm and right columns are experimental results by D&C Rectangular Algorithm with angular resolution constraint

3.2.3 ANGULAR POLYGONAL APPROACH

In the second stage, we confine partitioning angle to maintain the orientation of titling. This partition method is called angular polygonal approach which generates most polygonal titling in data representation. In a rectangular or square container, it generates mostly oriented rectangles (with rotation).

Angular Polygonal Approach illustrates the partitioning process in Section 3.2.3.1, discusses the algorithm in Section 3.2.3.2, and shows the experimental results in Section 3.2.3.3.

3.2.3.1 ANGULAR POLYGONAL PARTITIONING PROCESS ILLUSTRATION

In the polygon $P(N_i)$, vertices $v_{i,0,\ldots}, v_{i,i,\ldots}, v_{i,m}$ generates a number of m sides $L(v_{i,0}, v_{i,1}),\ldots, \ell(v_{i,i}, v_{i,i+1}),\ldots, l(v_{i,m-1}, v_{i,m})$.Among the sides, the longest side is defined as $\boldsymbol{\ell(v_{e-1}, v_e)}$. The very first partitioning line begins from the longest side $\boldsymbol{\ell(v_{e-1}, v_e)}$. The partitioning direction is against to the longest side $\boldsymbol{\ell(v_{e-1}, v_e)}$ with selected angle α and then following partitioning line continues from the previous created line in vertical direction against itself. In the implementation, α is restricted to 15º, 30º, 45º, 60º, 75º, and 90º for demonstration purpose. The angular partitioning applies until the leaf nodes of this particular polygon are reached. As a result, the $P(N_i)$ be partitioned into sub-polygons for $\{ N_{i,1},\ldots,N_{i,i},\ldots,N_{i,n} \}$ with corresponding polygonal boundary $\{ p(N_{i,1}),\ldots, p(N_{i,i}),\ldots, p(N_{i,n})\}$ and with weight $\{w(N_{i,1}),\ldots, w(N_{i,i}), \ldots, w(N_{i,n})\}$.

We use an example to convey the approach with different selected *angle α*. For example, a tree rooted at 0.0 has 6 child nodes *{1.0, 1.1, 1.2, 1.3, 1.4, 1.5},* whose weights are *{8, 6, 1, 2, 4, 1}* respectively, according to the number of sub-nodes contained (see Figure 1-19). Figure 3-18 shows the partitioning at 60 degree.

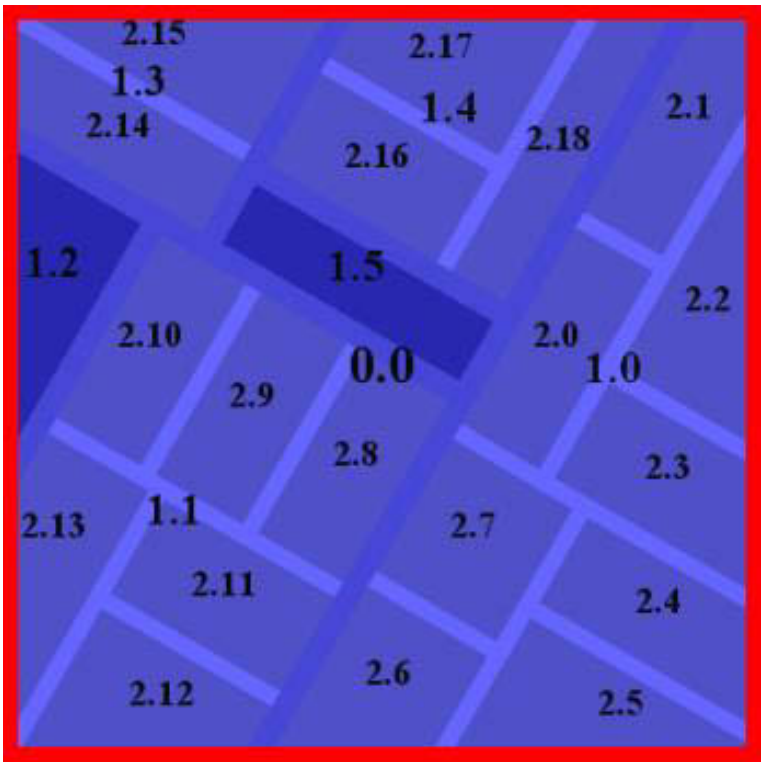


Figure 3-18: Angular Polygonal partitioning process output illustration: Layout outputs of the same sample data as in Figure 1-19 using our algorithm with angles (60 degree in the case).

3.2.3.2 Angular Polygonal Algorithm

We start the partition on the longest side of the polygon. This property is aimed to improve the balance when we divide the polygon into two halves for the two corresponding sub-lists of vertices. The following partitioning processes use the prior cutting side as the starting side to do the division. This property ensures the following cuttings are followed the initial angle. The algorithm for dividing a polygon into two sub-polygons is outlined as below

Algorithm: Angular Polygonal Approach

Input: a polygon P with m side vertices $\{v_1, v_2,..., v_m\}$ in R^2 for partitioning and the starting vertex V_s and partitioning angle α

Output: two sub polygons P_1 and P_2 with area of a_1 and a_2 and $P = P_1 \cup P_2$

Algorithm: ***AngularDivide()***

1: t = 1; // index of the side vertex V_t

2: P* = null; // current processing polygon

3: $v_s = v_{start}$; // start point v_s is at v_{start}

3: while $a_1 > a^*$ // a* is the area of P*

4: find the cutting point v_c on P with angle α;

5: if the area of the new polygon divided by segment $v_s v_c < a_1$

6: shift v_s slightly on the side $v_{start}v_{start+1}$;

7: end while // repeat on other side(s) if no solution is found

10: $P_1 = P^*$

11: $P_2 = P - P_1$

Note: v_c is the cutting point of the polygon and the line that is drawn from v_s and with angle α . If the angle α= is 90°, we have a special case of horizontal-vertical rectangular partition, similarly to traditional treemap algorithms (see Section 3.2.4).

3.2.3.3 Angular Polygonal Treemap Experimental Results

We adopt Angular Polygonal algorithm in larger data set visualization to demonstrate experimental results. For example, Figure 3-19 shows Layouts of a large data set with approximately 12,000 nodes a) 75 degree, and b) 45 degree.

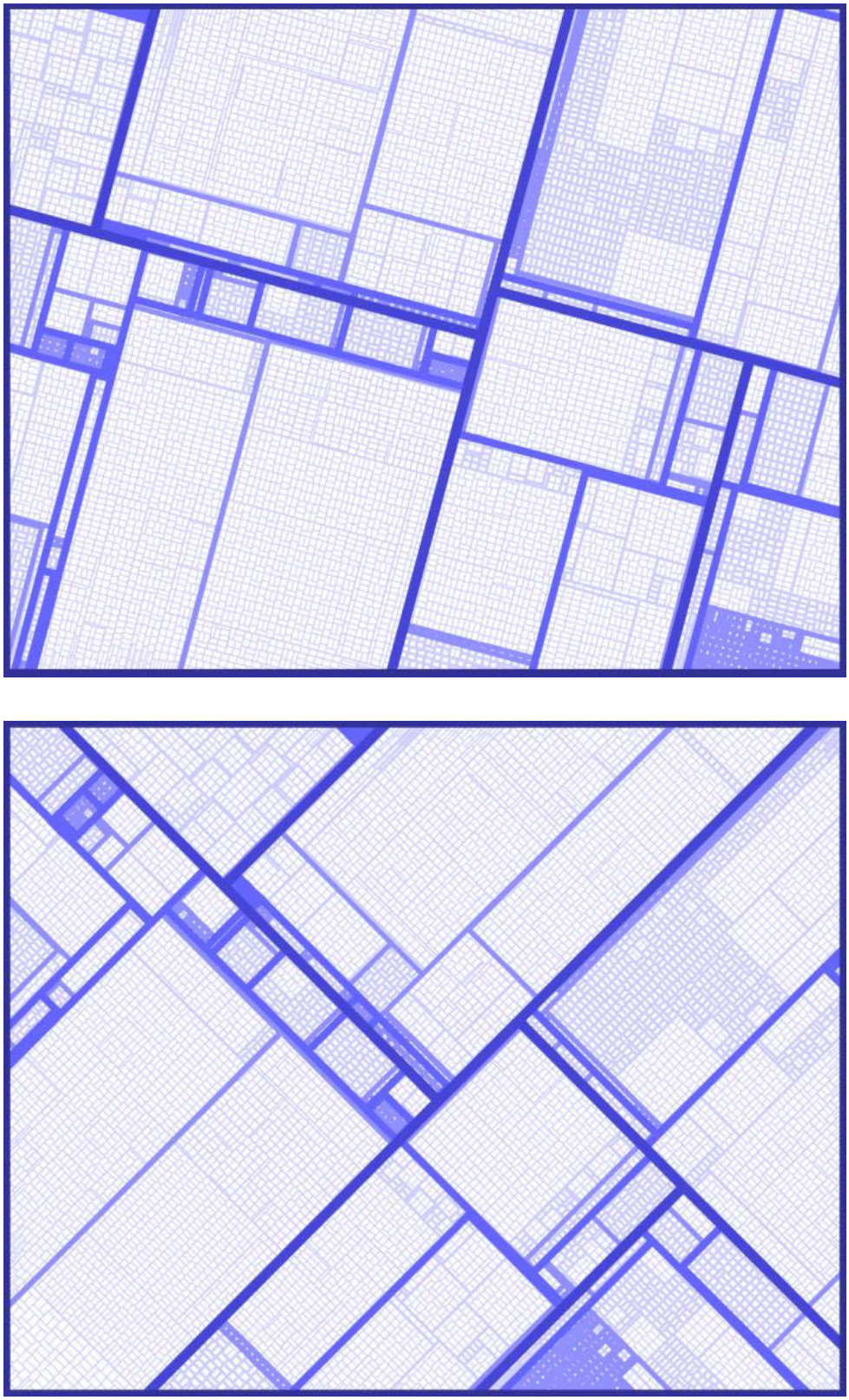


Figure 3-19 Experimental results of Angular Polygonal Treemap: Visualization of a large data set with approximately 12,000 nodes a) 75 degree, and b) 45 degree.

3.2.4 D&C Rectangular Approach

In the third evolution of approach, we restrict partition angular to right angle. This approach preserves the orientation of titling with horizontal and vertical directions. This partition method is called D&C Rectangular treemap which generates most rectangular titling in representation. In a rectangular or square container, it generates rectangular tree maps, which looks similar with traditional tree maps.

D&C Rectangular Approach illustrates the partitioning process in Section 3.2.4.1, discusses the algorithm in Section 3.2.4.2, and shows the experimental results in Section 3.2.4.3.

3.2.4.1 Partitioning Process Illustration

We illustrate the process of D&C Rectangular Approach with the same dataset in section 3.2.3. The partitioning process is showed in Figure 3-20 with a small dataset visualization result.

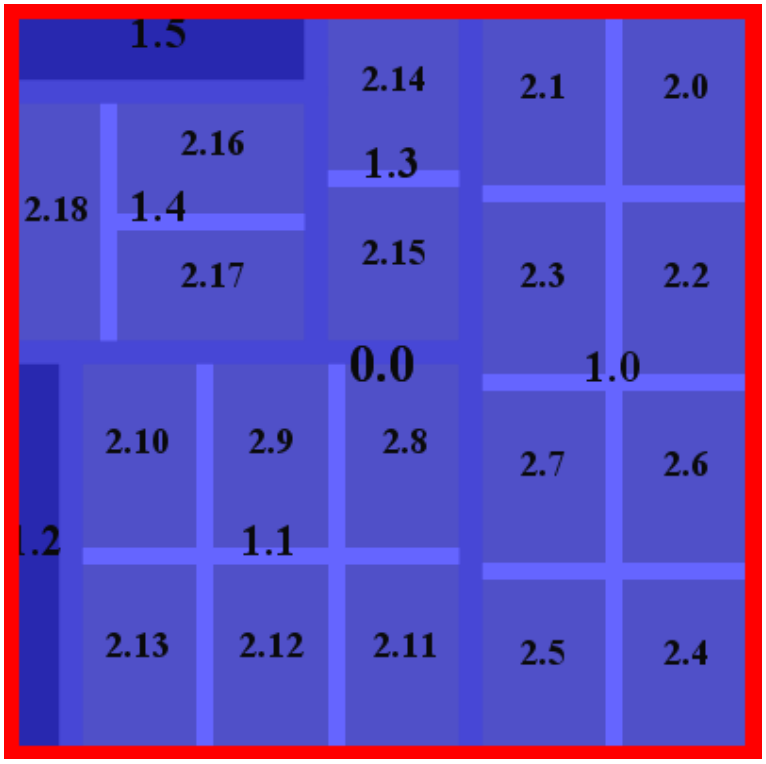


Figure 3-20 D&C Rectangular partitioning process output illustration: Layout outputs of the same sample data as in Figure 1-19 using D&C Rectangular algorithm

3.2.4.2 D&C Rectangular Algorithm

The D&C Rectangular Algorithm employs the same algorithm in Section 3.2.3.2, with one condition that the partitioning angle α is 90o at each process. So that we have a special case of horizontal-vertical rectangular partition, similar to traditional treemap algorithms (see Figure 3-20).

3.2.4.3 D&C Rectangular Experimental results

Figure 3-21 to Figure 3-23 are examples of the visualizations of the D&C Rectangular Treemap layout algorithm on a variety of data sets and container shapes. Figure 3-21 shows Layout of a large data set with approximately 12,000 nodes with 90 degree. Figure 3-22 shows the visualization of a data set with 272 nodes using D&C Rectangular Treemap with various angles on a triangle display area. Figure 3-23 illustrates the visualization a large data set with 16,000 nodes with more than 10 hierarchical levels, using D&C Rectangular Treemap on an octagon. The visualization results use the colour and edge's thickness to enhance the visibility whilst Figure 3-23b uses gaps to produce the similar effect.

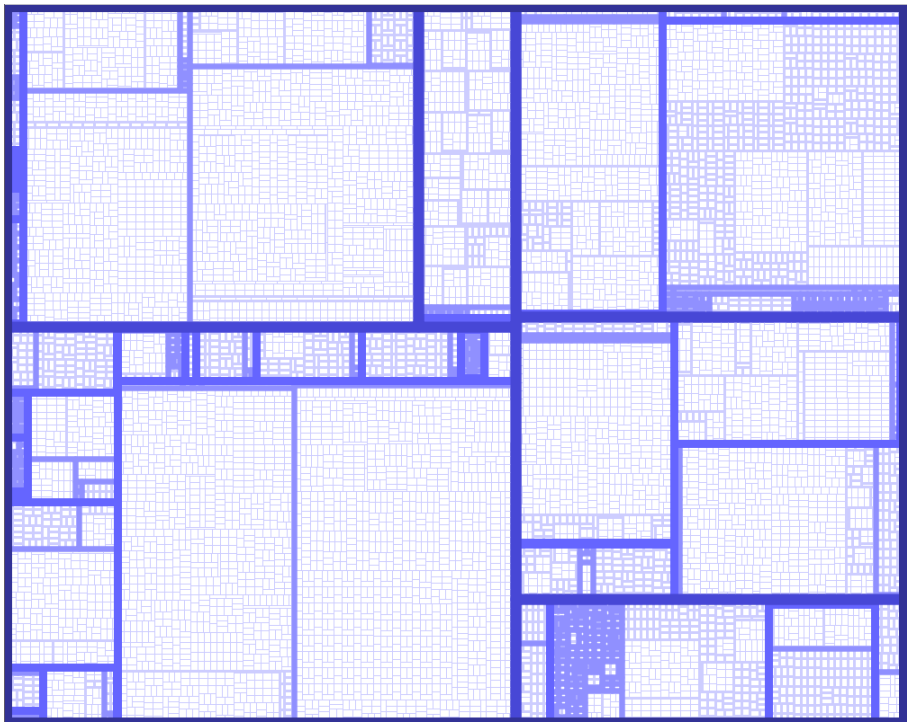


Figure 3-21 Experimental result of D&C Rectangular Treemap in Rectangular container: layouts of a large data set with approximately 12,000 nodes

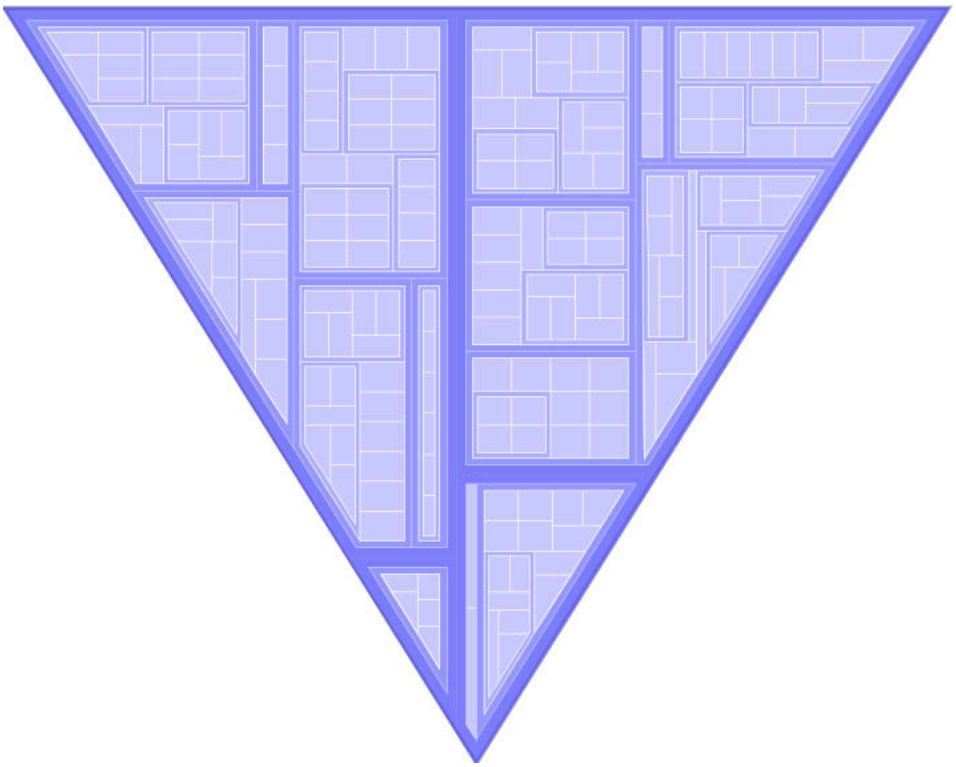


Figure 3-22 Experimental result of D&C Rectangular Treemap in triangular container: Visualization of a data set with 272 nodes using D&C Rectangular Treemap

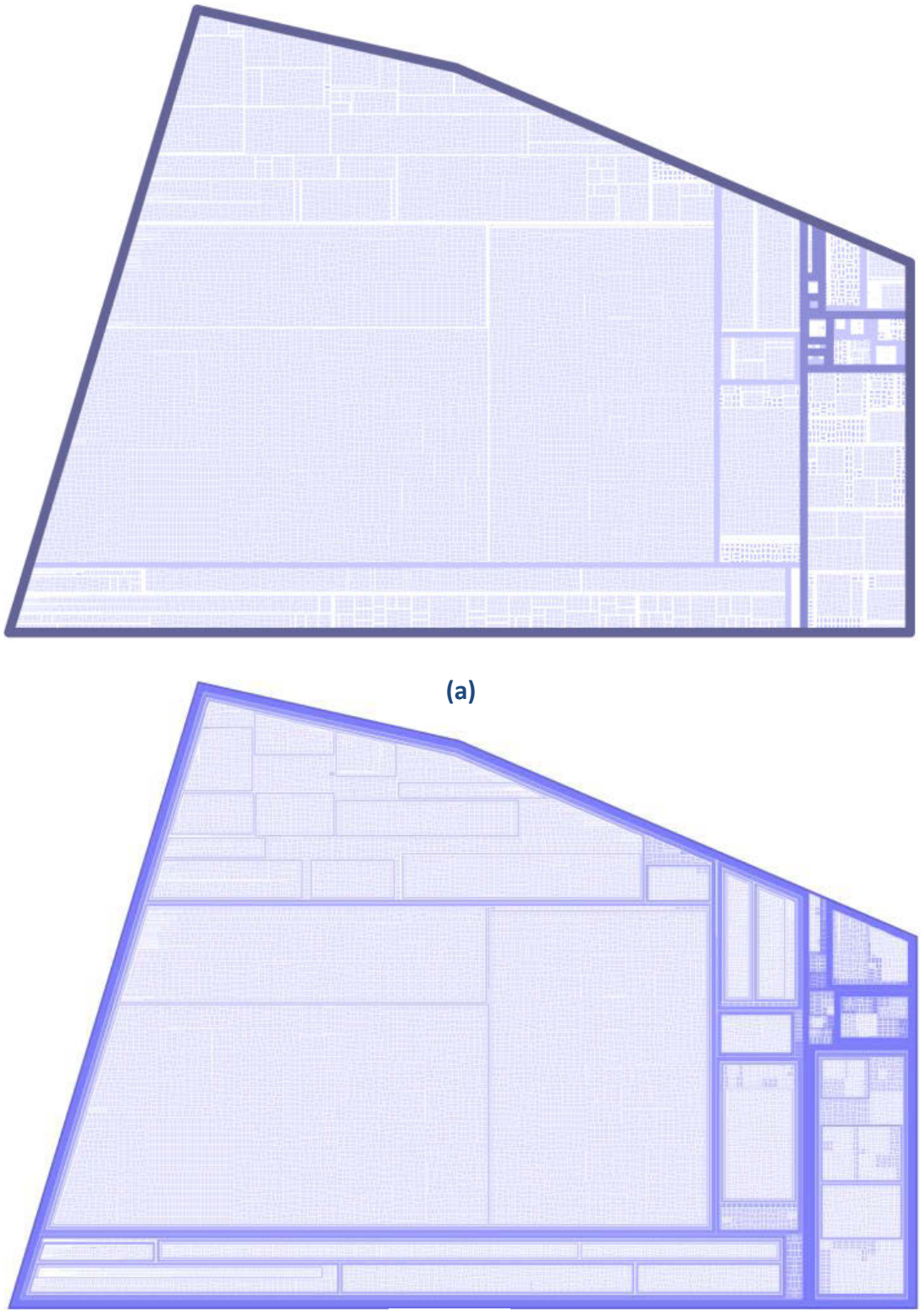


Figure 3-23 D&C Rectangular Treemap experimental result in polygon container: Visualization of a large data set with approximately 16,000 nodes (> 10 hierarchical levels) using D&C Rectangular Treemap. a) colour and edge's thickness to enhance the visibility b) boundary gaps to produce the similar effect.

SECTION 3.3 TANGRAM TECHNIQUES

Tangram Treemaps represent hierarchy using containment which enclose child nodes in parent nodes. Tangram Treemaps also encode values using area. Therefore, the Tangram treemaps can be used in hierarchy exploration and value comparison, two typical categories tasks in treemap visualization. Particularly for Tangram treemaps, the containment we examined has three ways: firstly, treemaps with rectangular container have the control of changing shapes of child; secondly, treemaps can tailor representation into different shaped containers, with the control of changing shapes of children. Thirdly, both containment and container are encoded to present structure datasets.

Tangram Treemaps with these different algorithms provide an ability to mix different sub-layouts in the visualization, including horizontal-vertical rectangular layout, angular rectangular layout and polygonal layout for emphasizing the importance or the focused sections. System implemented with data mining query implemented can automatically generate layout based on query to draw users' attention and recommend the focus of interests. Also users can rotate particular sections, based on user's own interest in particular task. It can help them to track their mental map during exploration and also to compare objects with another. We demonstrate its effectiveness in Chapter 7. The usability study in Chapter 6 proves that the mixture of different layouts in real scenarios has successfully enhanced the visibility and highlighting focus of interests by differentiation.

In the following sections, we present layouts with containment control in Section 3.3.1, with container control in Section 3.3.2 and Section 3.3.3, and with both of containment and container in Section 3.3.4.

3.3.1 CONTAINMENT CONTROL

For containment variation, the combination of two or three algorithms could be utilized in representation to emphasis place of interest by users. For example, we can use Angular Polygonal algorithm or D&C rectangular algorithm embedded in D&C rectangular algorithm. This approach can be used to highlight with similarity or differentiation.

Firstly, we can highlight one focus in the same level. Figure 3-24 highlights one focus with angular polygonal algorithm in D&C rectangular treemap, which is contained in level 1 in Figure 3-24, level 2 in Figure 3-25. Figure 3-26 emphasizes one focus with D&C triangular algorithm in D&C rectangular treemap at the same level.

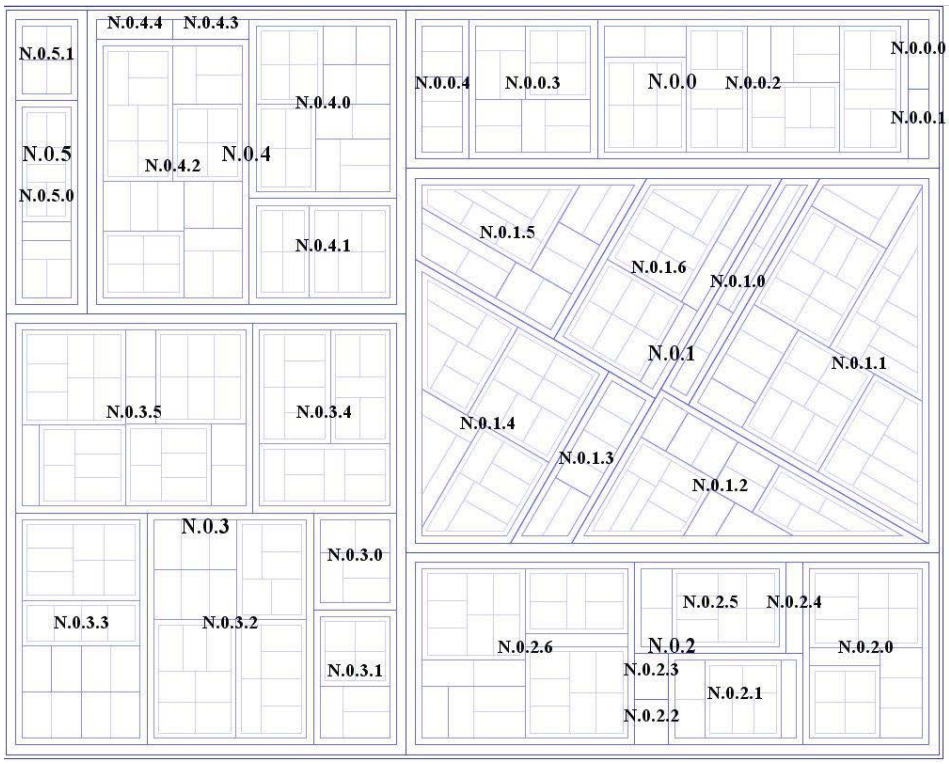


Figure 3-24 Containment control with one focus illustration 1: N 0.1 contained in level 1 is highlighted as one focus with angular polygonal algorithm in D&C rectangular treemap.

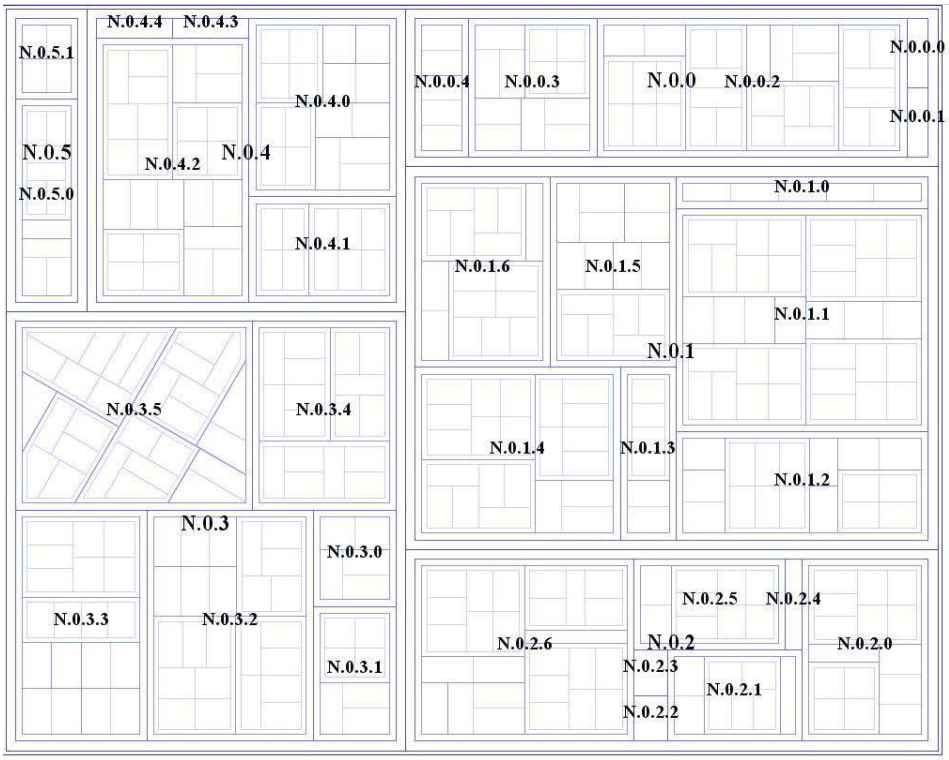


Figure 3-25 Containment control with one focus illustration 2: N0.3.5 is contained in level 2 is highlighted as one focus with angular polygonal algorithm in D&C rectangular treemap.

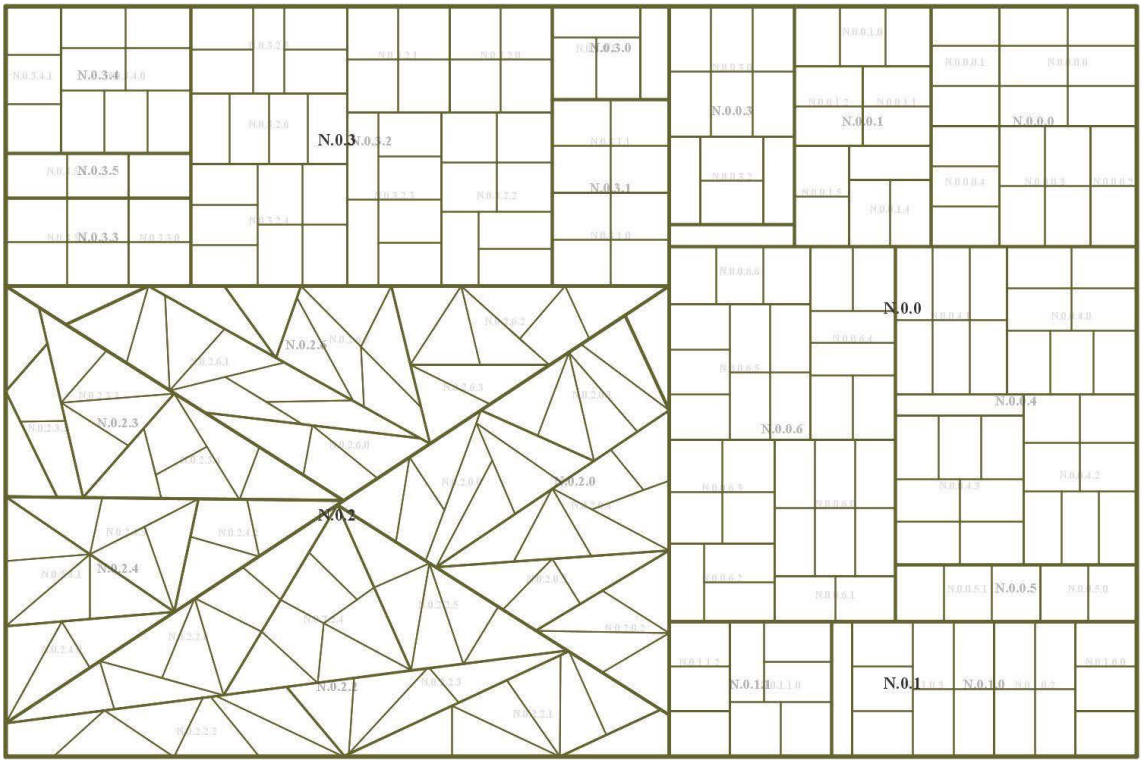


Figure 3-26 Containment control with one focus illustration 3: Tangram Treemaps emphasizes N 0.2 with D&C triangular algorithm in D&C rectangular treemap in the same level.

Secondly, Tangram treemaps can highlight two focuses on the same level or different levels. Figure 3-27 highlights two focus of N 0.2.1 and N 0.2 5 in the first level by similarity, where D&C Triangular algorithm is embedded in D&C Rectangular layout. Figure 3-28 emphasizes two focuses in different levels. It applies combination of Angular Polygonal Treemap and D&C rectangular treemap. The nodes of N 0.2 in the first level and N 0.4.1 in the second level are rotated. Figure 3-30 draws users' attentions on N0.0.5 on the first level and N 0.3 on the second level by using D&C Triangular Treemaps in D&C rectangular treemap.

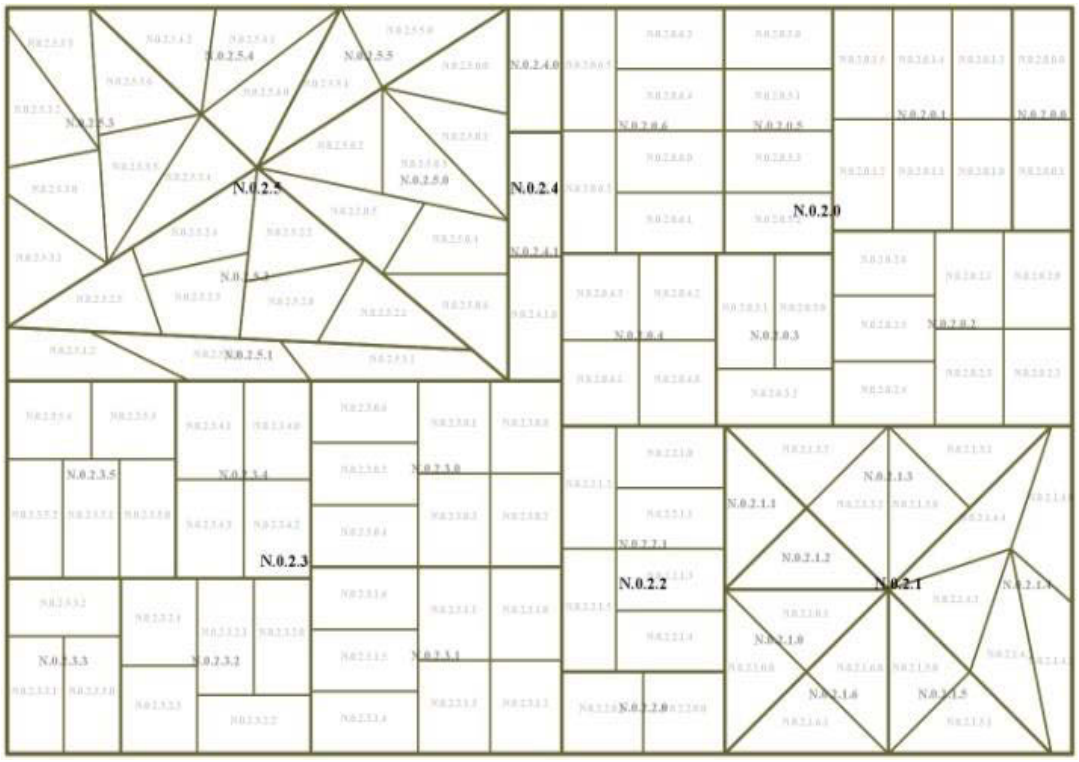


Figure 3-27 Containment control with two focuses illustration 1: Tangram Treemaps Highlights two focus of N 0.2.1 and N 0.2.5 in the first level by similarity. D&C Triangular algorithm is embedded in D&C Rectangular layout.

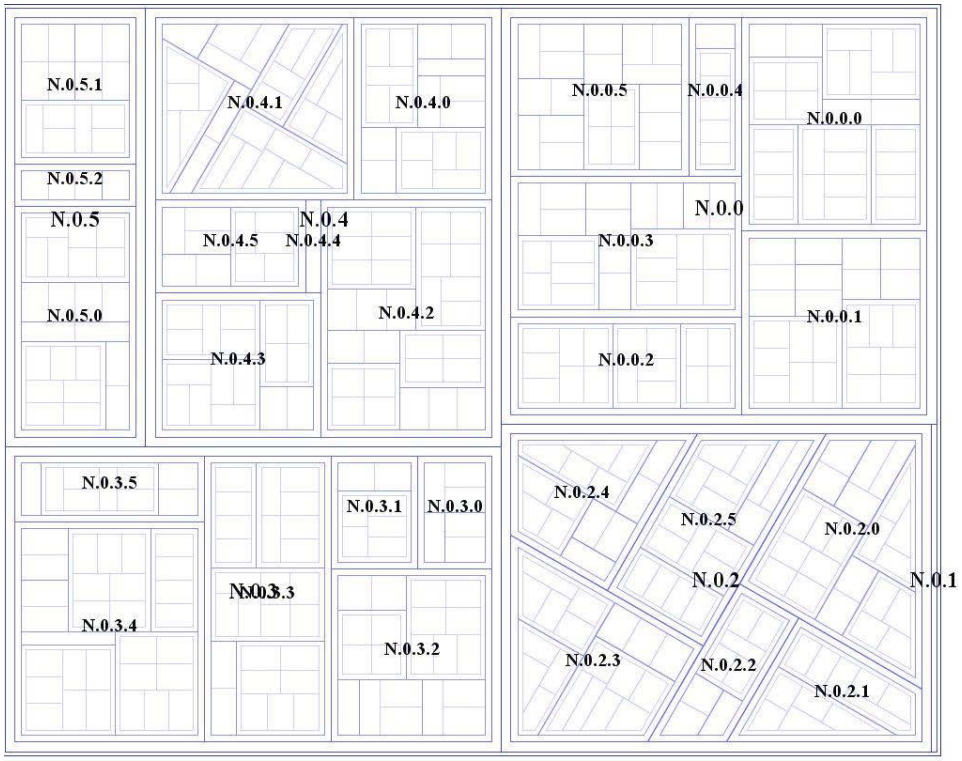


Figure 3-28 Containment control with one focus illustration 2: Tangram Treemaps emphasizes two focuses in different levels. It applies combination of Angular Polygonal Treemap in D&C rectangular treemap. The nodes of N 0.2 in the first level and N 0.4.1 in the second level are rotated.

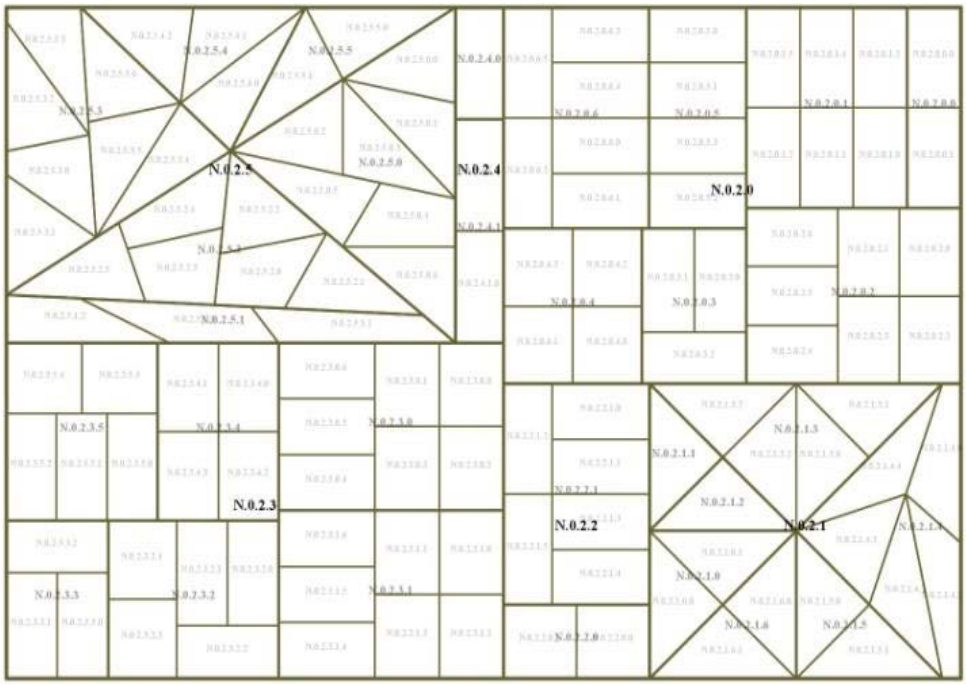


Figure 3-29 Containment control with two focus illustration 3: Tangram Treemaps emphases two focuses N 0.2.5 and N 0.2.1 at the same levels. It applies combination of D&C Triangular Treemap in D&C rectangular treemap.

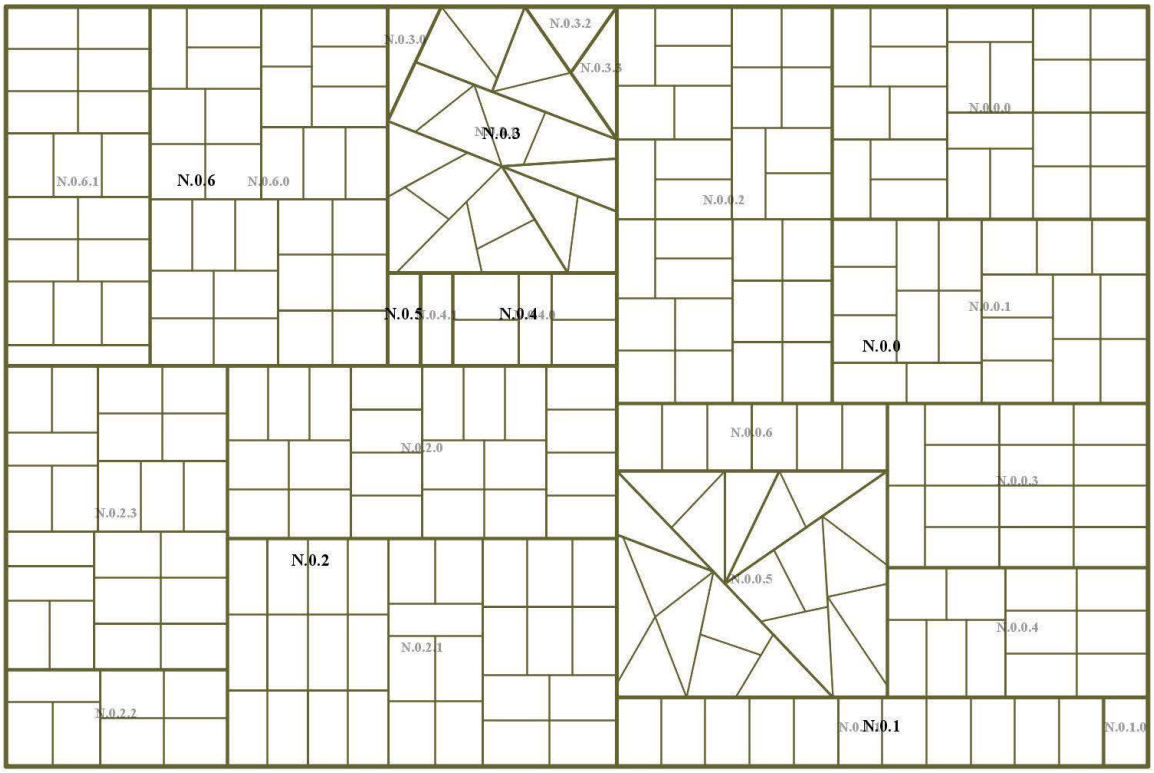


Figure 3-30 Containment control with two focus illustration 4: Tangram Treemaps draw users' attentions on N0.0.5 in the first level and N 0.3 in the second level by combination of D&C Triangular Treemap in D&C rectangular treemap

Thirdly, it can also be applied to highlight three focuses on different levels. Figure 3-31 exploited D&C triangular in rectangular treemap to highlight three focuses in different levels, N 0.0 in first level, and N 0.1.1 and N 0.1.4 in the second level.

Lastly, Tangram treemaps are able to employ D&C angular polygonal algorithm with multiple partition angles, to emphasis the data structure. For example, in Figure 3-32 , when level of visualization goes deeper, the rotation angular decrease, which creates a sense of levels changing for structure tracking.

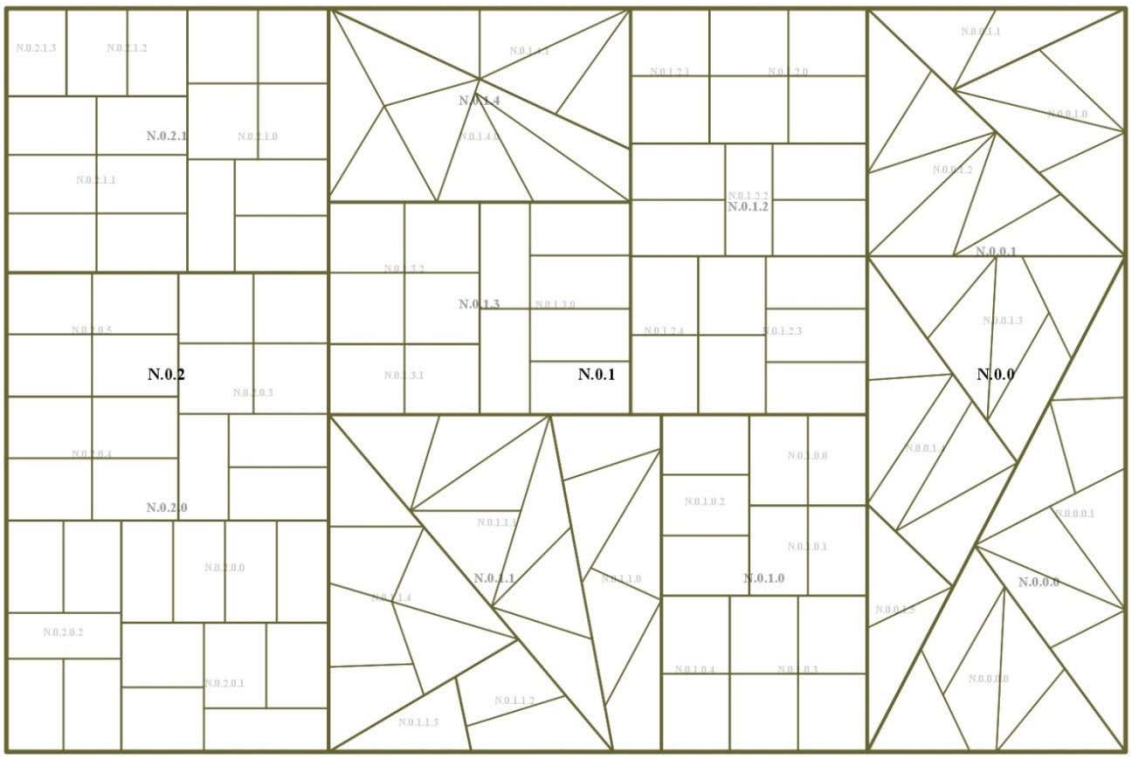


Figure 3-31 Containment control with three focus illustration: Tangram Treemaps exploit D&C triangular in rectangular treemap to highlight three focuses in different levels, N 0.0 in first level, and N 0.1.1 and N 0.1.4 in the second level.

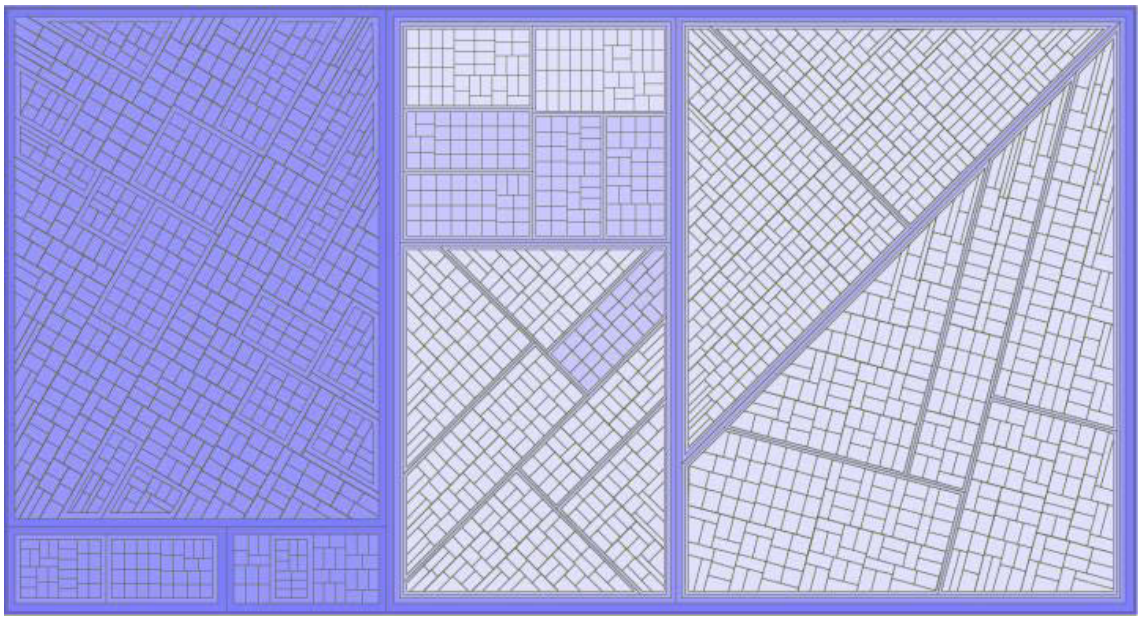


Figure 3-32 An example of a visualization using an angular polygonal algorithm on sub-structures with various partitioning angles.

3.3.2 CONTAINER CONTROL

The flexibility of our approach is also applied to the container variation. The advantage of the algorithm is the ability to be extended to various enclosed display container rather than rectangular only. This allows Tangram Treemaps with other visualization method to be adopted into a wider range of domain and applications. This section shows the ability of angular polygonal treemap with container control in Section 3.3.2.1 and D&C triangular treemap with container control in Section 3.3.2.2.

3.3.2.1 ANGULAR POLYGONAL TREEMAP IN CONTAINER CONTROL

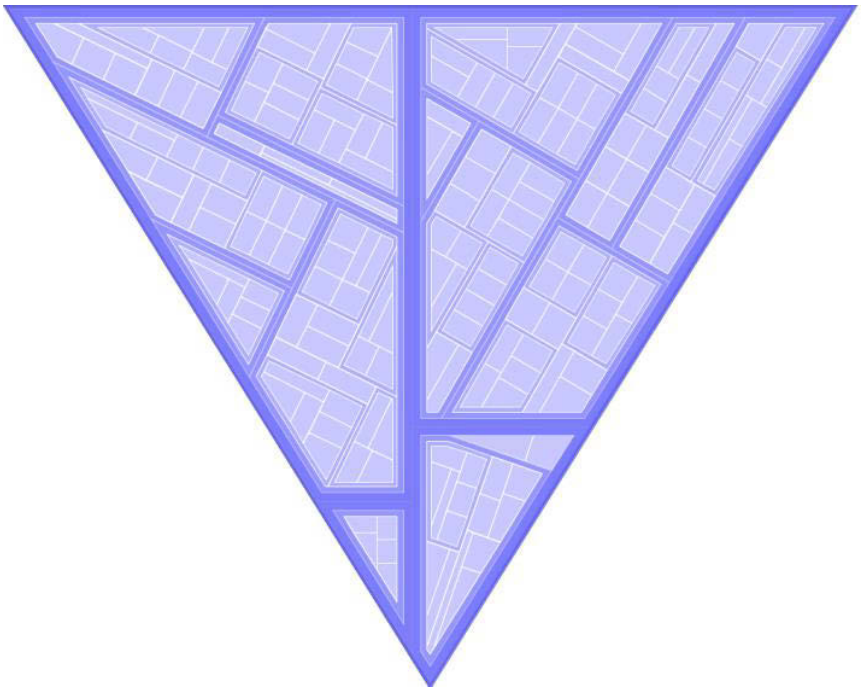


Figure 3-33 Triangle container illustration: Visualization of a data set which has 272 nodes with two sub-hierarchy rotated by different angles in four divided segments

We employ the angular polygonal treemap in various shapes. Examples for extended angular polygonal treemap are presented in different containers, e.g. triangle, pentagon and hexagon (Figure 3-33 & Figure -34). Figure 3-33 illustrates Angular polygonal Treemap layout of 272 nodes within a triangle shape. It applies different rotated angles to four divided segments. Figure -34 presents the visualization of a very large data set with over 81,000 nodes, generated by Angular Polygonal Treemap with 45 degree and 15 degree on a hexagon container.

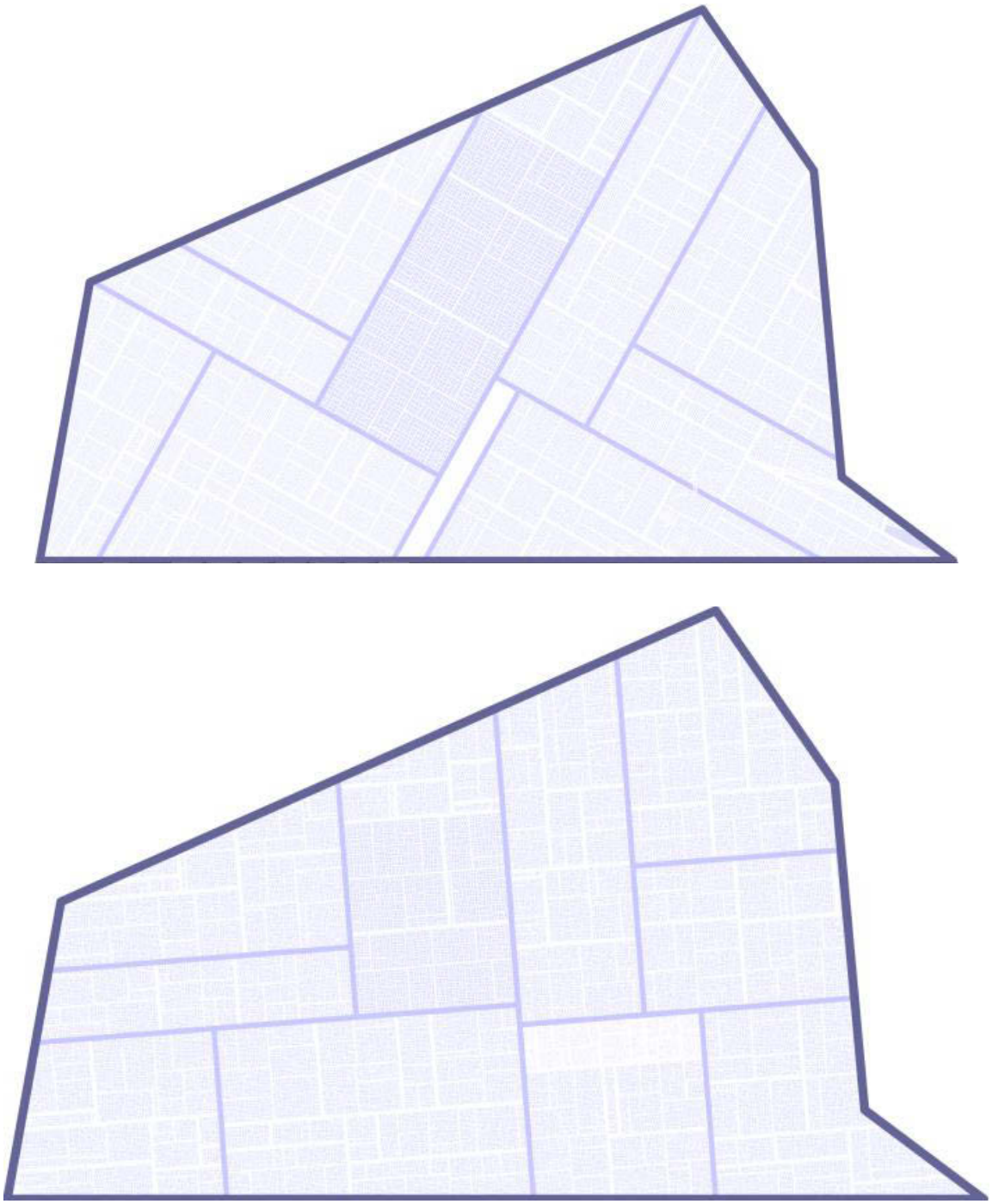


Figure -34 Hexagon container illustration: Visualization of a very large data set which has over 81,000 nodes on a convex polygon, with partition angle of 45 degree (up) and 15 degree (down).

3.3.2.2 D&C Triangular Treemap in Container control

The improved D&C Triangular algorithm with angular Resolution constraint can be further applied to regular and irregular shapes with some minor modifications (see Figure 3-35 to Figure 3-40). It offers a great flexibility to be applied into various containers.

Adaptive algorithm can also be combined with other enclosure portioning algorithms to create more effective outcomes. For convex shapes, Figure 3-35 to Figure 3-40). Figure 3-35 applies Triangular algorithm with angular resolution constraint in a pie shape with a large data set with approximately 30,000 nodes. Figure 3-36 is a visualization using the D&C Triangular algorithm with angular resolution constraint in a ribbon shape. The data set is approximately 16,600 nodes in 10 levels. Figure 3-37 further displays a uniform data set with 5461 vertices on an ellipse. This uniform data contains a 6-level tree which has 4 child vertices on each none-leaf vertex. For concave shapes, Figure 3-38 visualizes a dataset of approximately 1000 nodes, using the D&C Triangular algorithm with angular resolution constraint. Figure 3-39 adopts the algorithm into a “coin” shape with a uniform data and a random generated data. Figure 3-40 display non-uniform data in the same coin shaped container.

This adaptability can be further extended to task-based scenarios, and different tasks for different domain purpose. For example, the visualization in Figure 3-36 can be potentially adapted to visualize the change of stock markets or periodic patterns of business data. Figure 3-39 shows an example of the visualization for a hollow bounded “coin shape”. Tamassia et al. (1988) and Purchase et al. (2002) proved that placing important nodes near the centre is an important constraint on aesthetic guidelines and is the preference of most users. It provides the potential for layout with the focus in the centre.

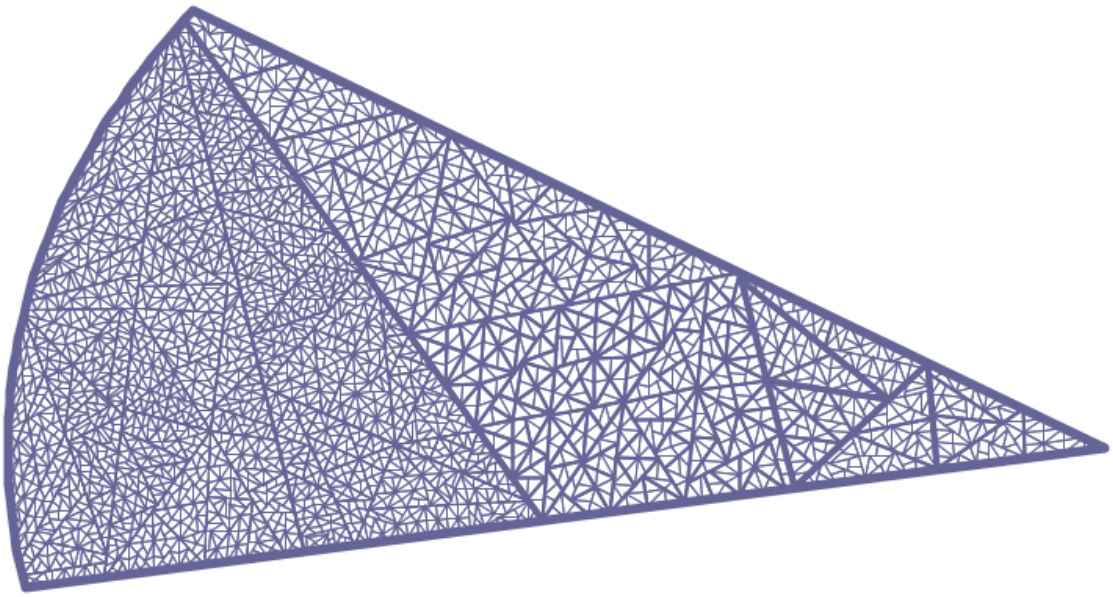


Figure 3-35 A visualization using the D&C Triangular algorithm with angular resolution constraint on a pie shape (a large data set with approximately 30,000 vertices).

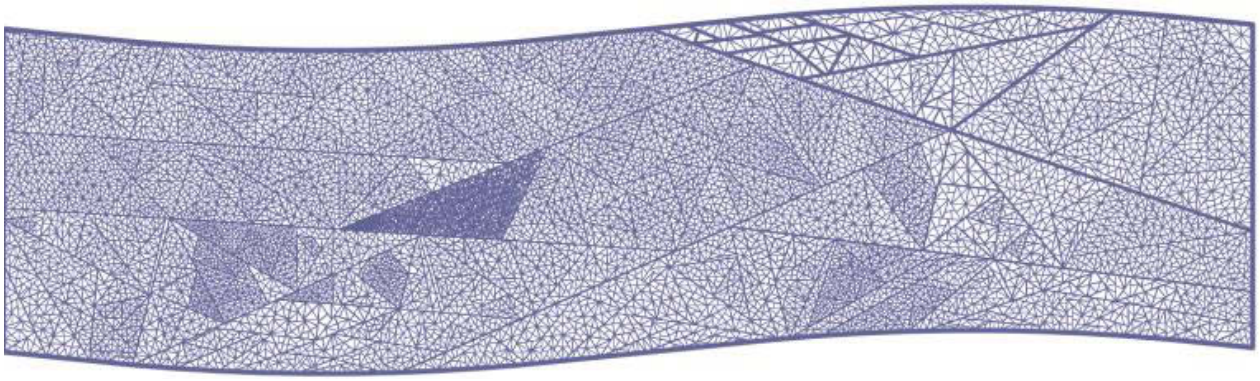


Figure 3-36 A visualization using the D&C Triangular algorithm with angular resolution constraint on a ribbon shape (a data set with approximately 16,600 vertices and 10 levels).

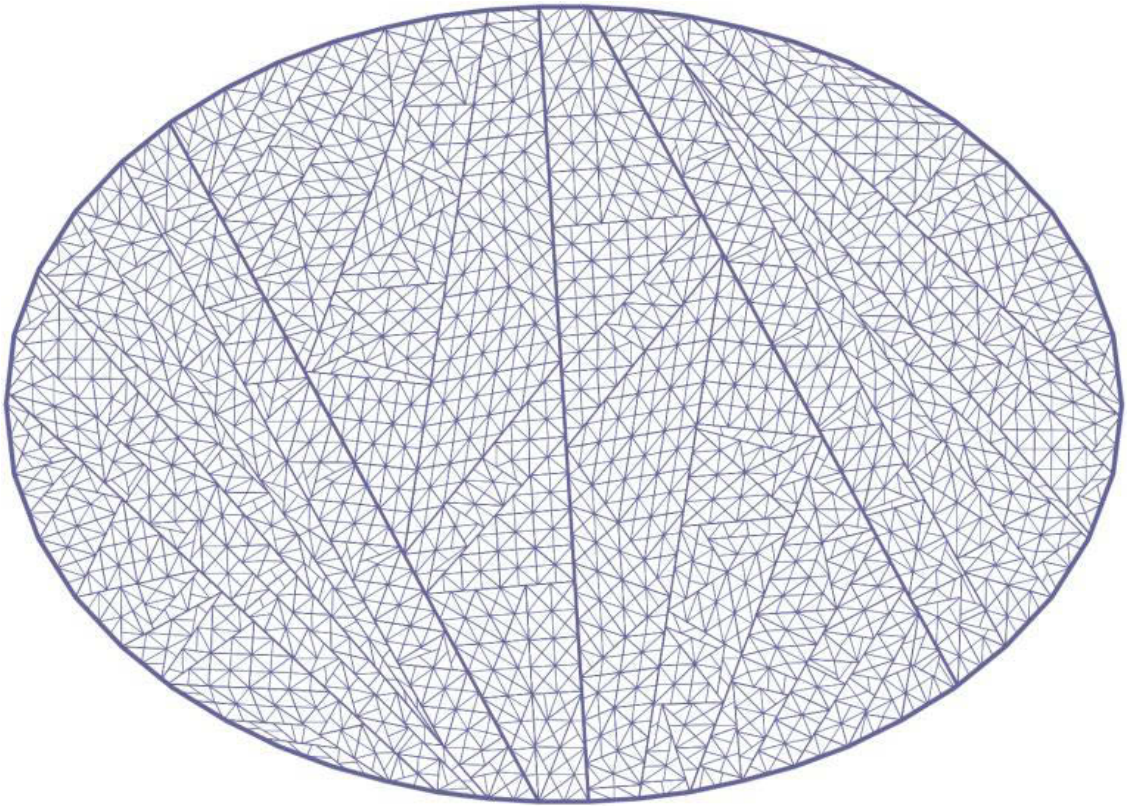


Figure 3-37 A visualization using the D&C Triangular algorithm with angular resolution constraint on an ellipse (an uniform data set with 5461 vertices; a 6-level tree whose has 4 child vertices on each none-leaf vertex).

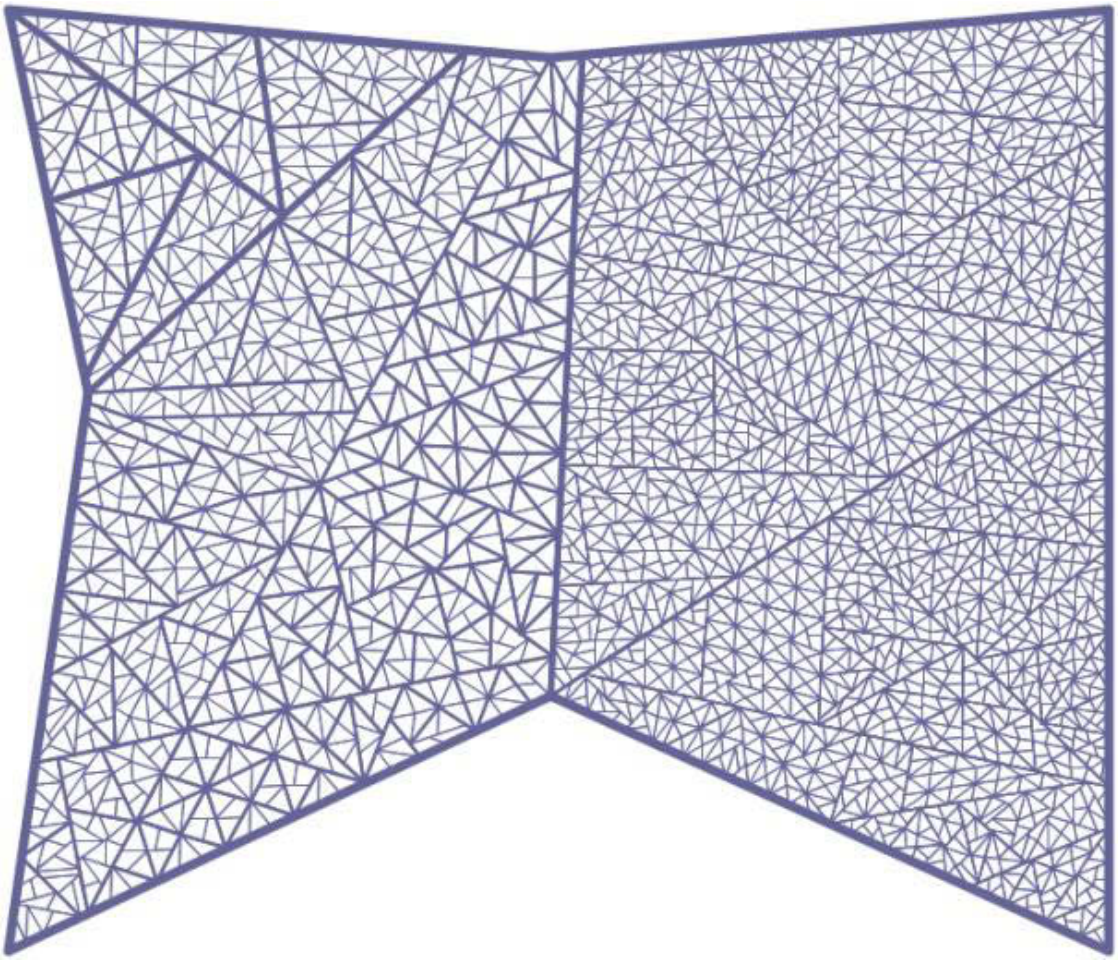


Figure 3-38 Visualizations using the D&C Triangular algorithm with angular resolution constraint approximately 1000 nodes.

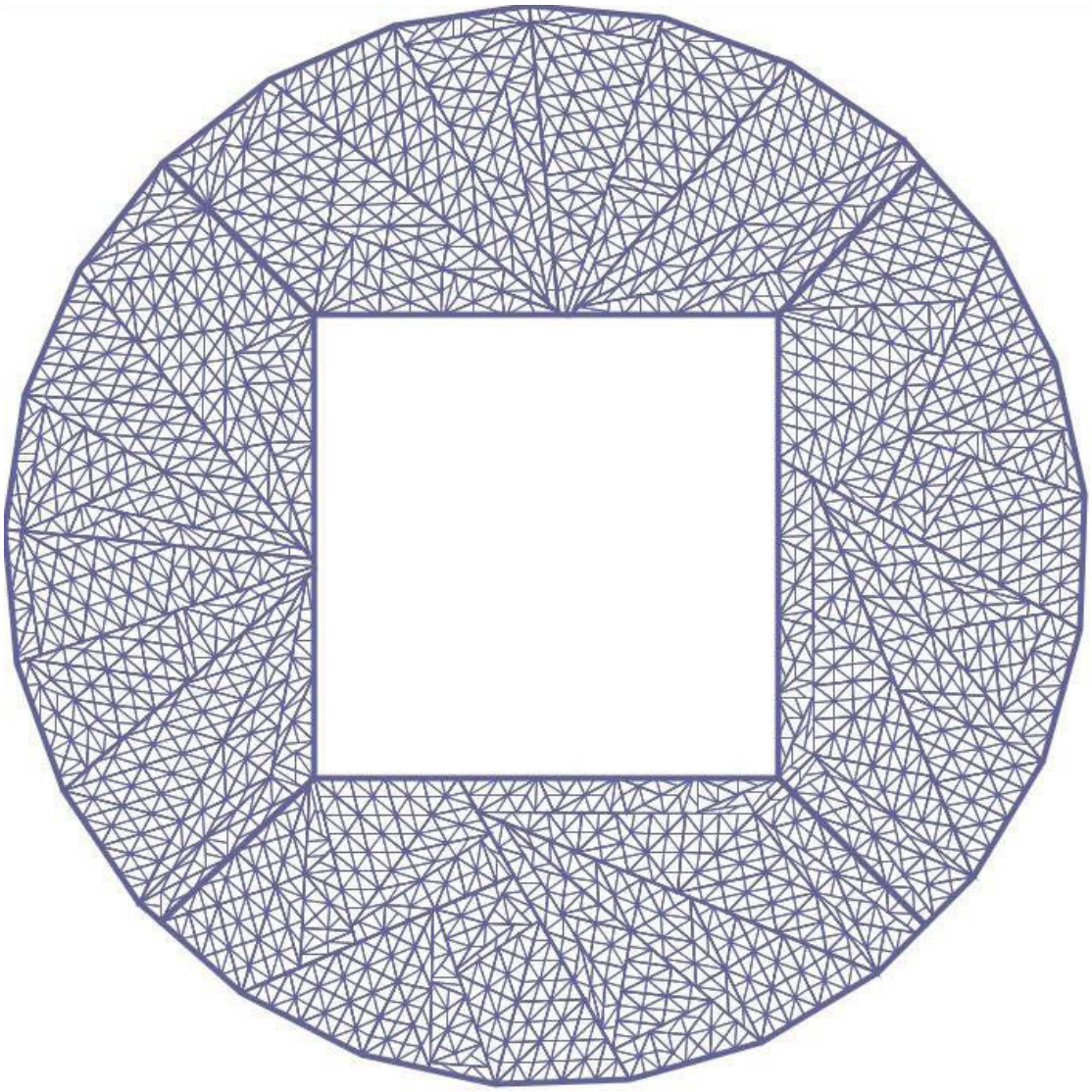


Figure 3-39 A visualization using the D&C Triangular algorithm with angular resolution constraint on a "coin" (uniform data)

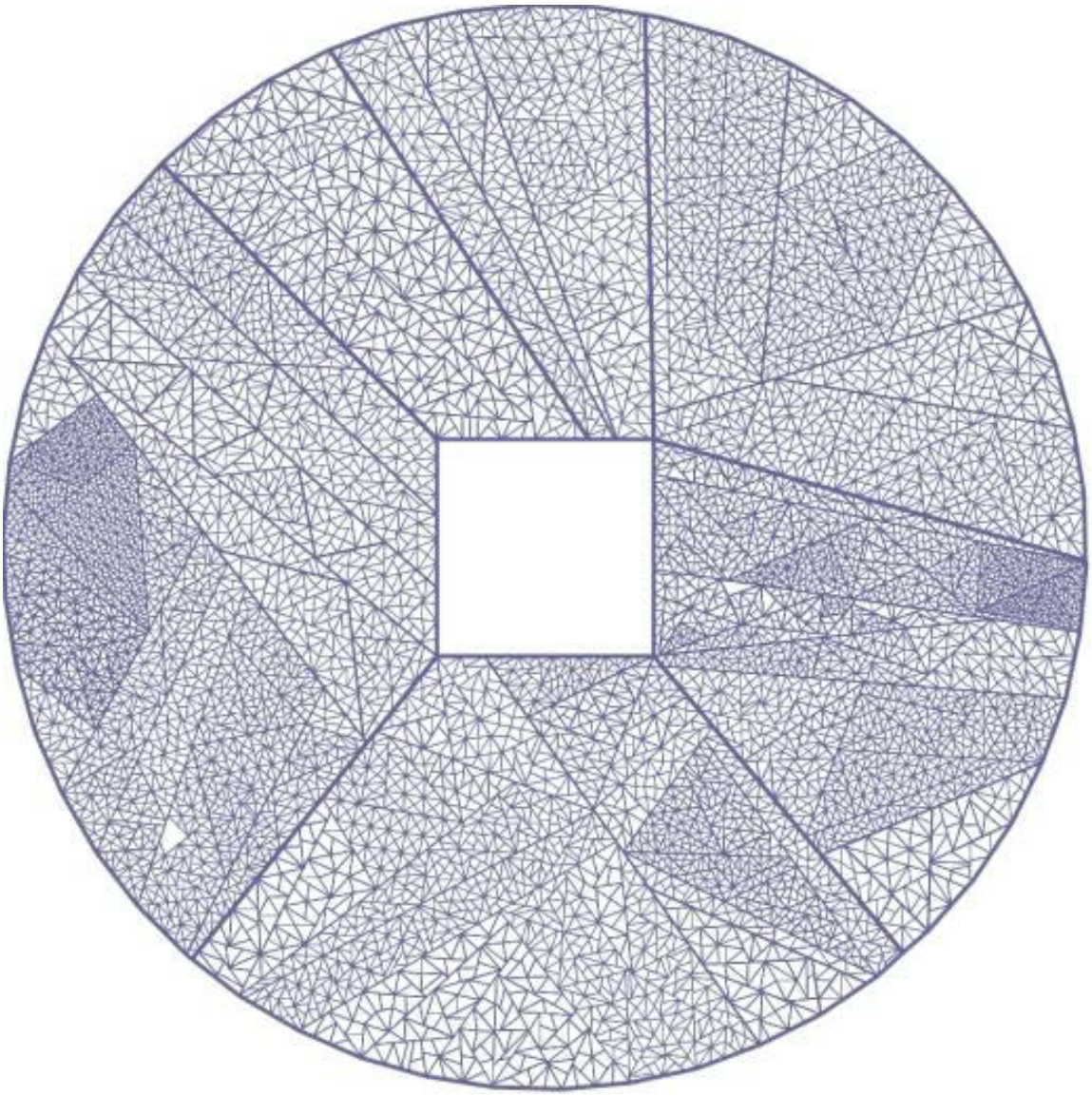


Figure 3-40 A visualization using the D&C Triangular algorithm with angular resolution constraint on a "coin" (non-uniform data)

3.3.3 EXTENDED CONTAINER WITH VISUAL PROPERTIES

In order to enhance the visibility in the hierarchical structure, we employ visual properties. Firstly, we enhance the visibility with colours and edge's thickness. Edges of higher level nodes have thicker and darker colours than those of lower level nodes (Figure 3-41 - 44).

Secondly, we use boundary gap to provide a clear view of the hierarchy. Some experimental results are showed here.

Figure 3-41 displays an example of angular polygonal treemap in a pie convex container and Figure 3-42 in a ribbon shaped container. The Figure 3-43 shows a smaller data set with Triangular Treemap in a book-shaped concave container and Figure 3-44 displays a larger dataset in axe shaped concave container.

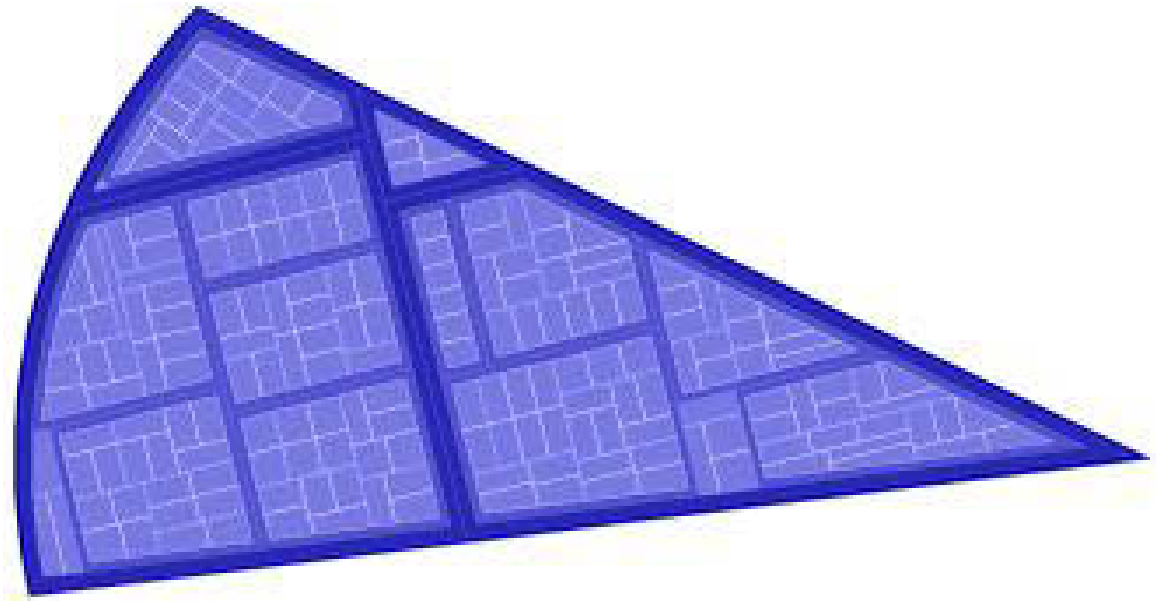


Figure 3-41 an extended example of angular polygonal treemap in a pie convex container

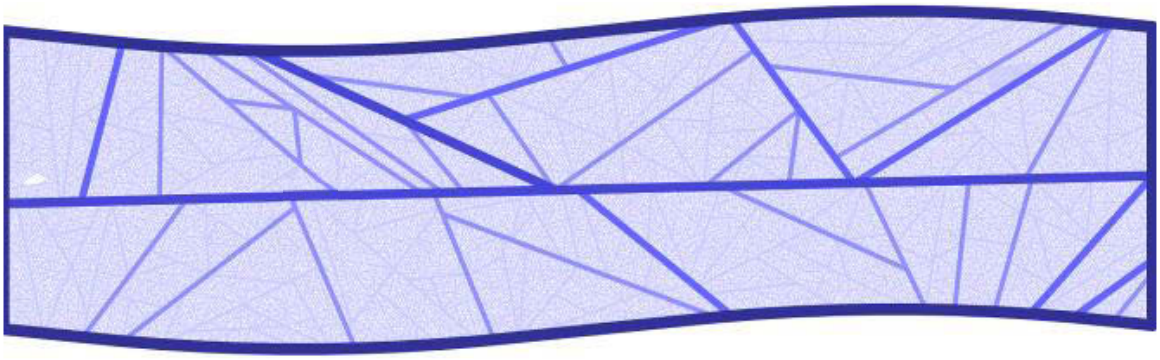


Figure 3-42 an extended example of angular polygonal treemap in a ribbon shaped container

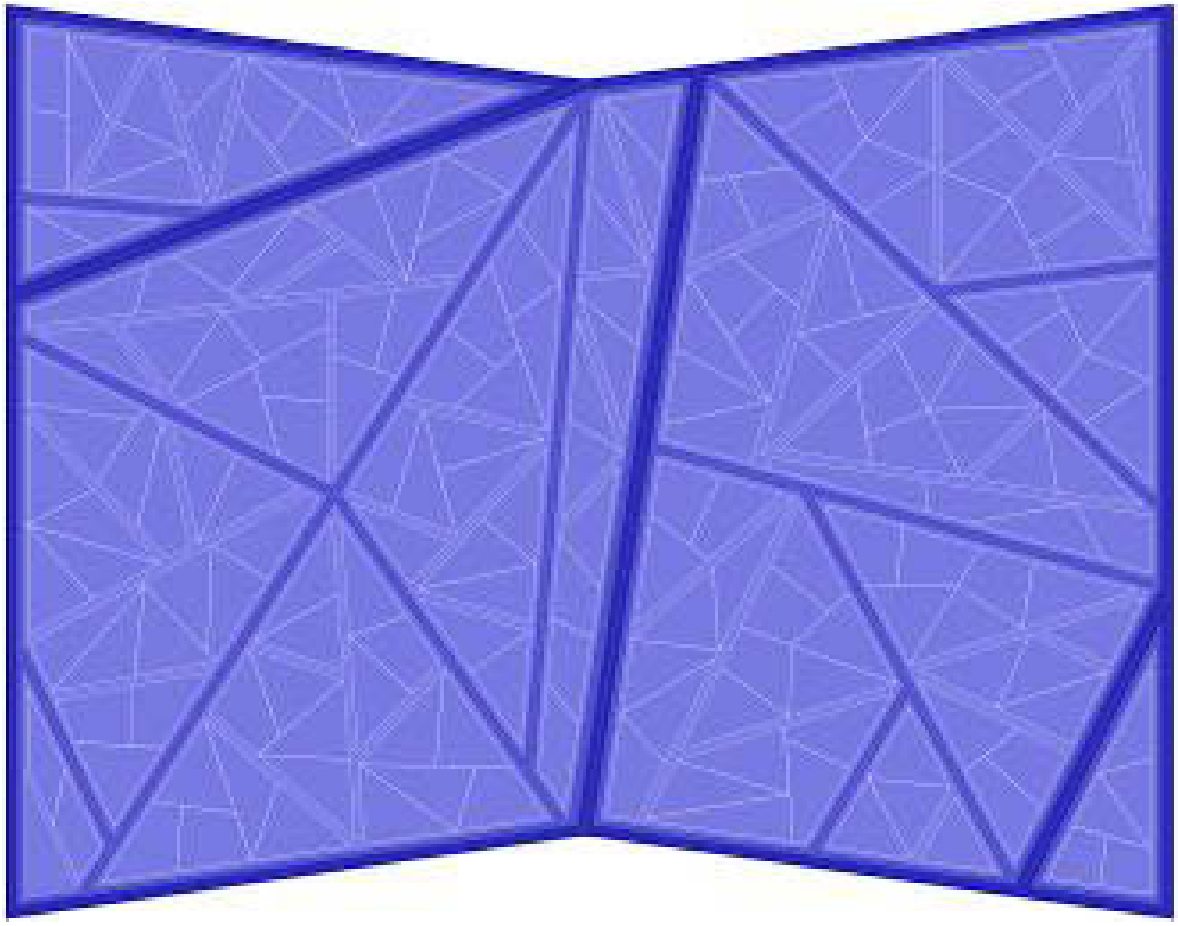


Figure 3-43 an extended example of Triangular Treemap in a book-shaped concave container

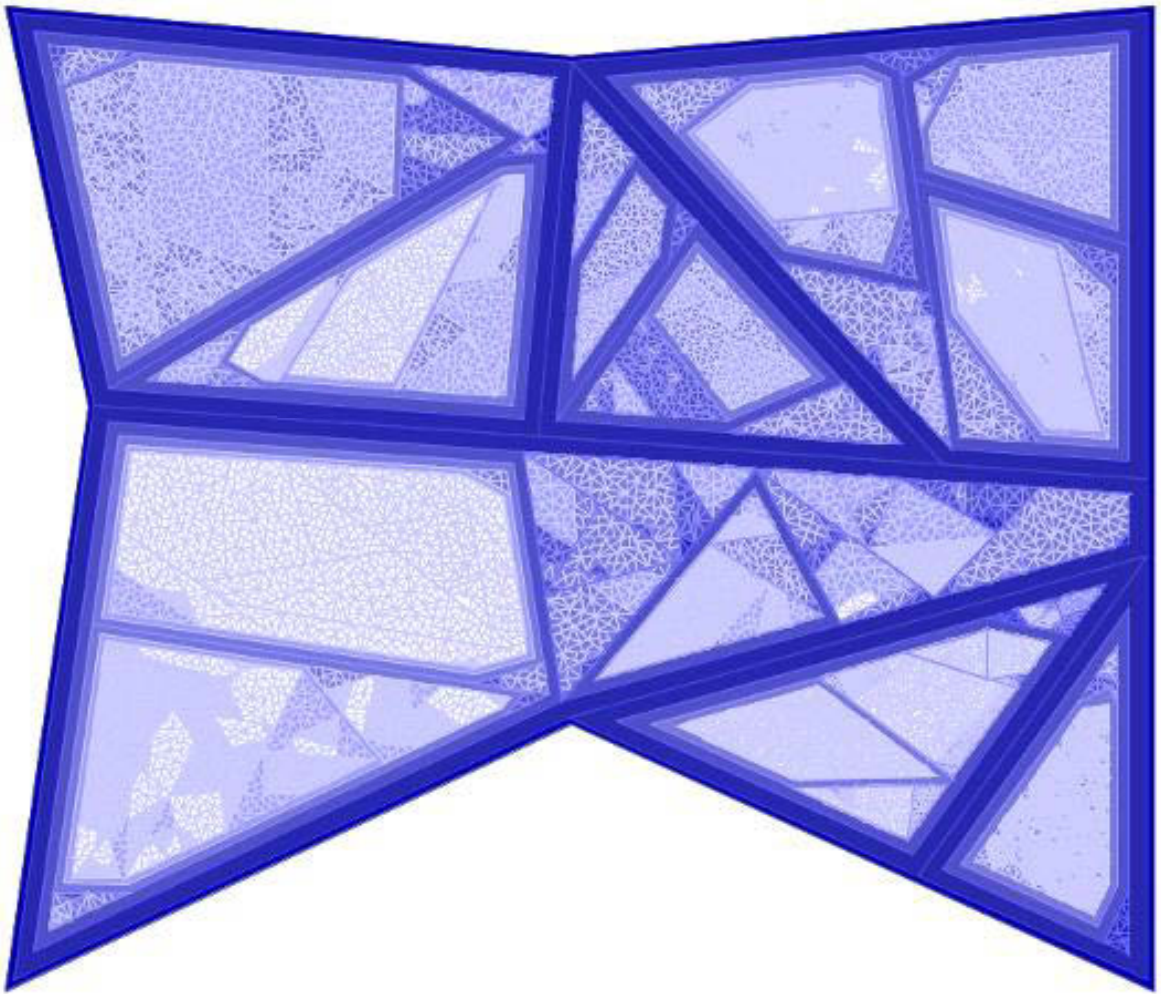


Figure 3-44 an extended example of Triangular Treemap visualizing a larger dataset in axe shaped concave container.

3.3.4 Container and Containment Control

Tangram treemaps are capable to offer flexibility on both container and containment. We show an example of visualization layout results, when both container and containment are controlled. Below is a visualization result of a data set with 16,000 nodes. The container is octagon convex shape and the containment is angular polygonal algorithm embedded in D&C rectangular treemap.

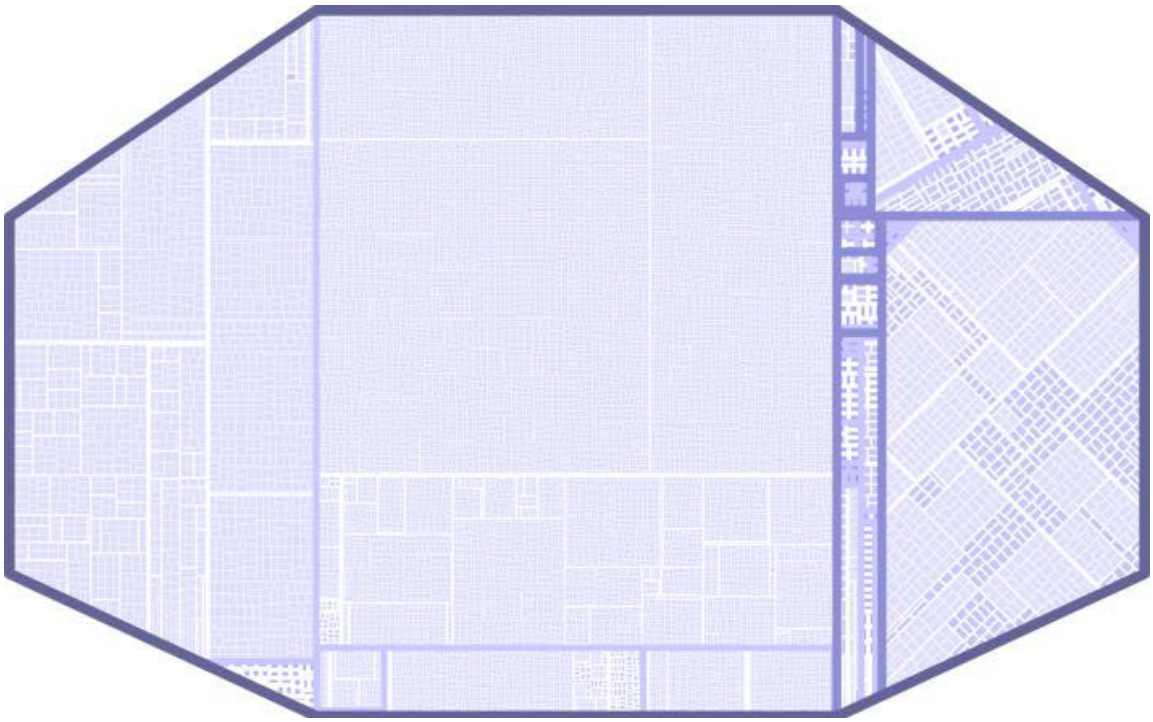


Figure 3-45 Container and containment control illustration: Visualization of a data set with 16,000 nodes using Angular Polygonal Treemap in an octagon container.

SECTION 3.4 SUMMARY

This chapter presents new tree map visualization, called *Tangram treemaps*. It can not only ensure the efficient utilization of display space, but also offers flexible layouts in various shaped container.

The chapter firstly mentions the original idea inspired from a puzzle game. After that, Section 2.2.1 goes through the evolution of the new approach and then Section 2.2.2 introduces the pipeline of visualization. Section 3.1 explains each procedures of the visualization pipeline in details. Section 3.2 discusses three approaches with partitioning process illustration, algorithm, and experimental results, according to the timeline of evolution. The method and implementation algorithms presented have the capability to visualize large relational structures in virtually any chosen shaped. After discussion of three approaches, section describes the utilization of three approaches in Tangram treemaps. To bring forth, Section 3.3 shows the advantage of Tangram treemaps in container control, containment control and both of them.

To summarize, the new approach breaks through the limitation of rectangular constraint. It has the ability to provide flexible layouts in various shapes container. It also has the ability to provide flexible child nodes, including vertical-horizontal rectangular, angular rectangular and polygonal layouts. These abilities make Tangram treemaps easier to combine with other enclosure approach in order to emphasize the importance and highlight the difference.

CHAPTER 4. INTERACTION MECHANISM

Enclosure partitioning has been widely used in real-world visualization applications, as it provides an alternative approach in presenting overall data structure with optimized space utilization. Especially, Treemaps (Burls, Huizing and Van Wijk, 2000), has shown its high applicability and commercial value in many areas such as finance analysis (Jungmeister & Turo, 1992), sport reporting (Jin&Banks, 1997), image browsing and software and file system analysis (Baler, Deussen & Lewerentz, 2005).

To enable Enclosure (or space filling) method more useful, the interactivity is the key. It is an important step involved in the Interactive Enclosure Visualization process is view navigation. Chaomei Chen (2004) states in his book "Information Visualization: Beyond the Horizon" that "navigation in a hierarchical structure involves moving from one node to another, along the existing hierarchical links in the structure. When the size of a hierarchy becomes large, it is desirable to enable users to have easy access to contextual information, as well as local details". One of the most important issues involved in navigation is that the users are always able to see (or have easy access to) contextual information. This allows users to maintain the perception of where they are and where they can move from during the navigation of large information spaces. This also assists users to make further decisions about where they should go next, as well as where they are, while interactively navigating through the information space.

SECTION 4.1 INTERACTION MECHANISM

Although the layouts generated in space-filling visualization are very efficient in terms of space utilization, the issues of "view-ability" to produce user-friendly interactive interfaces and the ability to explore data accurately are critical, especially for visualizing large and deep relational datasets. This is because in space-filling visualizations as well as other visualization techniques, it is hard to discern among nodes and edges, hierarchical levels, labels and other properties when thousands of items in datasets are displayed concurrently. Therefore, an efficient and effective navigation scheme, combined with a visualization which provides users with necessary knowledge where to go is essential when navigating large data structures.

The navigation scheme should enable users to interactively adjust views to reach the final view of a sub-graph, allowing them to obtain an optimal understanding of the data items and surrounding relational structure they are currently interested in, with minimal navigation steps. Without interaction mechanisms that let users explore data, capabilities of visualization tools cannot maximize human capabilities to perceive and understand complex and dynamic data.

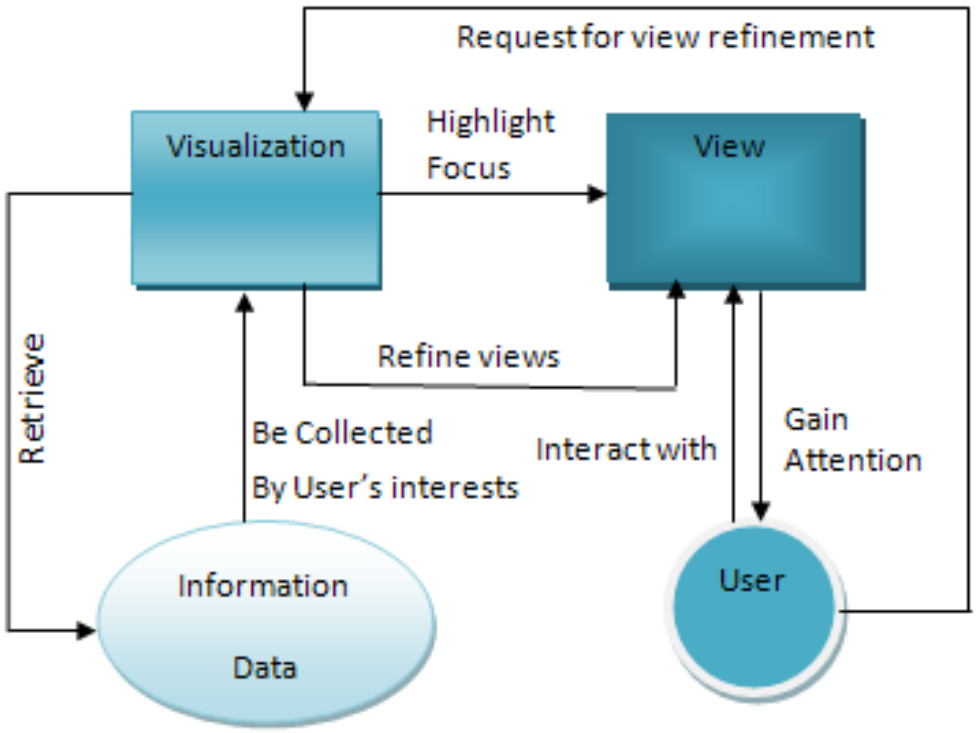


Figure 4-1 The concept map of Interaction control for Tangram Treemaps

In interaction control for Tangram Treemaps, the interaction is applied in navigation process that guides users to progressively refine their focus views to reach the final target view. Figure 4-1 is the navigation concept map in the process of data retrieval. With the support of available information, users may modify view transformation and browse and switch the data by different granularity based the details degree of user requirements. The interaction control provides two methods which take advantage of differentiation in size and differentiation of shape to provide focus and background information.

SECTION 4.2 INTERACTION METHODS

4.2.1 DIFFERENTIATION IN SIZE

Up to now, there are many interaction (or navigation methods) techniques, such as overall + details view (Plaisant et al, 1995), semantic-zooming (Baker & Erik, 1995), focus + context viewing (Stasko, 2000) and fish-eye view (Schaffer, 1996), etc. that have been proposed in

the traditional node-link diagram based visualization, and only very few techniques are applied effectively in enclosure visualization.

"Drilling-down + Semantic –Zooming" is the most commonly used interactive navigation technique. This quick and simple navigation scheme is widely established in current operation systems. (Baker & Erick, 1995) To be able to explore large hierarchies with detailed viewing, we use the semantic zooming technique (Herman et al, 2000) to zoom in the detail of a particular part of the hierarchy based on user's interest at a time. We use the zoom to view the detail of this substructure of the graph. Users can zoom out to obtain the context view. Positions of all vertexes in a sub-tree will be recalculated at a time in corresponding to "zoom in" or "zoom out" each time.

The typical user interaction for locating a node is clicking on a selected node and moving to the sub-structure rooted and its sub-structure and users can recursively select another sub-graph or tree until reaching the final substructure that contains the target node. This form of interaction is analogous to zooming into a region of interest with each step of the zoom operation being a sub-structure in the hierarchy.

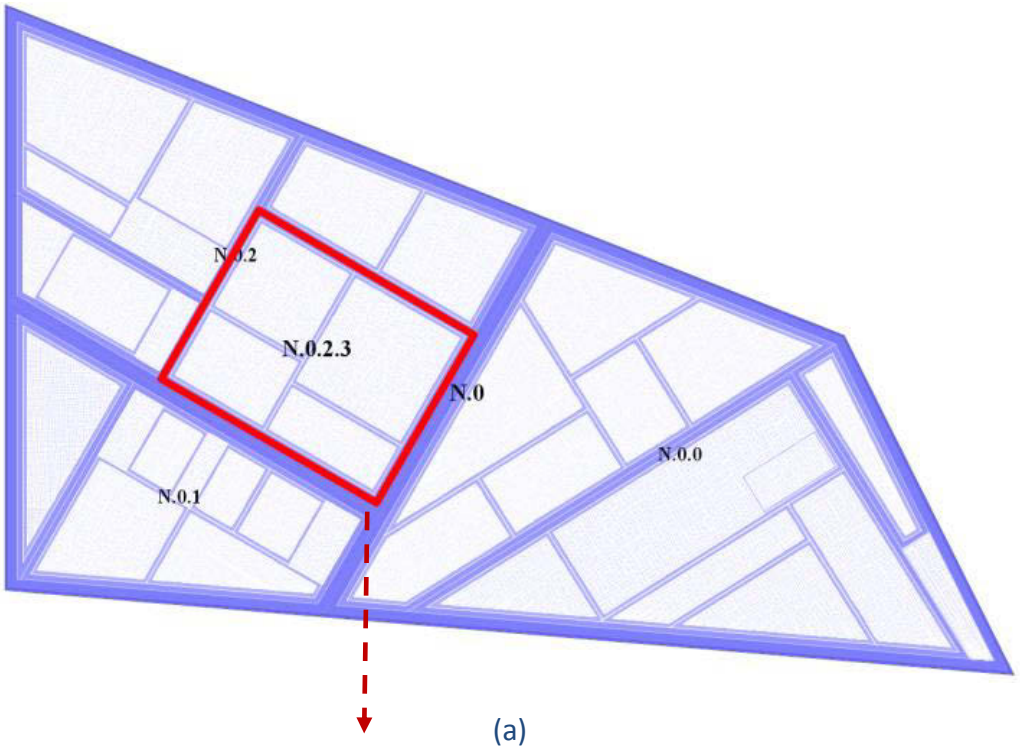
Three steps in location or searching an object are:

1) Perceive relational structure from the view

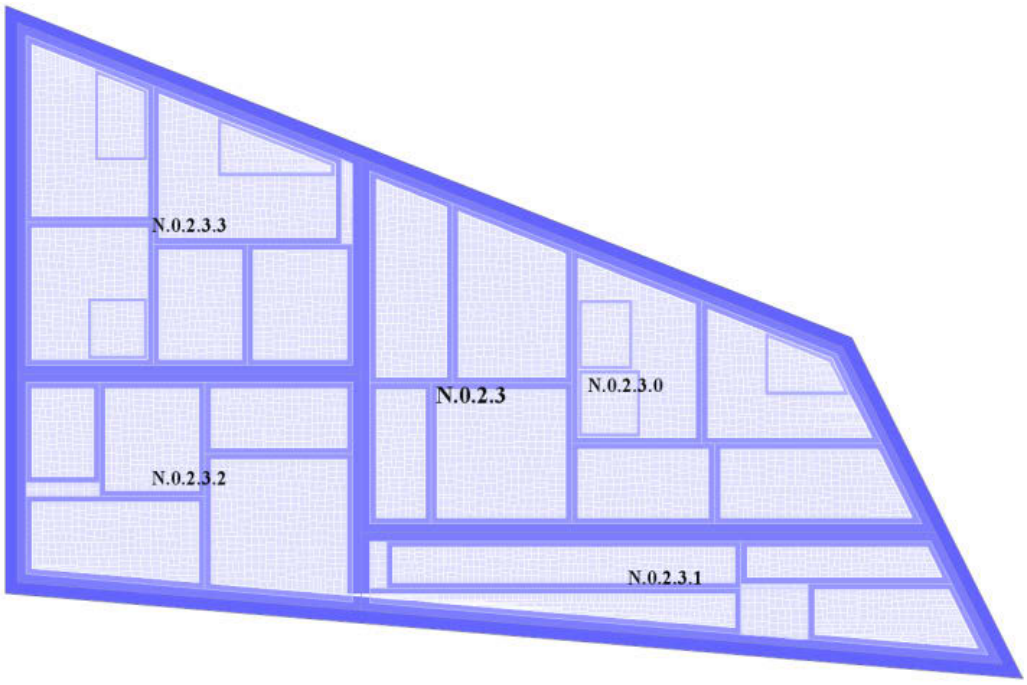
2) Make decision about where to go

3) Click on a selected objective and move to the sub-structure until locate the target.

For instance, in **Figure 4-2** a and **Figure 4-2** b, when being selected as focus view by a mouse-click based on user's interest at a time, the selected sub-structure expands to the entire display area. When being selected by a mouse-click, the selected sub-structure expands to the entire display area.



(a)



(b)

Figure 4-2 **The Interaction method 1: Figure (a) is an example of the navigation using semantic zooming. Figure (a) shows the visualization of the entire data set with approximately 43,000 nodes, and Figure (b) shows the focus view when the zooming is applied to the selected node (highlighted with the red boundary at the** Figure 4-2 **(a)).**

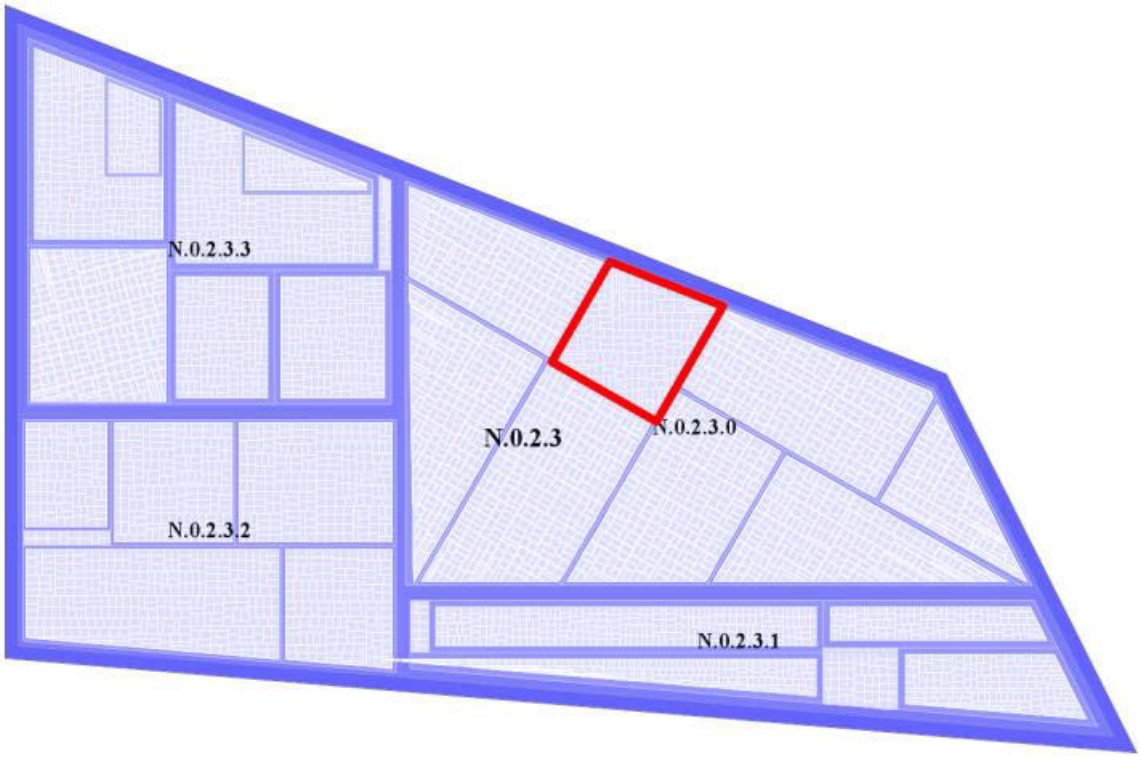
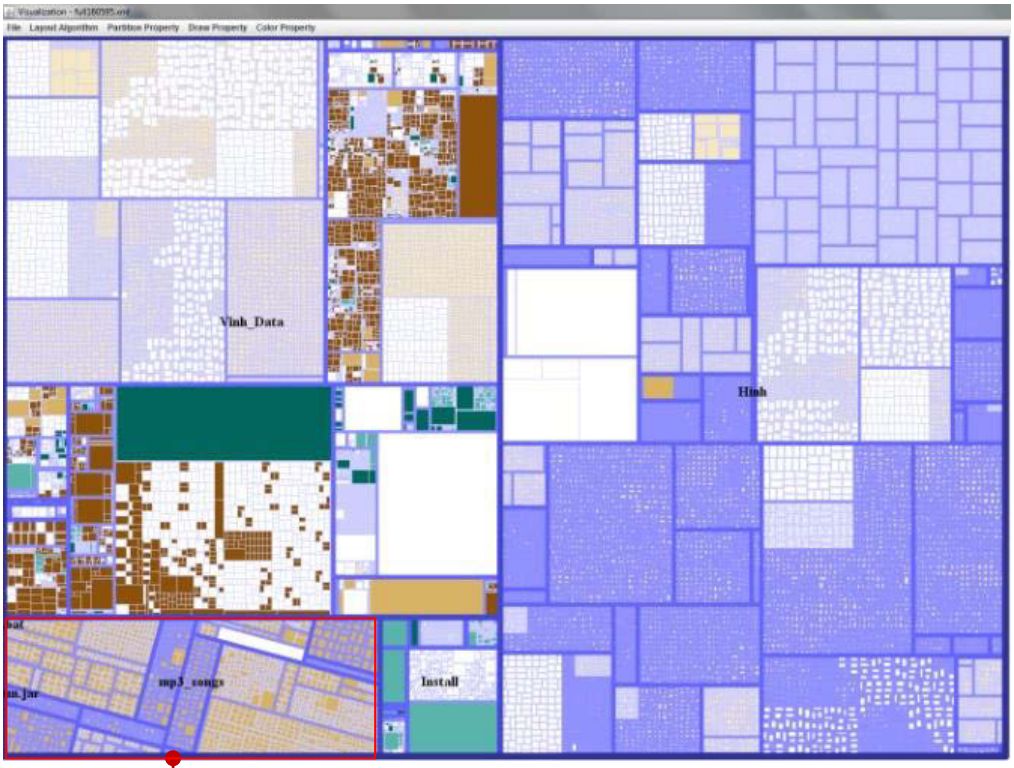


Figure 4-3 The Interaction method 2: An example of the layout at a navigational stage (same sub-structure as Figure 4-2).

4.2.2 DIFFERENTIATION IN SHAPE

Based on Weighted Divide and Conquer algorithm, Tangram Treemaps has the ability to present the data with Differentiation in shapes. In this interaction method, the shape property of data elements is used to indicate and prioritize the focus. It enables active visual thinking based on user experience. For instance, angular polygonal Treemap provides a function menu for quick rotating any particular sub-hierarchies. Users can use drop box to rotate particular sections, based on user's own interests in particular task. This property opens a new opportunity to generate a variety of layouts within the available visualization without affecting the overall display, enabling by partition angles at any particular sections. The use of angular partitioning could enhance the visibility and highlighting of one or multiple sections in the large structure. Figure 4-3 illustrates the visualization of the same sub-structure at the **Figure 4-2** b at a navigational stage. In this figure, the structure of the node N.0.2.3.0 has been rotated by an angle. In addition, a sub-structure in its sub-structure (highlight with the red colour) is also further rotated.

Differentiation of shape not only attracts attention but also helps users to track his or her mental map during exploration and also to compare one objective with another. During the navigation process, the interaction can be stored with the actions of highlighting and the history of the highlighting can potentially track the change process. For example, refer to Figure 4-4, user selected "Mp3- songs" for the focus view in first step and move on in subfolders of "Mp3- songs"; user reviews "English" in second step. The interaction control records the trace of steps. All the viewed folders were highlighted with rotation. This function is particularly important for collaborative work, when multiple analysts work on one single platform. What is more, in most cases, analysts and decision maker may be different person. The Highlighting acts as the bridge which helps decision maker use the highlighted results to make final decision.



(a)

(b)

Figure 4-4 Interaction process illustration: A series of screen shots for the views in angular polygonal treemap application for level 0 (a), level 1 (b), and level 2 (c) for the organization of file systems.

System can also implement data mining algorithms based on user's requirements and domain experts' recommendations. The highlighting can also be performed automatically with the predefined property at each sub-structure. Highlighting lets analysts concentrate on the focus of interests and analysis question. In that way, this interaction method can also act as intelligence tool offers visual suggestions and provides recommendations for analysts to consider.

SECTION 4.3 CONCLUSION

Interaction control assists visualization to express data and presents the focus of interest so that user can communicate with data. On data level, Interaction method can simply only attract focus of particular data. On information level, it provides visual navigation and helps user allocate right information or retrieval data. On knowledge level, it supports visual communication and cognitive thinking involved in decision making. This interactive technique helps analysts take action with pre-attentive cues in visual analytic environment.

The usability test in Chapter 6 proves that Tangram Treemaps in real scenarios successfully enhance the visibility and highlighting focus of interests by the differentiation of various sub-hierarchical structure layouts. The positive results have demonstrated its efficiency and effectiveness in locating and identifying tasks.

However, as interaction control is directly involved in analyst's cognitive thinking. Designing interactions require us to study how people develop and use visualizations in the larger context of the work they do. The interaction should be built in specifically for particular domain and case.

CHAPTER 5. TECHNICAL EVALUATION

This chapter evaluates our approach according to a set of design guidelines. The objective 1 of this research is to simplify visualization algorithms in order to implement them into the real-time applications by reducing the Computational Complexity (CC). Hence, we need to accurately measure the Computational Complexity of these algorithms and then compare them with other CCs of the traditional Treemaps. Note that in our evaluation we only consider Computation Time (CT) to measure the CC. The objective 2 is to design a layout which meets treemap design requirements. Therefore, we will investigate Tangram Treemaps in three areas: Computational Time (Section 5.1), Aspect Ratio of Rectangular Areas (Section 5.2) and Proximity of Nodes positions (Section 5.3), in comparison with other standard treemaps techniques.

SECTION 5.1 COMPUTATIONAL COMPLEXITY

This section firstly defines and measure the computational complexity of the partitioning (section 5.1.1) with general polygon shape and then the computational complexity with a variety of shapes (section 5.1.2). Finally we compare the computational time of our approach with other existing Treemap techniques (section 5.1.3).

5.1.1 Computational Complexity of the Partitioning

The partitioning process of Treemaps involves an in-order traversal through the tree. Therefore, it takes linear steps to visit each vertex in the partitioning process. The computational complexity of our approach is defined as followings.

In a geometrical polygon, whose boundary has been defined as a polygon *P*, a parent vertex be divided into n child vertices $\{P(v_{k+1}), P(v_{k+2}),\ldots, P(v_{k+n})\}$. The partitioning process of all child vertices $\{P(v_{k+1}), P(v_{k+2}),\ldots, P(v_{k+n})\}$ is a divide and conquer process that follows a generic pattern to tackle the list of vertices of size n by recursively solving, say, a sub-problems of size a = b = 2 and then combining these answers is (Dasgupta et al, 2006):

$$T(n) = 2T(\frac{n}{2}) + f(n)$$

Equation 4 Partitioning time complexity formula

Where *f(n)* is the cost of the work done outside the recursive calls, which involves with either algorithm: ***D&C Partition ()*** or ***AngularResolutionConstraint()*** or ***AngularDivide()***

Since the iteration (the 'while' loop) times is equal to the number of side vertices of the polygon *P(v)*, the computational time spent for partitioning of *P(v)* by using Algorithm- ***D&C Partition ()*** is equal to *O(n) accordingly*.

The ***AngularResolutionConstraint()*** algorithm also includes moving the vertex point along the side if the angle does not satisfy the condition. It repeats *k* times of the *D&C Triangular Algorithm* and *k* is also a constant. Therefore, this algorithm's computational cost is also *O(kn)*.

Similarly, the ***D&C Partition ()*** algorithm repeats *k* times on the side in order to find the solution. As all processes are constant, the algorithm's computational cost is *O(kn)*

5.1.2 COMPUTATIONAL TIME IN DIFFERENT SHAPES

The **experimental environment** is described below. Firstly, Java 1.6 with Eclipse Platform was used to develop the prototype (the Java program) that implements our partitioning algorithms. The Java program was executed on a Personal Computer with the CPU: AMD Phenom 2.1Ghz and 2GB of RAM. Secondly, the Computation Time (CT) is the time of partitioning process, excluding the file reading, rendering and displaying time for testing; we ran the Tangram Algorithm and Tangram Algorithm with Angular Resolution Constraint in the above environment, with a list of **data sets** which have an increased number of vertices, from 272 to 12,000 to 122,000.

We also employ the algorithms on a variety of partition shapes. For convex shapes, we included triangle, rectangle, hexagon, pentagon, and octagon. For concave shapes, we used the polygons with 6, 7 and 8 vertices. Table 5-1 summarizes the computational time spent for completing the partitioning of a list of datasets with different container shapes, including convex and concave. From the Table 5-1, we can see computation time increase depending on the increase of the size of data sets rather than the differences of container shape. Based on the testing results, we designed the evaluation of computational time with other techniques in section 5.1.3 under the condition of increasing data density only.

Table 5-1 The computational time (in milliseconds) of our Tangram algorithm and the Tangram with angular resolution constraint on a variety of data sets and shapes of the container

Number of Vertices	Container	TANGRAM (D&C Triangular)	TANGRAM with Angular Constraint
Convex polygons			
272	Triangle	8	12
2400	Triangle	28	38
2400	Rectangle	29	49
12,000	Rectangle	81	128
16,600	Hexagon	87	161
43,200	Pentagon	205	328
122,000	Octagon	389	691
Concave polygons			
2400	8 vertices	29	38
12,000	6 vertices	93	131
16,600	7 vertices	108	237
81,000	6 vertices	271	791

5.1.3 Computational Time Comparison with Other Techniques

A comparison of CT with other Treemap methods is conducted under the **same environment** and we calculate CTs for five different treemap methods, including Slice-and-Dice Treemaps, Squarified Treemaps, Space-Optimised Tree, Tangram Treemaps, and Angular Polygonal Treemaps. The evaluation of CT Comparison was conducted in the same rectangular container with 1000x1000 pixels, containing a variety of trees with different sizes.

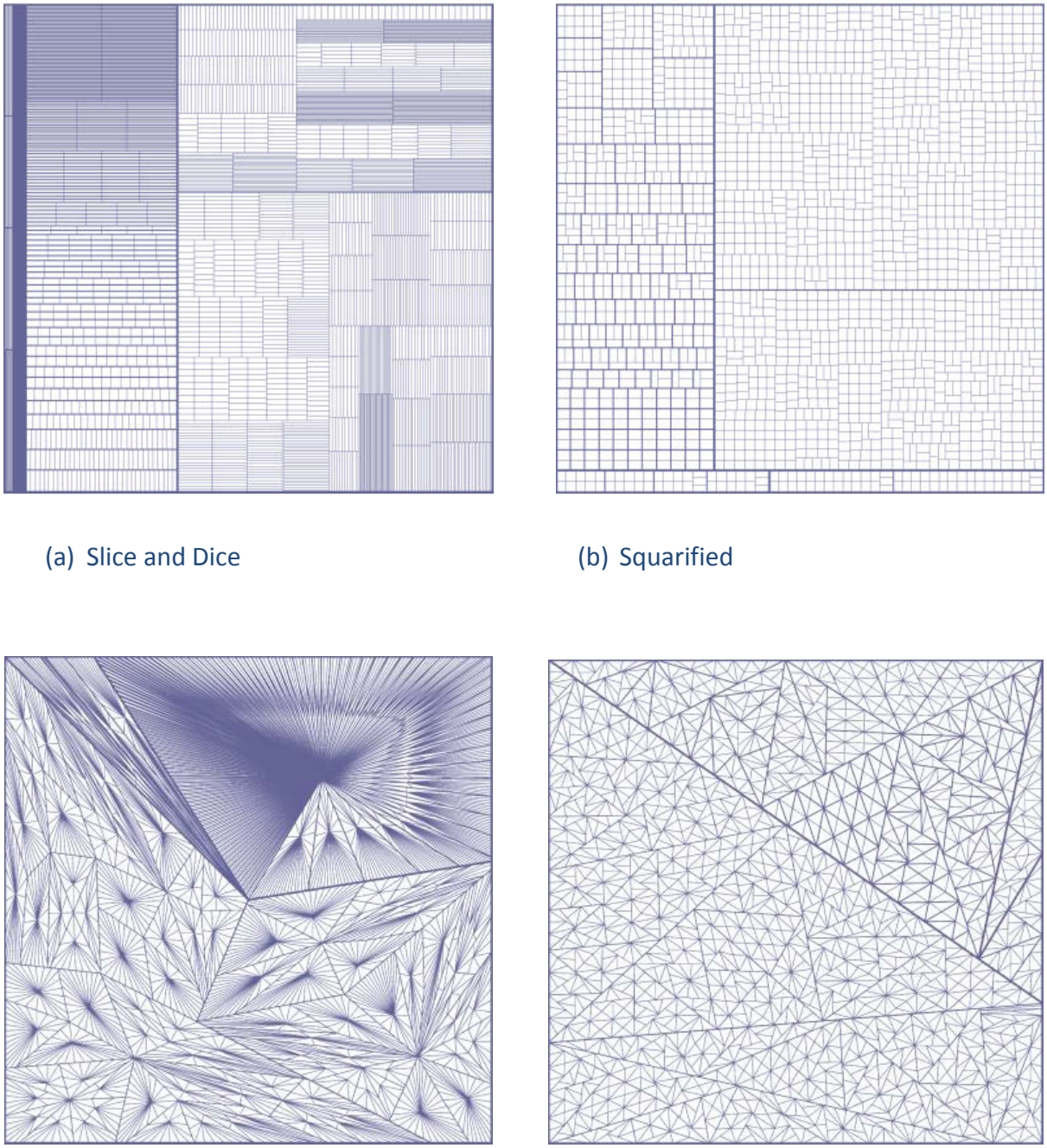
Table 5-1 illustrates the results of comparison among five different treemap layouts, including Slice-and-Dice Treemaps, Squarified Treemaps, Space-Optimised Tree, D&C Triangular Treemap, and Angular polygonal Treemaps, using the same data set with approximately 2400 vertices. The *Voronoi Treemaps* partition algorithm was not considered in our experiments due to its very low computational performance.

Table 5-2 illustrates the comparison results in computational time between Tangram algorithm and other partition algorithms, including *Slice-and-Dice Treemaps*, *Squarified Treemaps*, and *Space-Optimised Tree* for each different **datasets with from 272 vertices up to 650,000 vertices.**

Table 5-2 indicates that D&C Triangular Treemap and angular polygonal treemap are very close to other three simple algorithms. The running time of Tangram algorithm with angular constrain are relatively longer than the other three algorithms, but competitively close. This evaluation results prove that Tangram treemaps are capable to be applied into real time application. Taken in account its real-time performance, Tangram *Treemaps* could be a very effective and flexible interactive tool for producing visual representation of large data structures and be adopted into a wider range of applications, within different boundary of polygonal shape of containers.

Table 5-2 The computational time (in milliseconds) of the Tangram and the Angular Resolution Constraint algorithms in comparison of Slice-and-Dice Treemaps, Squarified Treemaps, and Space-Optimised Tree on various data sets using the same rectangular container.

Number of Vertices	Slice & Dice	Squarified	SO-Tree	D&C Triangular	With Angular Constraint	Angular Polygonal
272	2	4	7	8	12	80
2400	7	21	24	28	39	120
16,600	27	138	156	194	238	220
43,200	31	38	92	149	443	234
81,000	49	61	155	266	783	343
122,000	70	87	225	387	711	186
650,000	319	2903	1010	4029	5730	1505



(a) Slice and Dice

(b) Squarified

(c) Space-Optimized

(d) Tangram

Figure 5-1 Compared treemap techniques illustration: visualization outputs of different techniques for a data set with approximately 2400 vertices.

SECTION 5.2 ASPECT RATIO

Previous research on treemaps primarily focused on developing new layouts algorithms that attempted to optimize the Aspect Ratio. To examine treemaps effectives, we usually use AR to assess the quality of Treemaps. Bederson et al.'s work (2002) assesses the ordered treemaps with other treemaps algorithms using metric of average aspect ratio. The results imply that squarified treemaps had the lowest average aspect ratio from 1.19 to 1.75 and slice and dice treemaps had the highest aspect ratio (from 26.10 to 30.41). Heer & Bostock's results (2010) indicate rectangles with aspect ratio closing to 1, leads to increased errors. Kong's work (2010) analysed the impact of aspect ratio on judgment accuracy. They found accuracy improves when rectangles for comparison have diverse aspect ratios. Their results suggests judgment accuracy increase when the layout optimizes towards a 3/2 aspect ratios.

The computational environment is the same as described in Section 5.1. The same rectangular container (size of 1500x1000 pixels) was used entirely in this experiment. We also define the average aspect ratio of a treemap layout as the un-weighted arithmetic average of the aspect ratios of all leaf-nodes (Bederson et al, 2002). We ran the experiments on several data sets with various sizes and properties, including both artificial data sets and real file system data sets.

This section evaluates the quality of aspect ratio of our D&C Triangular algorithm in comparison with *Slice-and-dice* and *Squarified Treemaps*. Because the traditional Treemaps use vertical-horizontal partitioning approach, we use D&C Rectangular to provide matching layouts. This evaluation follows the outcomes of the usability studies in Kong's work (2010) that indicate the accuracy of the comparisons with 3/2 (or 2/3) aspect ratio rectangles is better than perfect squares or skinny rectangles. Although it is more useful to carry out further studies to evaluate more specifically the optimal aspect ratios in the range of 3/2 and 1/1, we use the bench mark of 3/2 aspect ratio in our evaluation.

The Table 5-3 shows the average aspect ratios of three algorithms from data set which contains 38 vertices up to 160,600. The average aspect ratio on various data sets shows that *Slice & Dice* layout has diverse average aspect ratio (from 1.6 to 11.1). Squarified

treemaps has average aspect ratio (from 1.3 to 1.8). D&C Rectangular treemaps has most stable average aspect ratio between 1.5 to 1.7, close the bench mark of 3/2.

Based on the assessment over design parameter of aspect ratio, D&C Triangular treemap should be the preferable choice over slice and dice treemaps and Squarified treemaps.

Table 5-3 Average aspect ratios of layouts

Dataset	Slice & Dice	Squarified	D&C Triangular
38	1.6	1.3	1.6
170	3.4	1.5	1.6
620	2.6	1.5	1.6
2400	6.3	1.4	1.5
4085 (real data)	7.1	1.6	1.7
12,000	6.6	1.4	1.6
18579 (real data)	8.3	1.8	1.7
81,000	2.9	1.4	1.5
143,100 (real data)	8.3	1.8	1.6
160,600 (real data)	11.1	1.7	1.6

SECTION 5.3 PROXIMITY OF NODE ORDERING

As discussed above that most of the previous Treemap research were focusing on developing algorithms to create more useful enclosure visualization by controlling the aspect ratios of the rectangles that are the basic elements of a treemap. However, while these algorithms can certainly improve the visibility of small items in the visualization, they introduce instability over the time flow when displaying dynamic data. They failed to preserve the order of the underlying data, and created layouts those are hardly to be used for visual navigation or exploration. Moreover, even occasional abrupt changes mean that it is hard to find particular data items on the treemaps by memory, decreasing efficacy for long-term users.

The display of dynamic data sets may affect visual query and navigation. To solve this problem, Bederson & Shneiderman proposed an ordered treemaps to maintain the ordering of underlying data. To guarantee the quality of treemaps layout, the design parameter of proximity should be also examined.

We evaluate the quality of treemap layouts by a metric with node positions in relation to their structural orders. The evaluation environment is same as section 4.1. The definition of Proximity is illustrated below.

If two nodes have the same parent and stay next to each other, they will be placed closely together in the visualization. We only apply the evaluation on direct child nodes in the same hierarchical level and have the same parent node. For iteration, we calculate the distance between two consecutive nodes. The distance between two nodes $v_1(x_1, y_1)$ and $v_2(x_2, y_2)$ is calculated by the following formula. The same rectangular container (with the size of 1500x1000 pixels) is used in this experiment. The smaller the distance (in pixels) between a pair of nodes, the better indication of the proximity of node ordering in the layout. The distance between two nodes is defined below:

$$d(v_1, v_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Equation 5 Distance between two nodes formula

Table 5-4 Average distance of proximity

No. of child nodes	Slice & Dice	Squarified	D&C Triangular
43	35.2	274.83	220.9
94	24.8	248.0	162.3
157	9.5	230.2	127.5
383	3.9	157.1	80.5
579	2.6	135.0	66.8
1001	1.5	121.3	51.2

We have run the experiments on several real (file systems) data sets. We only chose the files and folders at the same level and at the same parent folder in the evaluation. Table 5-4 shows the average of proximity for three treemap algorithms. For slice & dice, average of proximity varies from 1.5 to 35.2; for squarified from 121.3 to 274.83 and for D&C Triangular, from 51.2 to 220.9.

The Results illustrated in Table 5-4 strongly suggest that the *Slice & Dice* method can produce the best average of distance, while the D&C Triangular algorithm generates layouts with shorter range for proximity, which is better than the *Squarified* method. In order words, our algorithm improves significantly the metrics in node positioning in relation to their structural orders, rather than Squarified treemaps.

In summary, the evaluation results we described in this chapter have proved that the speed of Tangram Treemaps is compatible with other standard treemaps. It made Tangram treemaps more suitable for dealing with real time applications of wide domains. Tangram has also met two important criteria, Aspect Ratio and Proximity, according to graphical perception design guidelines. TANGRAM is close to optimal aspect Ratio 3/2 and improves its proximity in comparison with squarified treemaps. Based on the control over, both

aspect ratio and proximity of node ordering, Tangram Treemaps generate more effective perceptual layout.

CHAPTER 6. USER STUDIES

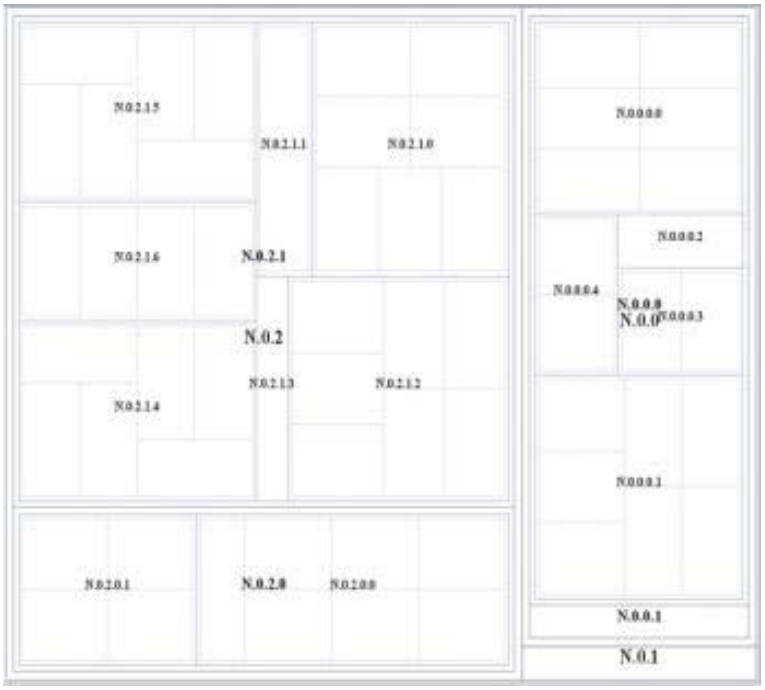
Tangram Treemaps represent hierarchical data structures using containment approach which enclose each child node within its parent node(s). Tangram Treemaps also encode values of data using area. Therefore, the Tangram treemaps can be used for both hierarchical exploration and value comparison; two typical categories of operation in enclosure (tree) visualization. Particularly for Tangram Treemaps, the containment we are examining conducted in two ways: firstly, treemaps with rectangular container have the control of changing shapes of child; secondly, treemaps can tailor representation into different shapes containers, with the control of changing shapes of child.

To further investigate how well Tangram Treemaps work in the scenario based tasks during the visual analysis process, we have conducted three user studies to compare Tangram Treemaps with other typical rectangular treemaps. According to the typical categories of user tasks, first study is to conduct controlled experiments in locating the object(s) in the structure as a typical task of hierarchical exploration; Second study combines hierarchical exploration and value comparison; Third study moves to assess the effect of rectangular area size judgment, as one common task of visual comparison.

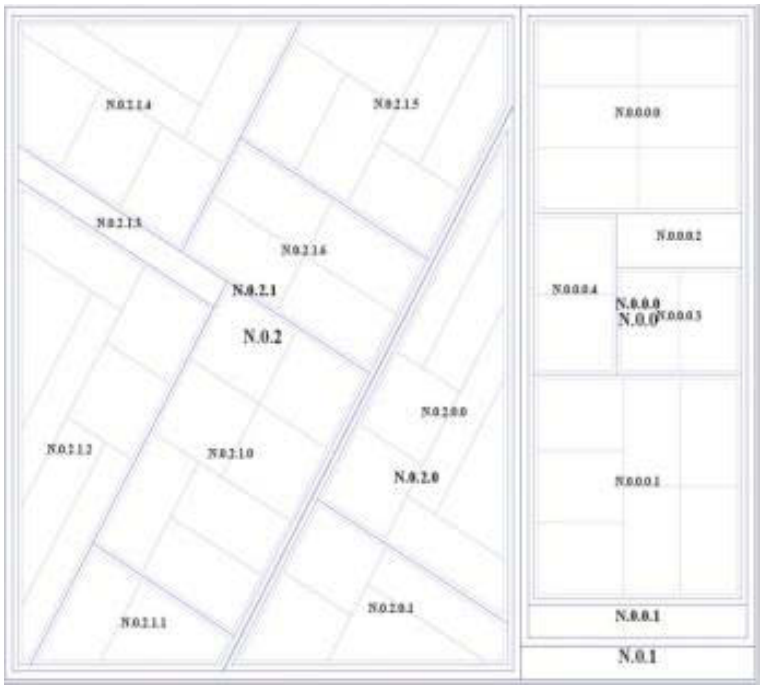
Specifically, the first preliminary study is described in Section 6.1 that is to test the performance of D&C Rectangular Treemap which has the ability to change the orientation; Section 6.2 describes the second formal user study that is to determine the performance of Tangram treemaps with different containment: The first experiment is to study Tangram treemaps with the capability of changing the shapes of child nodes, on the support of data mining query; the second experiment to test the performance in rectangular area comparison task and the third experiment is to valid the needs and benefit of various shaped container in certain analysis scenarios. Third study in Section 6.3 is to investigate how orientation affects visual comparison of area size.

SECTION 6.1 PRELIMINARY STUDY

The purpose of preliminary usability study is to investigate whether Tangram Treemaps can better perform highlighting functionality; that is to access how well Tangram Treemaps could assist users to quickly identify emphasized (or highlighted) visual objects in comparison with other rectangular treemaps in performing certain analysis tasks. A preliminarily usability study has been conducted with a control group of 8 subjects for the application. The designed experiments were performed with 28 datasets and 112 scenarios. The datasets are categorized by four different types of datasets, and each type has 7 alternative datasets, with increasing number of hieratical level and complexity of data. In order to prevent users from remembering labels, experiments use generic labels which contain characters and numbers. The dots between characters and numbers are used to distinguish the levels. For example, N0 presents a name for a node in level 0, N0.0 is contained in the one level lower of N0 and N0.0.0 is located in the second lower level of N0.



(a)



(b)

Figure 6-1 Preliminary study sample: Illustration of the traditional rectangular treemaps (a) and the Tangram Treemaps (b).

Before conducting the usability study, we introduced the basic knowledge of hierarchical data structure and the current literature of visualization methods, particular the treemap approach to the subjects (or subjects). We explained to our subjects about the way of measurement that how people perceive treemaps. Then subjects were given two visual presentations for each application; 1) the traditional rectangular treemaps and 2) Tangram treemaps (See Figure 6-1). They were then asked to explore particular data item(s) in each visualization method, targeting to find out the difference in using these methods. To help subjects to be familiar with the trial, we show a few example trials and testing data to them; briefly explaining the application features and control procedures. Subjects were asked to try three trials for three particular tasks:

- **Task 1** - locating particular visual objects (nodes) with given labels (names) in the lowest level of the hierarchy;

- **Task 2** - locating particular visual objects (nodes) with given labels (names) in lower level 2 of the hierarchy; and

- **Task 3** - identifying two similar visual objects (nodes) with given labels (names) in lower level 1 and level 2.

During the usability study, the order of visualization methods they used varied randomly from subject to subject, to prevent order bias. In each method, subject was presented with an existing output and asked to answer the questions in the task. Subjects were asked to complete 7 questions for each task, in sequence of increasing difficulty degree. The completion time for each question was recorded. The average completion time for each task was measured. In the end, subjects were interviewed for preference and also encouraged to give feedback about further improvements.

The overall inter-subject consistency rate was very high, although, as expected, there was one case where subject behaviour was unique far from the average. This case has also been excluded in the result analysis. As result, the subject performances in three tasks for Angular method were all higher than traditional treemaps.

Figure 6-2 shows the average completion time of three tasks for both methods. Comparing with traditional treemaps, the time users using Angular Polygonal treemap spent for locating one node in level 1, has been saved by 17%, by using Angular polygonal treemap ; users efficiency in Angular Polygonal Treemap application for locating particular node in deeper level 2, increased by 19%. The advantage of Angular Polygonal Treemap is greatly reflected in task 3. When users need to identify two different focuses in different levels of the hierarchy, the performance by using Angular Polygonal Treemaps has been improved by 29%. Last but not least, all users selected Angular Polygonal Treemaps as preference in the interview session.

With the promising results from preliminary study of angular Polygonal treemaps in first category of visual analysis tasks, we had more confidence to move to a formal user study of Tangram Treemaps in both categories of visual analysis task.

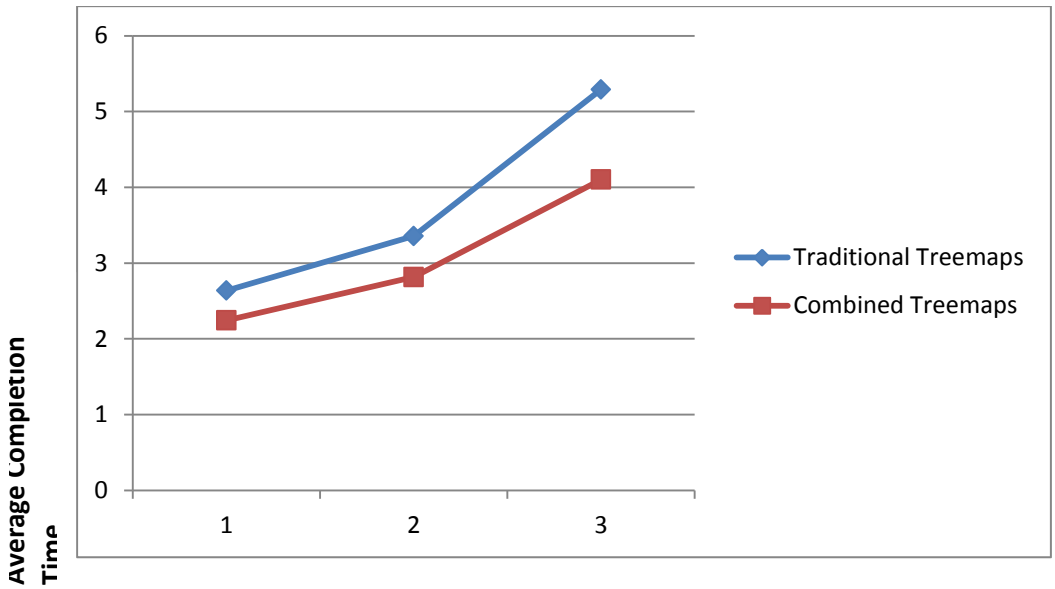


Figure 6-2 Preliminary study results: Line chart of the average completion time comparison between tangram treemaps and traditional treemaps in three categorized tasks.

SECTION 6.2 FORMAL USER STUDY

The major goal of the formal user study was to assess the effectiveness of Tangram Treemaps in achieving typical visual data analysis tasks. In addition, we considered the assessment of effectiveness of dynamic shaping of containers in visual exploration tasks. To evaluate both aspects, the usability study should be extended to assess the effect of the Tangram Treemaps (inside of rectangular container) and also assess the effect of non-rectangular container (outside of Tangram treemaps), which includes Experiment 1 Data element shape variation and Experiment 2 Rectangular Area size Comparison, and Experiment 3 Representation container Boundary shape variation.

This section includes Section 6.2.1 control groups of the subjects; Section 6.2.2 the hypothesis, which the experiments were set up on; Section 6.2.3 first and second and third Experiments; Section 6.2.4 user preference, Section 6.2.5 performance results, Section 6.2.6 user preference and feedback , and finally Section 6.2.7 discussion of results.

6.2.1 CONTROL GROUP

We conducted three controlled experiments to evaluate the effectiveness and efficiency of Tangram Treemaps for visual exploration tasks. We compared *Angular* Polygonal *Treemap* with other two treemap techniques, including 1) *Slice and Dice Treemaps* and 2) *Squarified Treemaps*. The user group has 20 subjects (aged 25 to 35, 14 males, and 6 females) from local university and industry. The subjects were from the fields of computer science, economics, business, and finance and art design. All subjects indicated that they had experience with visual data analysis.

6.2.2 HYPOTHESIS

We formulated the following hypotheses for these experiments

[H1] ***Tangram Treemaps have a positive impact for object locating tasks.***

Based on the preliminary study results, we assume that Tangram Treemaps would have better performance results than other conventional treemap techniques, in locating visual objects, especially when locating for multiple objects that are located in different hierarchical levels.

[H2] ***Distinction of container shapes supports human cognition process in certain scenario of shape identification and visual navigation tasks.***

If this hypothesis be supported, it will motivate the future work in combination of non-rectangular and rectangular approach. With evidence provided, the new approach will be effectively adopted for visual analysis tasks.

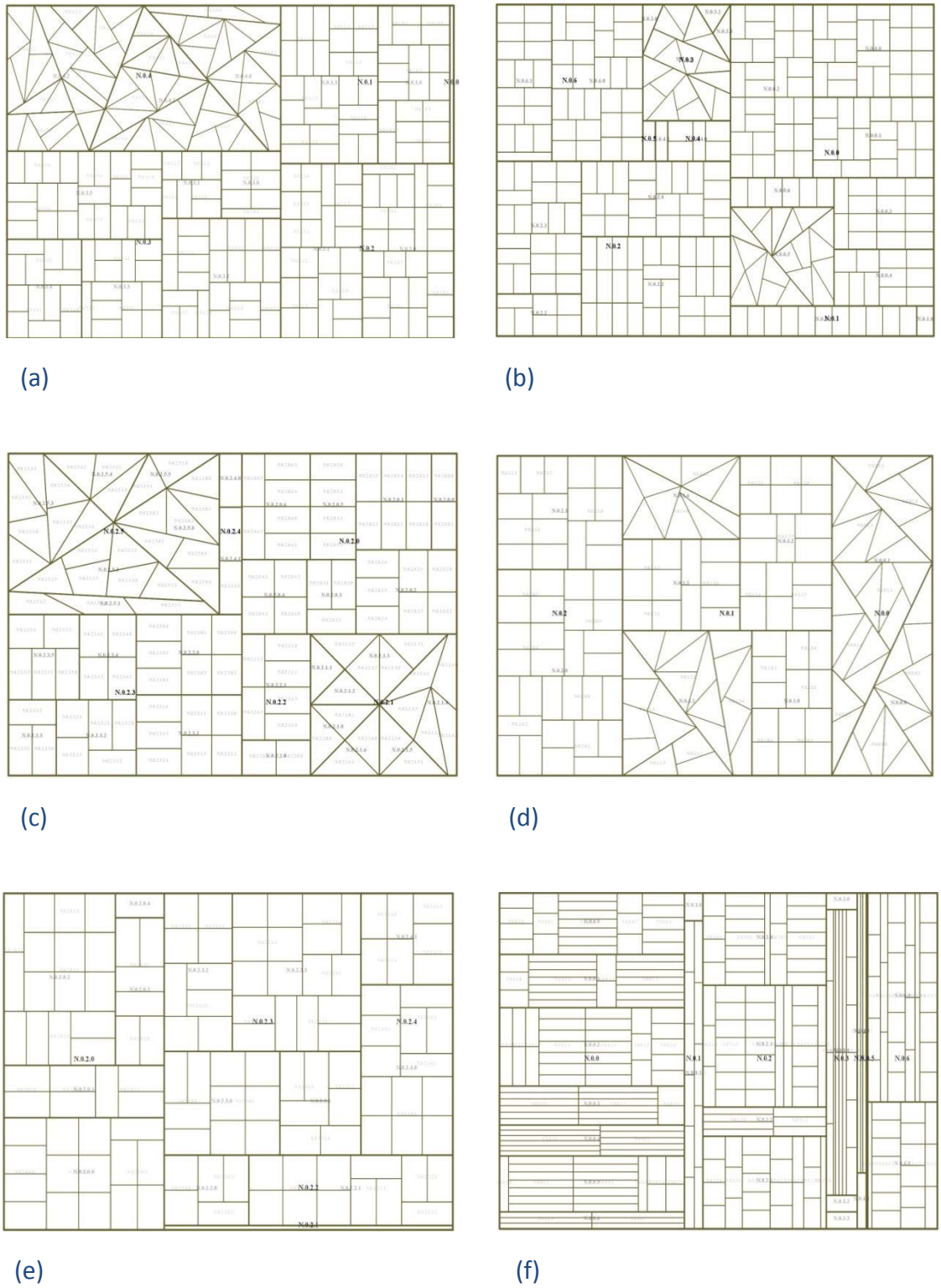
[H3] ***Tangram has a positive impact on user satisfaction of using visualization approach.***

We assume that users would prefer to use Tangram Treemaps rather than rectangular treemaps in the tasks of representing objects. We also assume in certain scenarios, a variety of representation shapes of containers different than rectangle may motivate users to engage with the use of visualization approaches in data analysis and other business processes.

6.2.3 Experiment and Design

6.2.3.1 The First User Experiment - The effectiveness of shape selection

The first User experiment is designed for users' performance in visual exploration tasks. Users were asked to perform four modified *visual navigation (or browsing) tasks*, including Task 1) browsing through the visualization to identify one target object with label; Task 2) browsing through the visualization to locate two target objects in the same hierarchy; Task 3) identifying two target objects in different hierarchical levels; and Task 4) identifying three target objects in different hierarchical levels.



(a) (b)

(c) (d)

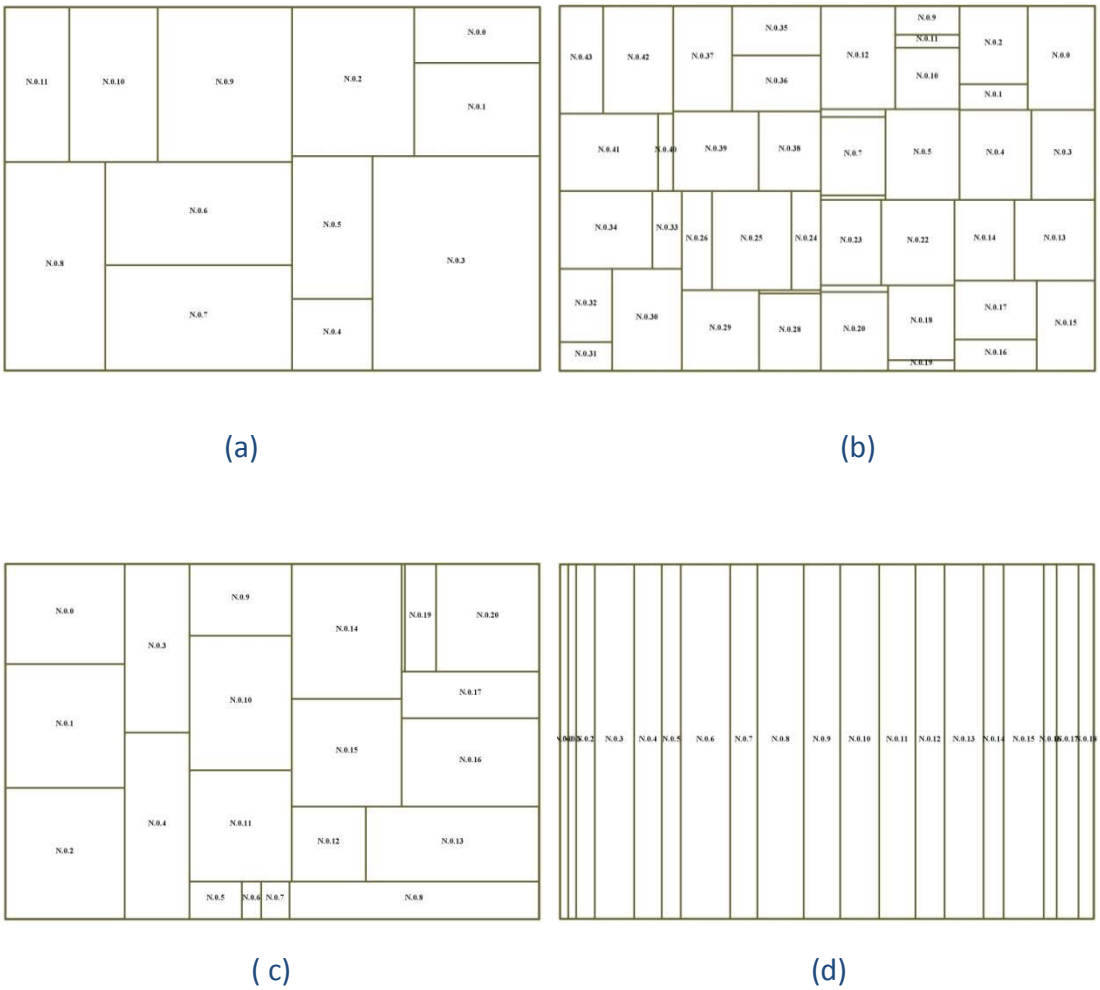
(e) (f)

Figure 6-3 Illustrations of the first user study experiment: (a) The image of Tangram Treemaps used in task 1; (b) The image of Tangram Treemaps used in task 2; (c) The image of Tangram Treemaps used in task 3; (d) The example image of Tangram Treemaps used in task 4; (e) The example image of Squarifed Treemap used in task 1, 2 ,3; (f) The example image of Slice and Dice Treemaps used in task 1

6.2.3.2 The Second User Experiment – Area size comparison

The second Experiment is to assess the human capability of distinguishing different sizes of visual objects (nodes) appearing in the same display through the comparison by human's eye-brain system. First, users were asked to identify the largest node. Then users were asked to identify and compare the two largest nodes. In the experiment, we used D&C Rectangular algorithm in a rectangular container.

Each task in the experiments 1 and 2 was not limited to a single scenario. We presented users with 7 different case scenarios for each task of each experiment and for each technique. These images included five artificial datasets and two real data sets from file system. For experiments 1 and 2, we used 5 groups of datasets and each group contains 7 datasets. We carefully measured the amount of nodes of each dataset. We made sure each scenario with the same range of nodes. In the experiment 1, sizes of datasets are evenly ranged from 10 to 1000 nodes. In the experiment 2, we found users were overwhelmed by the complexity of the datasets whose sizes were over 50 nodes. Therefore, the datasets were chosen in two ranges: 1) below 20 nodes and 2) between 20 to 50 nodes. In total, we used 36 datasets and 89 scenarios. Also, to avoid learning effects due to the within-subjects design, we varied the location of targets (i.e. nodes) in the three evaluated techniques while keeping the distances between targets approximately constant.



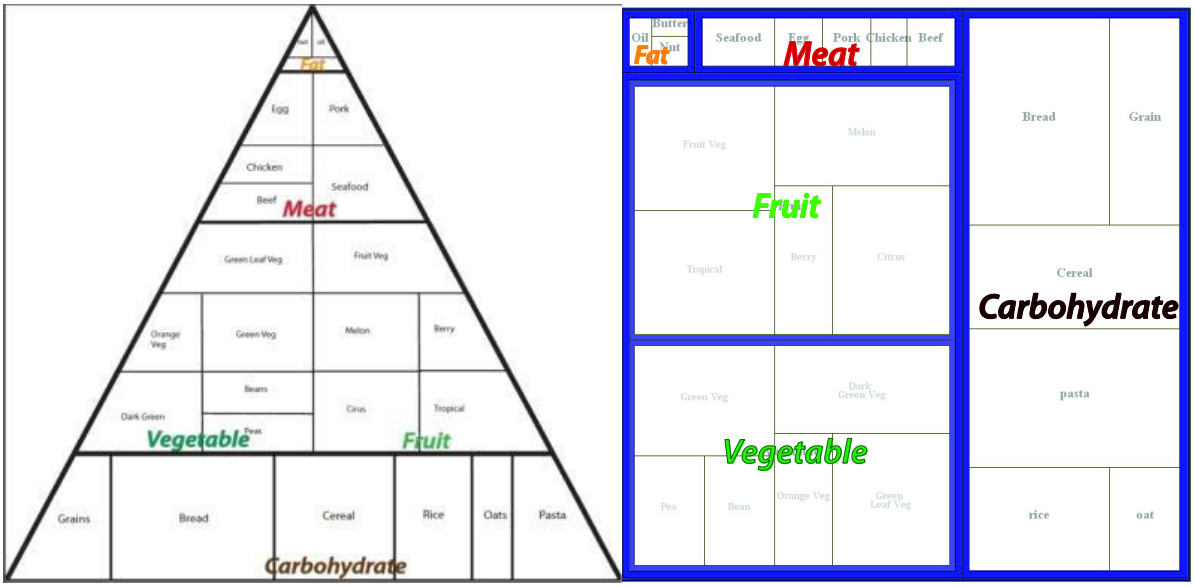
(a) (b)

(c) (d)

Figure 6-4 Illustrations of the second user study experiment for size distinguishing: (a) The example image of Tangram Treemaps with nodes below 20; (b) The example image of Tangram Treemaps with nodes between 20 to 50; (c) The image of Squarified Treemap with nodes below 20; (d) The example image of Slice and Dice Treemap with nodes below 20.

6.2.3.3 The Third User experiment - The effect of shapes I human's interpretation of data

The third user experiment tried to prove that the constrained shapes of containers do affect users' visual interpretation and exploration of data. We designed a simple dataset of a personalized food structure that are presented in treemaps with two container shapes, *triangle* and *rectangle* (see Figure 6-5). This food structure contains 25 dishes (presented as a set of nodes) presented in a three-level hierarchy. We then asked users to find the first preference in fundamental level of food structure and the first and second preference in given levels. The questionnaire in experiment 3 combines two categories of tasks, including object searching and area size judgement.



(a) (b)

Figure 6-5 Illustration of experiments in the third user study: A personalized food structure presented in both triangular (a) and rectangular treemaps (b) experimented in experiment 3.

6.2.4 PROCEDURES AND APPARATUS

Prior to the experiment, we introduced hierarchical data structure and current literature of visualization methods we used to the subjects. Then we demonstrated subjects with the three methods *Tangram Treemaps*, *Squarified Treemaps* & *Slice and Dice Treemaps*. Subjects were asked to explore data by using each visualization methods and to find out the differences among these three methods. Then we asked them to try three trials for each different task, which was described in experiment and design section.

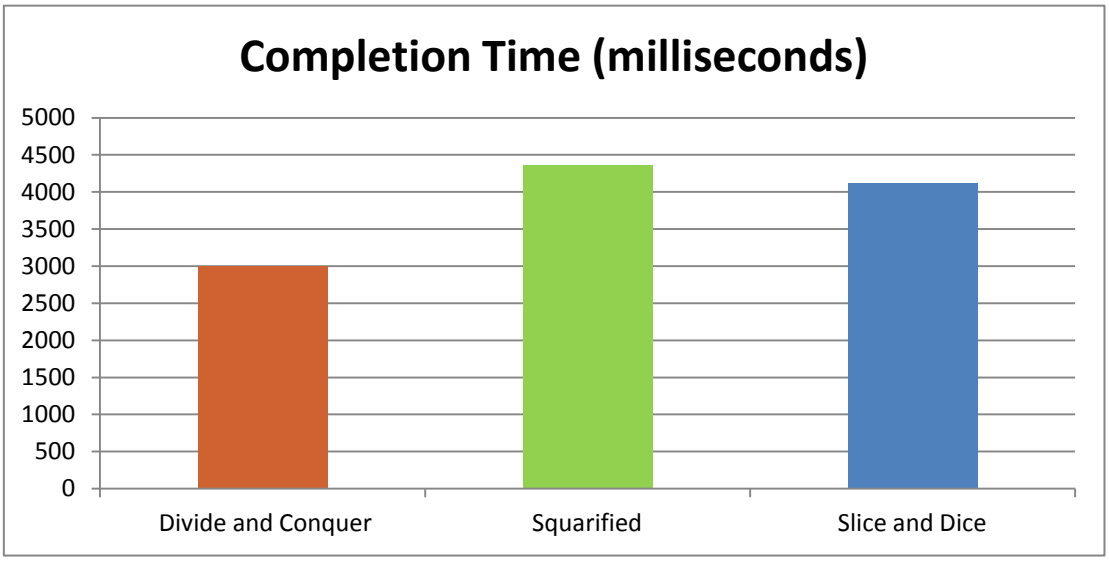
To assess whether subjects could identified all the objects in the static image, we asked them to locate and compare nodes as accurately and as quickly as possible. They were instructed that the accuracy was considered as the first priority, following by completion time. Images were presented on a 24-inches wide monitor with 1920×1200 pixels. Subjects were seated 80 cm from the monitor. The study was conducted as a within-subjects experiment with three experimental conditions (treemaps techniques) for each condition. The sequence of evaluated techniques was counter-balanced and the sequence of the questions per technique was also presented according to the increase of difficulty degree and complexity of data. For each trial, we measured task completion time and correctness. Users loaded a new image by pressing a key and were asked to write down the answers as soon as they find the answers. Between each trial there was a blank background to be displayed. Then, the user reported the where the targets were located to the experimenter, who noted it. Task completion time corresponded to the duration the image was measured. The reading time and writing time is not counted in completion time. After each condition, the users were required to assess their subjective satisfaction with the technique for each type of tasks in experiments. Upon completion of the experiment, they were asked to assess their overall preference and to participate in a semi-structured interview.

6.2.5 PERFORMANCE RESULTS

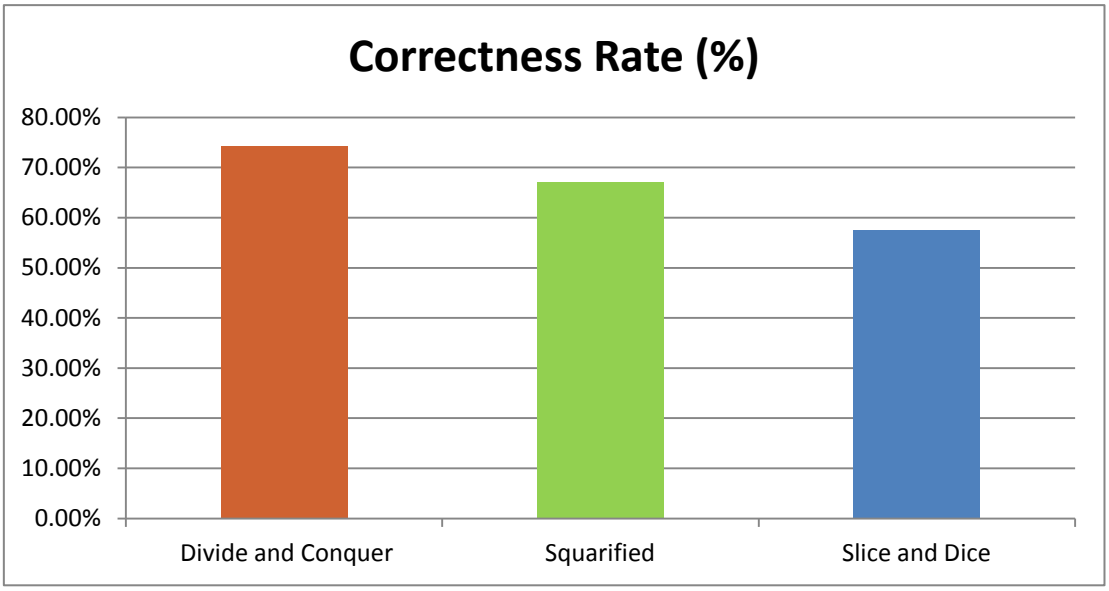
The performance measurements (i.e., task completion time and correctness ratio) were calculated by using repeated ANOVA (α = .05) with Bonferroni adjusted post-hoc comparisons. The efficiency results are illustrated in Figure 6-6.

The total completion times of experiment 1 are 2033.42, 2812.25, 2373.91 seconds, for *Tangram*, *Squarified* and *Slice and Dice Treemaps* respectively. In the experiment, Task 3 (i.e. locating two targets in different hierarchal levels), we found a significant difference in completion time $F_{2,18} = 5.5$, $p = .007$. Post-hoc comparisons revealed that completion time was significantly lower with $t_T = 3005$ ms than other techniques $t_{Squarified} = 4369$ ms and $t_{Slice\&Dice} = 4115$ ms, as illustrated in Figure 6-6. The difference of completion time between *Squarified* and *Slice and Dice* was not significant.

The total completion times of experiment 1 are 2033.42, 2812.25, 2373.91 seconds, for *Tangram*, *Squarified* and *Slice and Dice Treemaps* respectively. In the experiment 1, Task 3 (i.e. locating two targets in different hierarchal levels), we found a significant difference in completion time ($F_{2,18} = 5.5$, $p = .007$). Post-hoc comparisons revealed that completion time was significantly lower with ($t_{ANGULAR} = 3005$ ms) than other techniques ($t_{Squarified} = 4369$ ms and $t_{Slice\&Dice} = 4115$ ms), as illustrated in Figure 6-6. The difference in completion time between *Squarified Treemaps* and *Slice and Dice Treemaps* is not significant. In experiment 2, We have found a significant difference between these two methods in completion time ($F_{1,23} = 14.370$, $p < .001$). Completion time in square container was significantly higher (t_s = 4897 ms) than with in triangular container (t_t = 2148 ms).



(a)



(b)

Figure 6-6 First user study's performance results: (a) completion time of experiment 1 in Task 3; (b) correctness rate.

6.2.6 User Preference and Feedback

In the satisfaction survey of experiment 1, 18 out of 20 subjects preferred to use *Tangram Treemaps for visual navigation* and only 1 subject preferred to use *Slice & Dice Treemaps*, and 1 subject selected neutral (no preference) . In the satisfaction survey of experiment 2, 19 out of 20 subjects preferred to use triangle containers for creating treemaps. In general, it has been found that subjects had more difficulties in identifying node sizes than locating target objects with any techniques. Five users didn't like *Slice & Dice treemaps*, because the labels were hard to be recognized due to the visual clutters. They commented that the visual confusion seriously affected their decision making especially when the number of nodes increased over 50. Seven subjects mentioned that the images with enhanced *Tangram Treemaps* looked more attractive which stimulated them to actively engage with tasks.

6.2.7 Discussion of Results

Based on the results of our experiments, two of our hypotheses are fully verified, and one is partially verified. The hypotheses are described and briefly discussed as follows.

[H1] ***Tangram Treemaps have a positive impact on visual browsing (or navigation) tasks*****. Tangram Treemaps outperform other two treemap techniques, in all tasks in experiment 1, especially when the objects were in different hierarchical levels.**

[H2] Distinction of ***container shapes supports human cognitive thinking in certain scenario of shape identification and visual exploration tasks*****.** The lower completion time and higher correctness rate with triangular container in experiment 3 supports this hypothesis in this scenario. However, this only partially supports different scenarios.

[H3] ***Tangram Treemaps have a positive impact on user satisfaction of using visual approaches for data analysis and other business processes.*** *Tangram* treemaps are the overall subjects' preference for object locating tasks, because subjects found that it

encouraged the engagement with visualization approaches. In general, the combination in the visualization has successfully raised users' situation awareness during visual exploration process.

Based on formal user study which mainly emphasizes the effect in hierarchical structure exploration, we shift focus into the assessment in visual comparison in follow up user study.

SECTION 6.3 EXTENDED USER STUDY

The objectives of this follow up study were to further assess the effect of orientation of rectangular regions to the human visual perception process that will benefit the design rules of treemaps.

To achieve this, we conducted controlled experiments, investigating human perceptual ability of geometric area comparison, over the factor of true percentage, the same orientation, different orientation, orientation difference, with dependents of accuracy. These controlled experiments used the same control group and procedures and apparatus in second formal user study.

This section includes Section 5.3.1 the hypothesis prior experiments; Section 5.3.2 experiments design; Section 5.3.3 performance results and Section 5.3.4 discussion of results.

6.3.1 EXPERIMENTS HYPOTHESIS

These properties, like true percentage, the same orientation, orientation difference vary in treemaps. And our experiments were designed to investigate the effect of each property and test hypothesis as below.

[H1] ***True percentage*** impact subjects' performance in identifying the size of rectangular geometrical regions, when rectangles have the same orientation, which means rectangles rotated at the same angle. This hypothesis includes two sub- hypotheses.

[H1.1] True percentage impact subjects' responding time in identifying the size of geometrical rectangular regions, when rectangles have the same orientation

[H1.2] True percentage affects subjects' accuracy in identifying the size of rectangular geometrical regions, when rectangles have same orientation

[H2] ***True percentage*** impact subjects' performance in identifying rectangular geometric regions, when rectangles have different orientation, which means rectangles rotated at different angles. This hypothesis includes two sub-hypotheses.

[H2.1] True percentage impact subjects' responding time in identifying rectangular geometric regions*, when rectangles have different orientation*

[H2.2] True percentage impact subjects' accuracy in identifying rectangular geometric regions*, when rectangles have different orientation*

[H3] Both ***Rotation*** and ***Orientation difference*** do not suffer subjects' graphical perception in identifying rectangular geometric regions, when the aspect ratio is optimal 3/2, when true percentage remains consistently same or varies randomly.

6.3.2 Experiment and Design

In these control experiments, we used two stand-alone rectangles with the same aspect ratio which is 3/2. We controlled true percentage and orientations between rectangles. True percentage varied over 32%, 48%, 58%, and 72%. To reduce accuracy differences due to response bias, these values were explicitly placed at equal, symmetric distances from the nearest multiple of 5. We also controlled orientation angle of both rectangles varied over 0°, 15°, 30°, 45°, 60°, 75°, and 90°. We chose this measurement to follow up the prior work in graphical perception. Our experiment design thus consisted of 98 unique trials (HITs) in each experiment: 4 (True percentage difference) × 42 (orientation pairs with replication)

During the identifying of the size of geometric region, we asked subjects to compare rectangular area of varying size. The image with 600×400 pixels contained two centre-

aligned rectangles. Subjects were showed the image and instructed to identify which of the rectangles (A or B) was the smaller and then estimate the percentage the smaller was of the larger by making a quick visual adjustment.

The image used rectangles only, rather than within treemap display. Experimental design employed in our studies is based on Heer & Bostock's empirical results. Heer & Bostock found no significant accuracy difference between stand-alone rectangles and rectangles within a treemap even while varying aspect ratio. Hence our results are applicable to treemap design. (Heer & Bostock, 2010)

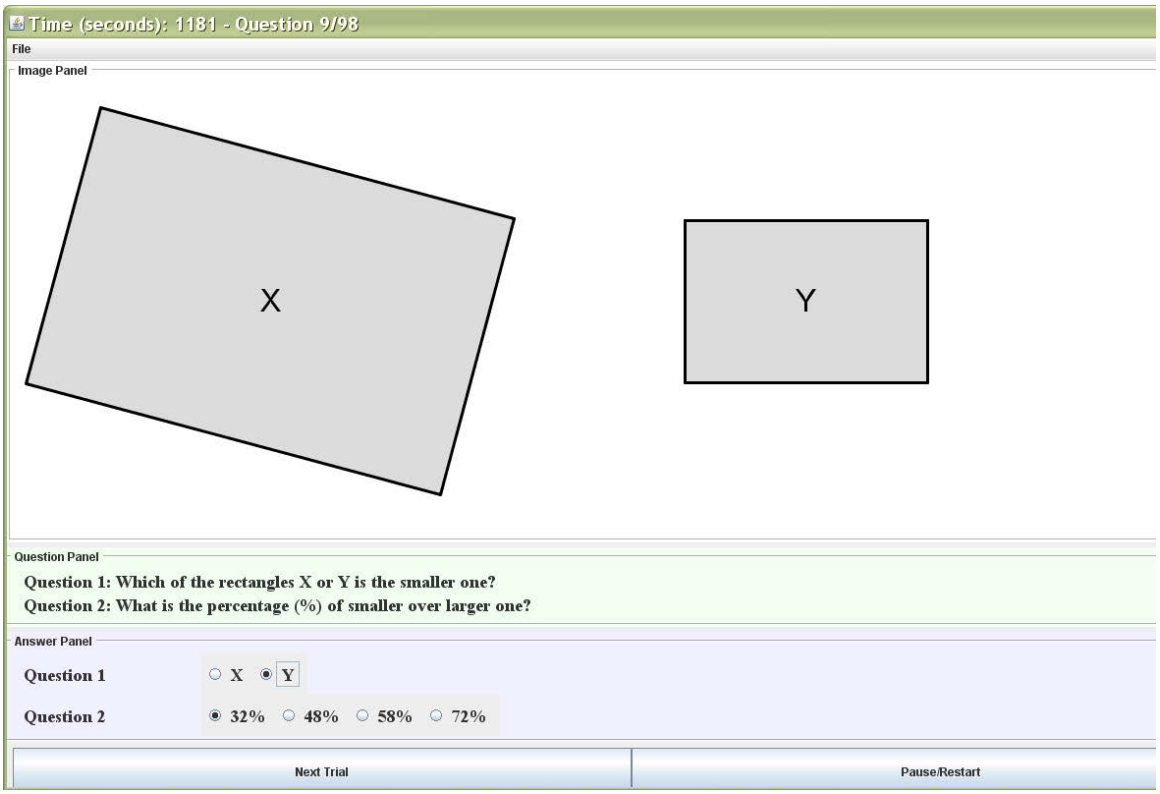


Figure 6-7 Questionnaire example in third user study: Upper section of window shows two stand-alone rectangular with different orientation; Lower section of window displays the question and answer panel.

6.3.3 PERFORMANCE RESULTS

We collected 98 × 20 = 1960 responses. Performance measurements (i.e., task completion time and correctness ratio) were evaluated using repeated ANOVA (α = .05) with Bonferroni adjusted post-hoc comparisons.

The statistical analysis was conducted on dependent of responding time on factors of true percentage, orientation pairs, same orientation, and orientation difference.

[1] True percentage: one of 32% 48%, 58%, or 72%.

[2] Orientation pairs: two of 0°,15°,30°,45°,60°,75°,90°

[3] Same orientation pairs: 0°,15°,30°,45°,60°,75°,90°

[4] Orientation difference: 0°,15°,30°,45°,60°,75°,90°

6.3.3.1 EFFICIENCY

For the effect of the same rotation angle with varying true percentage, we did not found significance in responding time (F (6,273) = 0.758, p =0.603) with the mean of 6.078 sec. For the effect of same rotation angle with the same true percentage choice over 32%, 48%, and 58%, there is no significant difference in responding time. Except when true percentage is closing to 72%, we found significance (F (6,65) = 3.381 > 2.25, p =0.603) . In the results of 72, the rotating angle with 75 degree has the lowest mean of 3.025 sec. The difference in responding time among orientation difference varying from 0°,15°,30°,45°,60°,75°,90°, was not significant as well (F(6,1952) = 0.673, p =0.671) with the mean of 5.916 sec.

For effect of true percentage with same or different orientation, we found no significance in responding time.

6.3.3.2 EFFECTIVENESS

For the effect of true percentage with the same rotation angle, we found significant difference in accuracy ($F(6, 273) = 8.516 > 2.13$, $p < 0.001$). When rectangles have varying orientations, the effect of true percentage makes more significant difference over accuracy ($F(6, 1953) = 36.819 > 2.1$, $p < 0.001$)

6.3.4 DISCUSSION OF RESULTS

Based on our experimental results, two of our hypotheses are fully verified, and one is partially verified. The hypotheses are described and briefly discussed as follows.

[H1] True percentage impact subjects' performance in identifying the size of rectangular geometric regions, when rectangles have the same orientation: True percentage affects subjects' accuracy rather than responding time in identifying rectangular geometric regions, when rectangles have the same orientation

[H2] True percentage impact subjects performance in identifying the size of rectangular geometric regions, when rectangles have different orientation*:* True percentage impact subjects' accuracy in identifying the size of rectangular geometric regions, when rectangles have different orientation. The effect of true percentage with different orientation is higher than with the same orientation.

[H3] Both rotation and orientation differences don't suffer subjects' graphical perception in identifying the size of rectangular geometric regions, when the aspect ratio is optimal 3/2, when true percentage remains consistently the same or varies randomly. This hypothesis is partially supported. For true percentage is 32%, 48%, or 58%, there is no significant difference in responding time. However, when the rectangular area size is closing to 72%, the desirable angle for orientation is 75 degree.

SECTION 6.4 SUMMARY

To summarize, we have conducted the first preliminary user study for the assessment of how well Tangram Treemaps could help users to quickly identify the "highlights", in comparison with other traditional rectangular Treemaps. The result has shown that the variation created by angular polygonal algorithm helps users to locate and identify "highlights" or focus of interests. It further proves that Tangram Treemaps are capable of laying out large hierarchical structures whose visualization can be varied to highlight important substructures.

The second user study finds out that Tangram treemaps have *a positive impact on targets locating tasks* especially when the objects were in different hierarchical levels *and also a positive impact on user satisfaction of using visualization approaches in business data processes. The results show that it* has successfully raised user's situation awareness during visual exploration process. The second user study also investigates the different containment with various shapes. The results partially support the hypothesis that the distinction of *container shapes assists human cognitive process in perception of graphic diagrams and pictures*.

The extended user study is designed to assess the effect of orientation in identifying the size of rectangular geometric region in the interpretation of pictures. We conducted controlled experiments, investigating human perceptual ability in geometric size comparison, over the factor of true percentage, the same orientation, different orientation, orientation difference, with dependents of accuracy. The result shows firstly, true percentage affects subjects' accuracy rather than responding time in identifying the size of rectangular geometric regions, when rectangles have the same orientation and secondly, true percentage impact subjects' accuracy in identifying the size of rectangular geometric regions, when rectangles have different orientation. The effect of true percentage is higher with different orientation than with the same orientation. The empirical results contribute the effect of orientation doesn't impact on visual region size judgment. This contribution also valid and re-emphasize the effect of the method of Tangram Treemaps in visual comparison.

CHAPTER 7. CASE STUDY

Initially driving by the purpose of maximization of space utilization, Tangram Treemaps further support interactive visualization tool. As a more generalized approach, Tangram treemaps provide view-based highlighting and help user track navigation process and mental map. Furthermore, Tangram treemaps can potentially support visual communication and cognitive thinking involved in decision making.

In Chapter 7, we apply the concept of Tangram Treemaps to real cases. Section 7.1 demonstrates the capabilities of Tangram Treemaps in interactive visualization tool. Section 7.2 delivers two examples in file system overview and focus viewing. Section 7.3 develops the interaction control into a file directory exploration with highlighting function. Section 7.4 discovers a potential example of Tangram Treemaps to uncover knowledge within interactive visualization.

SECTION 7.1 APPLICATION OVERVIEW

We combine Tangram Treemaps visualization technique and interaction functions for hierarchical visualization tool. To demonstrate in real cases, we implement into a number of real file directory system. Figure 7-1 shows an overview of the file system in application.

In the application, visual properties have been enhanced in hierarchical structure. In order to enhance the visibility and clarity for individual node and the hierarchical structure, we include an interactive option in the visual properties including: Section 7.1.1 use gaps for the boundaries of the child nodes from their parent's boundary, Section 7.1.2 use saturation colours for boundary and edge's thickness, and Section 7.1.3 use colours for indicating the

types of leaf nodes. The graphical properties can be turned on or off and adjusted via an interactive menu to suit user's preference of viewing.

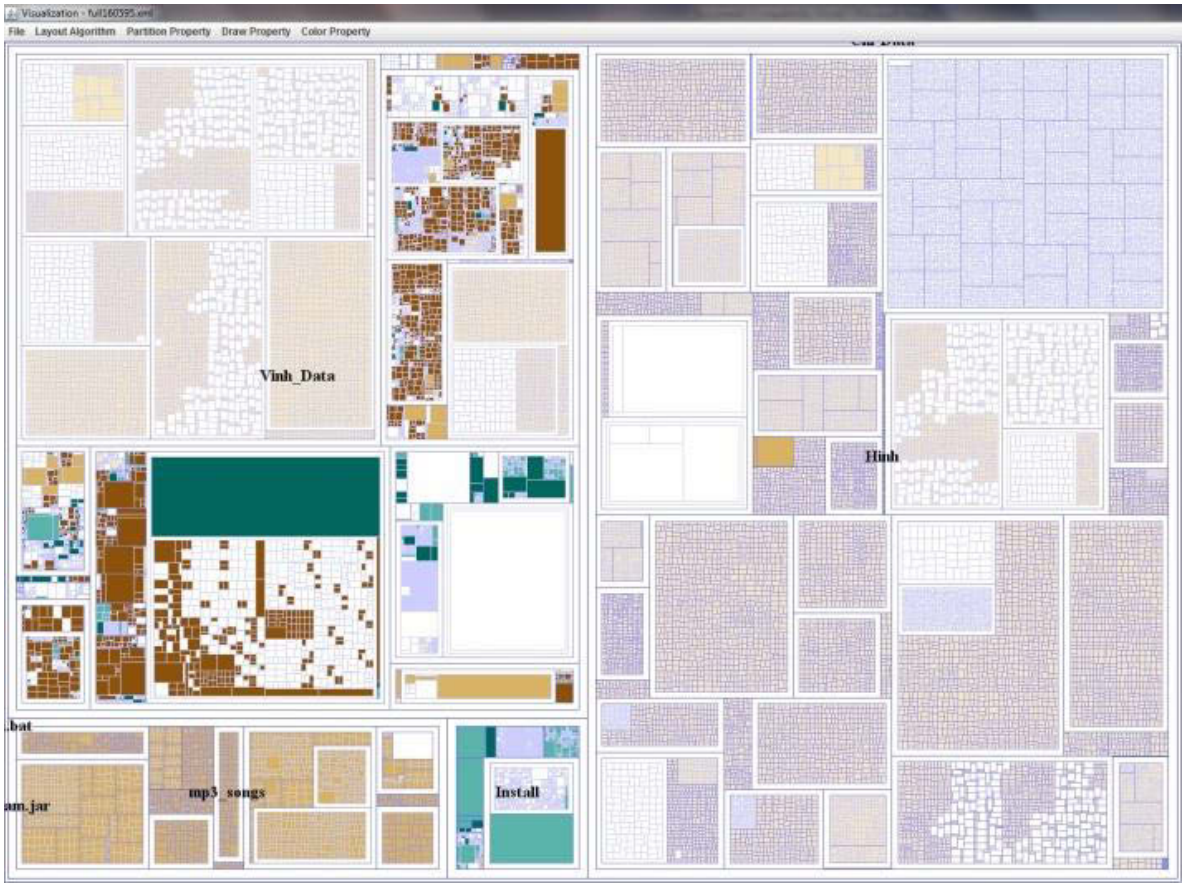
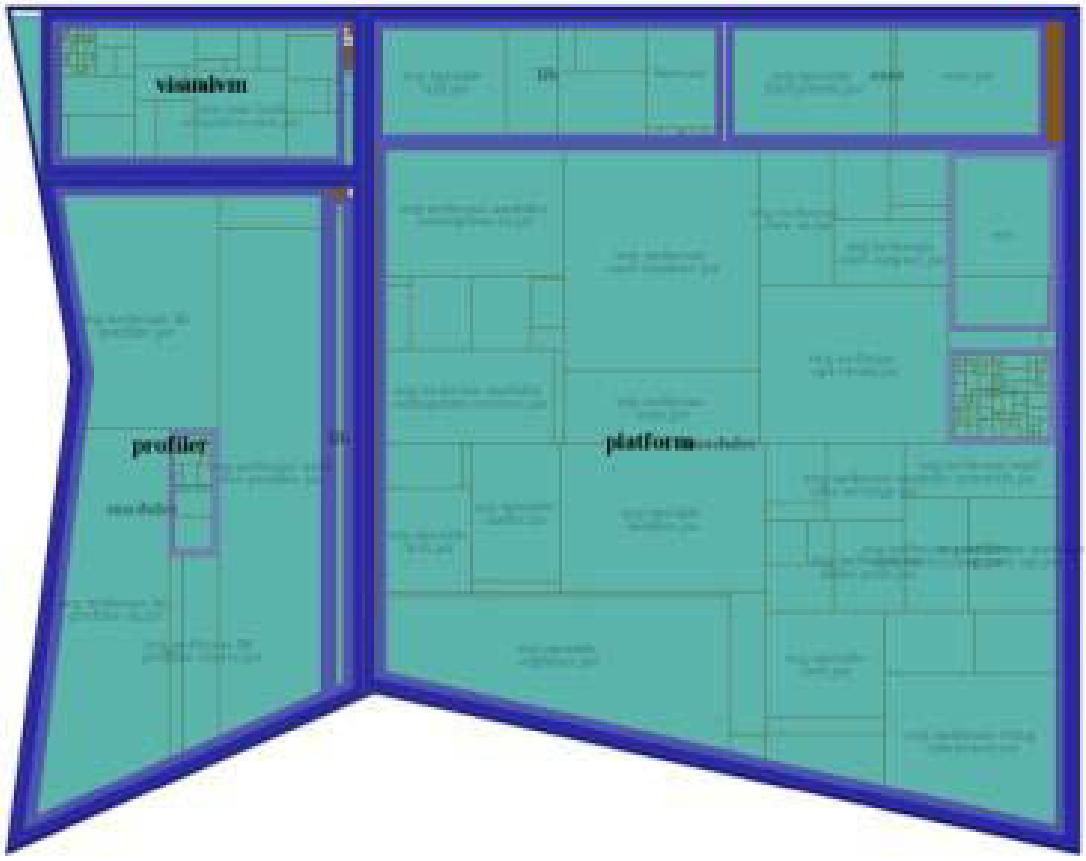


Figure 7-1 Application demonstration of file systems overview with boundary gaps feature

7.1.1 BOUNDARY GAP

Polygon offsetting approach is adopted in the application to generate uniformly gaps between boundaries of the polygons which represent either folders or files. We use the polygon offsetting method to calculate the inward offset polygon of the children's boundaries in relation to their parent's. Although boundaries gap sacrifices the space utilization, it would assist viewing the structure of hierarchies in most cases. Figure 7-2 demonstrates a file system with 478 files and folders using D&C Rectangular Algorithm. Figure 7-2 (a) shows original application view without boundary gap and Figure 7-2 (b) displays a visual enhanced application view with boundary gap on.

(a)



(b)

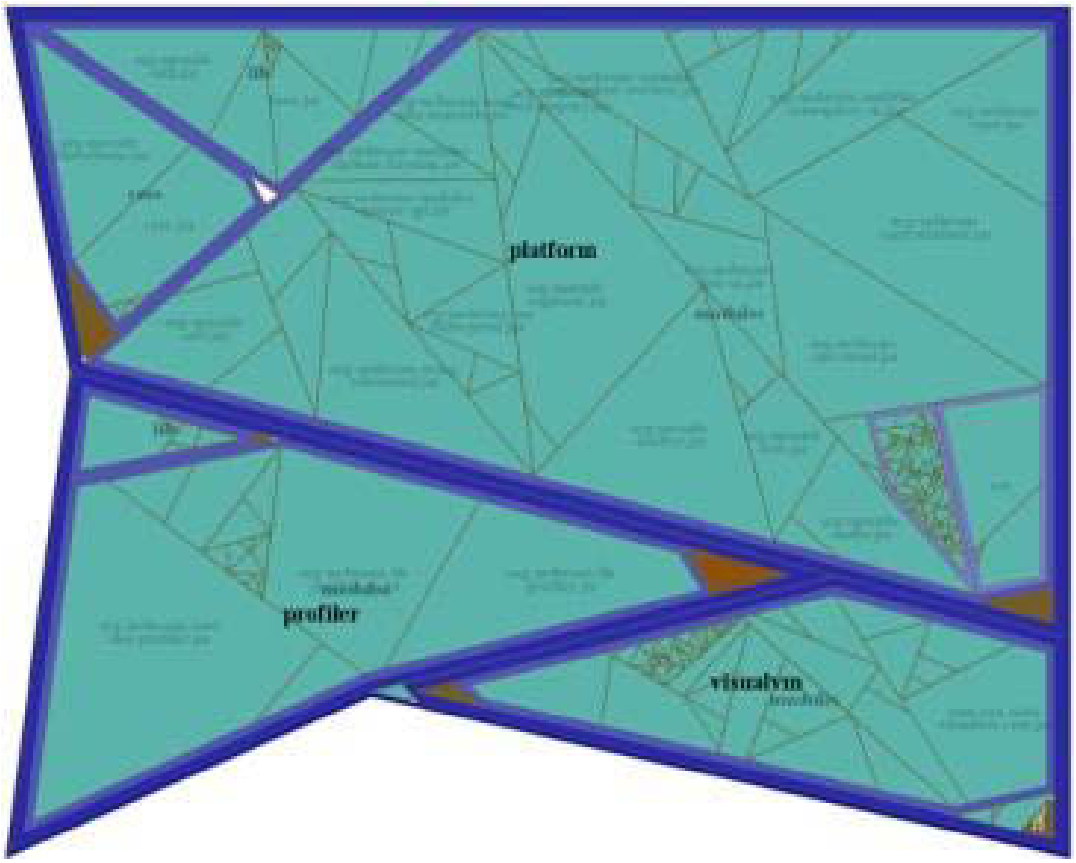
Figure 7-2 Overview illustration with boundary gap: a file system with 478 files and folders using D&C Rectangular Algorithm with (a) boundary gap feature off, (b) boundary gap feature on.

7.1.2 COLOUR AND EDGE'S THICKNESS

Colour saturation and edge thickness changes are used to assist views transferring from one level to another level. From overview to subfolder in lower lever, edges of lower level nodes are thinner and the colour of lower level nodes is lighter than those in higher level nodes. See Figure 7-3.

7.1.3 TYPES OF LEAF NODES

Using colour to visualize one more attribute of each folder or file. For example, in the file system visualization, different colours are brushed to distinguish the types of files, including documents, multimedia, images, programming code, system files, compressed files and unknown types. This option is useful to represent the embedded property of the data set. Each colour is represented a node type. The set of colours can be adjusted according to each application via a property file (Details in Section 7.2).



(a)

(b)

Figure 7-3 Overview illustration with colour visual feature: The visualization of a file system with 478 files and folders using (a) divide polygon at vertex and with angular resolution constraint (D&C Triangular Algorithm), b) divide polygon on a side with 30° (Angular Polygonal Algorithm). In the figure, cyan colour represents "jar" files and brown colour represents "xml" files.

SECTION 7.2 CASE ONE –OVERVIEW AND FOCUS VIEW

For the first case of application overview and focus view, initially, we provide an overall view of the entire file/directory hierarchy on a given shape. The context view provides overall understanding of the folder-folder and folder-file relationships. Users can also navigate to any particular sub folders for detail analysis. To enrich the visual aspects, the visualization inherits all visual properties described in Section 3.3.3 that are being explained further. Weights of vertices representing file systems are calculated based on the actual sizes of the corresponding files and folders. Colours represent different group of file types and they are carefully chosen for colour-blind safe. In this case study, we identify seven groups of file types, including:

Document group (office files and other documents): brown color

Multimedia group (multimedia files): light brown color

Image group (image files): very light brown color

Programming code group (source code): light cyan color

Executable group (executable and system files): cyan color

Compressed group (all compressed file types): dark cyan color

Other (any unspecified file types): white color

The first example is shown in a trapezoid shape. Figure 7-4 is a visualization of very large file systems inside “Program Files” directory, with approximately 145,000 files and folders. The visualization illustrates clearly that this folder contains many programs whose contents are

mainly executable and system files. There are also a large number of multimedia files in the structure. When user moves to subfolder of the program file directory, Figure 7-5 shows a detail look of a sub-folder that indicates clearly the file size and type property.

The second example is displayed in a circle shape, which likes a disc. Figure 7-6 shows an overview of the file system for Microsoft Office 2010 with approximately 5,400 files and folders. The property of the file system has been illustrated quite clearly in this visualization. There are four main fodders including “Office 14” (largest folder that occupy over 80% of the total space), “Template” (smallest size at the right side), “CLIPART” (second largest at the bottom right side) and “Document Themes 14” (at the bottom left side). Examining at each folder, we can unveil further the file properties such as 1) folder “Office 14” contains mostly executable and system files (cyan colour), there are also a large number of unspecified file types (white colour) and a few files in document and multimedia groups (brown and light brown colours); 2) folder “CLIPART” includes a majority of image files (very light brown colour); and 3) other two folders includes unspecified file types.

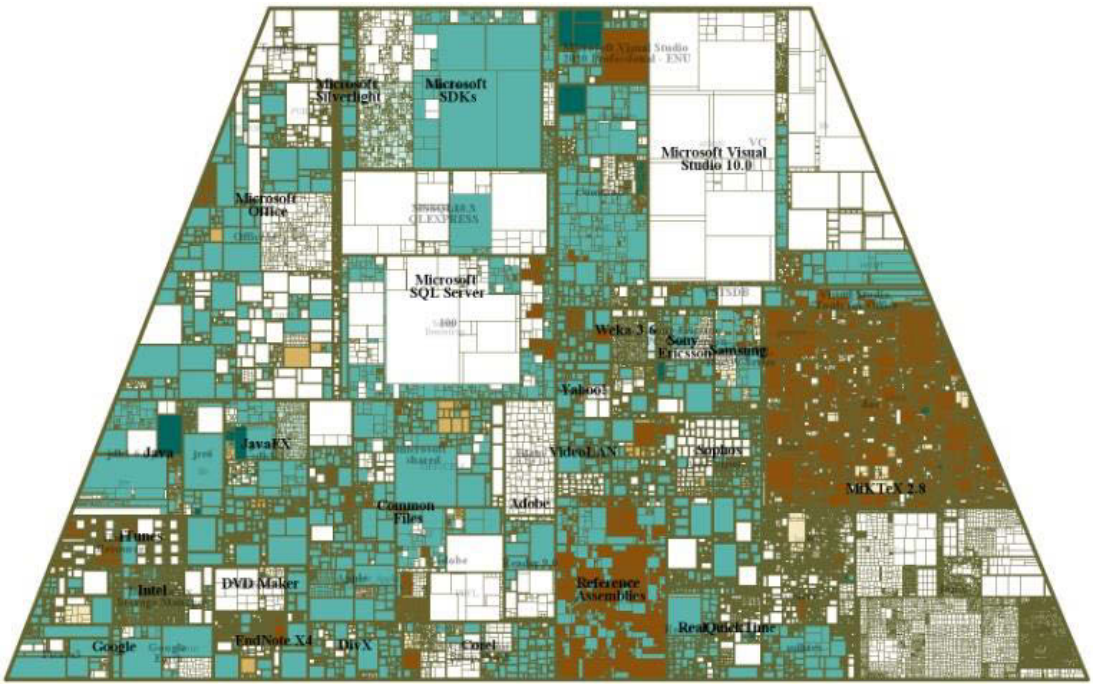


Figure 7-4 Case study 1-a: An example of the visualization of an entire large file system with approximately 145,000 files and folders in a trapezoid container.

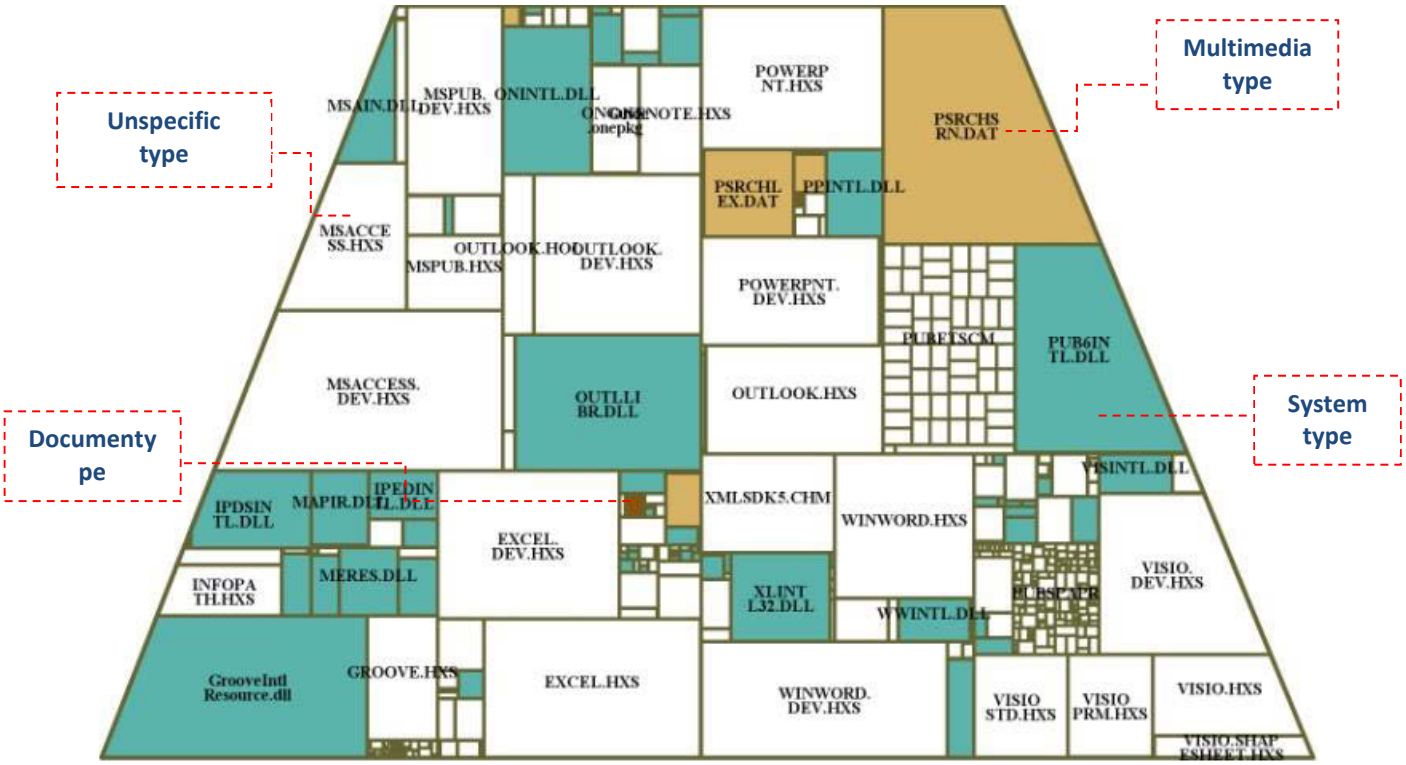


Figure 7-5 Case study 1-b: An example of the focus view in Figure 7-4.

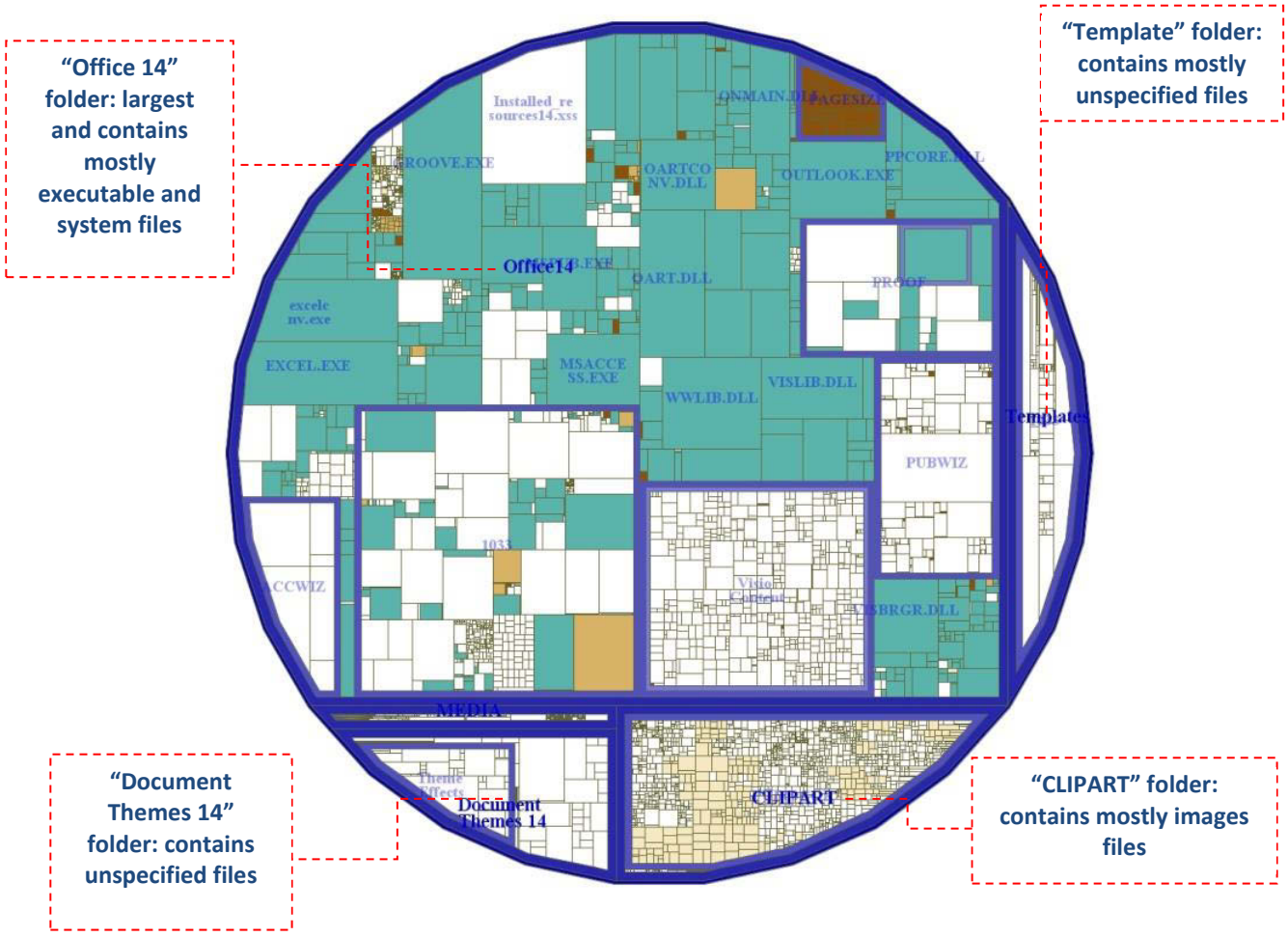


Figure 7-6 Case study 1-c: Overview of the file system for Microsoft Office 2010 with approximately 5,400 files and folders.

SECTION 7.3 CASE TWO –HIGHLIGHTING INTERACTIONS

As mentioned in Chapter 4, Tangram Application can distinguish the focus of visualization with differentiation of shape. The example shows here is a visualization of a file directory on a hexagon. In this case, the overview is generated by D&C Rectangular algorithm and the sub layouts of “My Pictures” and “Local Setting” are created by D&C Triangular algorithm. The different shapes in this visualization make the two folders standing out from the whole structure. The program can interactively change to their original vertical-horizontal layout when required. A closer examination indicates that the folder “My Pictures” contains not only images files (very light brown colour), but also multimedia files (light brown colour). To reduce this confusion of mixing different types at the same folder, it would be more effective if the multimedia files are moved into a new folder, called “My Movies”. The “Local Settings” also contains several large unspecified files in addition to the document and image files. It would be suggested to have an investigation to remove redundant or old files if necessary.

Figure 7-7 Case study 2 : Highlighting different sub layout to emphasize the interesting sections.

SECTION 7.4 CASE THREE – RECOMMENDATION OPTION

Combining Data mining algorithm, the application can provide intelligence tool which offers visual suggestions and provides recommendations for analysts to consider. System implemented with data mining query implemented can automatically generate layout based on query to draw users' attention and recommend the focus of interests. This function could potential helps user to discover unexpected.

In most situations, overloaded information hidden in complex data set visualized by traditional treemaps could exceed human cognitive ability to process. Showing emphasized sub-structures with different shapes would improve the ability of detecting the hidden items. The case study shows here utilizes angular polygonal algorithm and recommendation highlighting. For example, in

Figure 7-8, application automatically highlights two folders based on user's requirement. These two rotated folders in the visualization attract our attention. Then, we realize that two folders have similar layouts then. Shape variation has prevented human from neglecting similarity but contained in two rectangular with different aspect ratios. We then opened the dataset following the directory indicated by the emphasis. We found that they are almost duplicated, based on system recommendation.

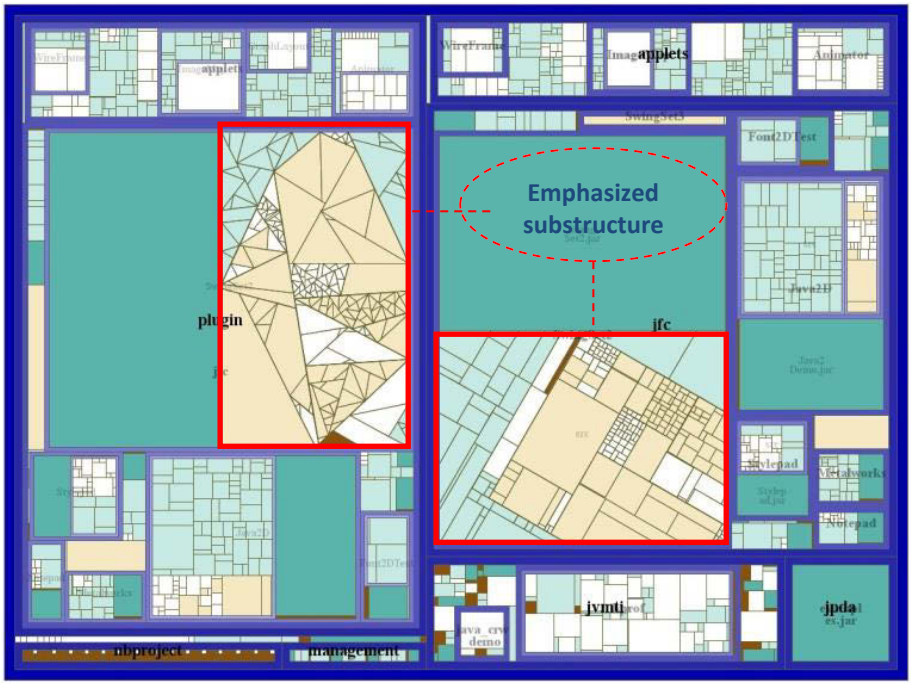


Figure 7-8 Case study 3: Recommendation option function to emphasize certain information.

SECTION 7.5 CONCLUSION

The main contribution of screen space utilization has been explored in Chapter 3. This Chapter mainly demonstrates Tangram Treemaps with practical examples. As the primary application area for treemaps is in hierarchical structured data, we apply Tangram treemaps with interaction control to file system exploration and discovery. We explore application overview and focus viewing in first case and highlighting interaction in second case and recommendation option in third case. These three case studies prove that Tangram Treemaps technique can be applied to a wide range of visual analysis scenarios. However, to achieve success in visual analytics environment, these case studies also imply that we should explore and extend treemaps further to be one of the important components of visual analytics.

CHAPTER 8. CONCLUSION AND FUTURE WORK

The preceding chapters have presented, discussed and illustrated all the relevant backgrounds and technical components of my PhD research. This chapter concludes the thesis by providing a summary of key contributions made by this thesis to the discipline of Data visualisation. Finally, this chapter outlines the directions for the future work.

SECTION 8.1 REFLECTIONS ON THESIS QUESTIONS

In almost every field of our lives, making timely, precise and correct decision for every event is crucial. However, it is getting increasingly difficult because we are moving to the age of "Big Data", while people are receiving a tremendous large volume of on-line data daily that forms a huge information source for people to analyse and make right decisions for solving their problems.

However, the analysis and understanding of such large, complex and dynamic datasets have exceeded the capacity of using the traditional analytic reasoning approaches. To make a profound difference from the traditional approaches of data analysis and presentation, successful visualization could enable humans to synthesize and simplify information, derive insights from massive data, and provide timely assessments. The advanced visualization techniques can enhance humans' capacity of perceiving, understanding and analytical reasoning of data.

As the typical hierarchical data visualization, Space-filling (or Treemaps) provides the capability of interpreting attributed hierarchical data structures in an enclosure visual form. It is to geometrically position the entire tree structure inside a limited display space. So far, Treemaps have been successfully applied to data sets from various domains, including financial data and various others.

However, the traditional Treemap approaches have three major challenges as mentioned in the introduction. Firstly, in real world, analysts often monitor multiple views and real time processes at the same time, due to the time demands or task requirements. More often the visualization tool shares the screen space with other concurrent projects or process sessions. To avoid overlapping or wasting the screen space, visualization researchers have to consider the screen space utilization. Space-filling (or Treemaps) approaches do very well in maximization of the display space utilisation of its own session (with rectangular container). However, they do not consider the display utilization of the whole computer screen, where a number of concurrent sessions are running simultaneously.

Secondly, most of the existing *Treemap* algorithms have a common limitation: they can only partition a rectangle into sub-rectangles. This limits both human perception on visual object recognition and Treemaps' adaptability to various display areas other than rectangles, such as non-uniform polygons which are more closing to physical or natural shapes, for example, like fluctuation, evolution, landscape, biological object and etc. Hence, limiting treemap visualization to rectangles blocks the utilization of human capability on object recognition, due to the same fixed size (90 degrees) of all the angles of the shapes in the tree visualization.

Thirdly, while conquering the first and second challenges, the complexity of algorithms should also be appropriate to be applied to real time applications. With the new polygon based enclosure, *Space-optimized tree* provides the possibility to be applied for convex polygonal shapes. *Voronoi Treemaps* attempt to break this restriction as well. However, the high computation time complexity increases the difficulties in practice for large hierarchal data sets without powerful computers, and thus decreases the potentiality to be applied to real time applications.

SECTION 8.2 ANSWERS TO THESIS QUESTIONS

Motivated by these three challenges, we propose the concept of Tangram Treemaps, inspired by the idea of Tangram Puzzle Game. We successfully create the tessellation method for Tangram Treemaps with implementation algorithms. In result, Tangram Treemaps manage effectively for screen space optimization on one hand. Tangram Treemaps are able to visualize large relational structures in virtually any chosen shapes. On the other hand, Tangram Treemaps offer flexibility for container and containment control to produce various layouts for visualization platforms. The flexibility of the algorithm makes it possible to create different types of layouts with a small adjustment in the algorithm, such as Triangular, Angular Polygonal and vertical-horizontal rectangle.

In terms of methodology, the new technique maximizes the space efficiency and achieves the optimization criteria required by the enclosure approach. The efficiency of the proposed method is due to the combination of the simple and effective *Divide and Conquer* approach with the *Treemap*'s paradigm. The optional inclusion of the angular constrains refinement offers the effective visualization with satisfactory angular resolution. Taking its real-time performance and the quality of the visualization layouts into account, *Tangram Treemaps* are very effective and flexible interactive tools for producing visual representation of large data structures. Therefore, Tangram Treemaps originally contribute in three aspects as below.

8.2.1 CONTRIBUTION 1 –SCREEN SPACE OPTIMIZATION

Tangram Treemaps have achieved screen space optimization when multiple-session display is required in one computer screen.

The new approach manages to quickly generate layouts to fill the display space without overlapping, which meets the requirement of the enclosure tree visualization method. Furthermore, Tangram Treemaps offer the freedom of development in different geometrical shapes of containers other than regular rectangles. This new approach can be applied into different enclosed shapes when multiple screen sessions are occupying the rest of the screen space.

Tangram Treemaps adopt the tessellation process which can produce nested polygons whose sizes are proportional to the weight of vertices. However, it does not restrict partition direction in vertical and horizontal directions. The algorithms effectively relax rectangular constraints. With the modification of the algorithm, the new approach is capable to produce multiple layouts for various polygonal shapes.

For contribution 1, Chapter 2 introduced the concept and idea evolution of Tangram Treemaps. To achieve screen space optimization, Chapter 3 described the methodology and discussed technical specifications of the implementation algorithms. Section 3.3 explored the experimental results in various enclosed shapes. Section 3.3.4 demonstrated Tangram Treemaps' ability of container control.

Section 3.2.2.3 investigated the angular resolution improvement with evaluation based on the experimental results. Chapter 5 evaluated the approach based on proximity of nodes, one of the graphical drawing aesthetics. Chapter 6 clarified the needs of different container and implied changing container of display may also support cognitive thinking in certain situation.

8.2.2 CONTRIBUTION 2 – VISUALIZATION LAYOUT FLEXIBILITY

Tangram Treemaps have the ability to effectively represent the hierarchical information with flexible visualization layouts.

The proposed new approach offers the flexibility to generate various layouts in visualization, including triangular, angular rectangular, polygonal layouts and vertical and horizontal rectangular layouts. The algorithms which generate these layouts facilitate construction of visualization which combines with each other or with other enclosure approaches. Therefore, Tangram Treemaps offer the ability to combine different types of layouts to emphasize the hierarchical structure and focus.

The ability of embedding different types of shapes in the visualization can improve the human perception because humans can very quickly detect one shape when it is different from the others (Ware, 2004).

To effectively represent the hierarchical datasets, the new algorithm leverages the treemaps paradigm, via adopting containment which enclosures child nodes in parent nodes and encode values using area. With the flexibility of layouts, users have three ways to control visualization outputs: changing the shape of the container, changing the shape of the child, or changing both of them. This function advances Tangram Treemaps towards the end-user centered design in the future.

For contribution 2, 0 answered how to generate various layouts, by proposing three algorithms, D&C Triangular Algorithm, Angular Polygonal Algorithm, and D&C rectangular Algorithm. Section 3.2 presented these methods in illustration and experimental results. Section 3.3 reviewed visualization layouts with the flexibility of container, containment and both. Chapter 4 demonstrated how to utilize the ability of flexibility layouts in interaction control. Chapter 5 evaluated visualization methods based on graphical optimization criteria. Chapter 6 conducted a formal user study in typical tasks of visual data analysis and assessed the performance of Tangram Treemaps in comparison with traditional treemaps. It justified

the advantage of visualization layout flexibility. The outcomes of the user study supported the claims in the introduction. The user study supported that *Tangram Treemaps* have met users' satisfaction, and it was further found that the combination of multiple algorithms in the visualization had successfully improved users' situation awareness during visual searching process. Finally, Chapter 7 demonstrated the use of layout flexibility in the case study.

8.2.3 Contribution 3 - Low computational complexity

While we proposed Tangram Treemaps to contribute to both the screen space optimization and the visualization layout flexibility, we also aimed to develop the approach with low computational complexity, because time complexity is a very important attribute that decides the environment in which the technique can be applied. In literature, most existing approaches can successfully meet graphical drawing criteria with high computational complexity cost. Because of the high complexity of their algorithms, these approaches failed to maintain the visualization of dynamic data stably. Nevertheless, Tangram treemaps are capable to effectively maintain the quality of layout with the optimal aspect ratio and the orientation of the visualization with lower computational complexity.

Tangram treemaps simplify the tessellation process by utilizing the divide and conquer methodology. We also carefully add constraints into the algorithms in order not to increase the time complexity.

For contribution3, Section 5.1.1 analysed the computational complexity in terms of time operations, and Section 5.1.2 evaluated the computational complexity based on the running time from experiments. Section 5.1.3 compared the running time with other traditional treemap techniques. The running time of the proposed algorithms is still comparable with other standard treemaps even after they are added with angular resolution constraints, which improves the readability. In brief, the efficiency of the algorithm and its competitive time complexity makes Tangram possible for real time applications.

SECTION 8.3 FUTURE WORK

For future work, we focus on four aspects: technical improvements in Section 8.3.1, alignment with industry in Section 8.3.2, treemap design guidelines in Section 8.3.3 and systematic treemap evaluation principles in Section 8.3.4.

8.3.1 TECHNICAL IMPROVEMENTS

The work presented in this thesis is still in the early stage for a mature visualization tool. There are still a number of technical imperfections which need to be improved in future. For example, overlapping labelling causes visual clusters when the dataset increases dramatically.

In addition, the work we provide very adopts basic navigation scheme. Drilling-down and semantic zooming only provides a single focus view or context view. Although we proposed a novel method which takes advantage of the differentiation of shapes, the context viewing should be considered in the future. The research proves that rich context information produced in the exploration path will greatly increase the accuracy of user decisions and reduce the “unsuccessful trips” and “unnecessary views” during visual exploration of large hierarchies. Generally, analysts are found to look over the full structure of a semantic graph and mentally partition the graph into natural clusters of high activity or dense sub-graphs (Wong, China, Foote, Mackey& Thomas, 2006). As in the viewing process, after one target dataset has been characterized, the analyst may have to return to a larger view of picture to examine the relationships between the target and whole organization. Hence, we should build a navigation mechanism with sufficient context information to guide their navigation through deep and complex relational structures. The solution should allow users to trace each step of their interaction and make it easy to jump or return to any level of the hierarchy that they have previously visited.

8.3.2 ALIGNMENT WITH INDUSTRY

As researchers in visualization moving to visual analytics, we should bridge visualization user centred design with real data process analysis process. In reality, even with well-designed visualization and sufficient data processing speed, analysts still find it actually takes significantly long time to exploit and interpret meaningful information by using visual analytic tools. In IBM for instance, several visualization functions both in generic business intelligence reporting tools and financial analysis tools are discarded by the analysts, e.g. Hyperion (formal brio), simply because development of these tools are separated from the traditional operational workflows. Generally, analysts normally find it either hard to fit into their daily task or uncomfortable to interactive with the tools.

The research shows that in almost every domain, many visual analytics tools fail to fulfilled analysts' needs, or fully meet client's requirements. This is mainly caused by the gaps between visual analytics and financial analytical reasoning in applying visual analytics instead of technical problems. By extension, on one side, system developers do not have the ability to tailor applications to fit into financial decision making processes due to the lack of the knowledge financial analysts possess. On the other side, without formal and reliable evaluation report, business consultants, who don't have enough technology knowledge, have to choose from their preferences. In result, impropriate visually structured representation may negatively influences the ways of thinking and communication between analysts and information.

In future, we will work closely with domain experts and understand better their data analysis process and classify business requirements. So that we can investigate the analytic process and the use of visualization tools from the end user's perspective.

8.3.3 TREEMAP DESIGN GUIDELINES

Prior works on treemaps have mainly focused on developing the new layouts. The existing treemaps generated from various algorithms require careful examination on design parameters. However, current research does not provide usability studies of treemaps guidelines on effectiveness of design parameters. Hence, selecting the most effective parameters for certain type of task is primarily based on intuition preference of visualization designer. In third stage of evaluation, we amend the investigation of treemap design guidance with assessment of orientation, as the impact of orientation remains unclear in visual analysis tasks performance. However, in the existing research, there is insufficient guidance on orientation for treemap design yet. Therefore, In future, we will extend other design parameters and assess two or more factors inter-effect. The future work will continue to fulfil perceptual guidance for treemap design.

8.3.4 SYSTEMATIC TREEMAP EVALUATION PRINCIPLES

We studied the efficiency and effectiveness of algorithms and layouts with experiments. To avoid shortcomings of empirical metrics, we also conducted a preliminary and formal user study to evaluate the visualization and interaction in the application prototype. Although there are some prior works in evaluation, there is still no consensus on how to evaluate visual analytics systems as a whole and no formal evaluation guidelines. Therefore, it is highly desirable to provide infrastructure for scientific evaluation of visualization.

SECTION 8.4 FINAL CONCLUSIONS

To summarize, the paper demonstrated the ability of proposed *Tangram Treemaps*, method and implementation algorithms to visualize large relational structures in virtually any convex polygon. The efficiency of proposed method is based on the combination of the simple and effective Divide and Conquer approach with the angle. The speed and the generalisation of the partitioning makes proposed technique suitable for in-depth visual analysis of hierarchical structures encoded in large data sets. Taken in account its real-time performance and the quality of the visualization layouts, *Tangram Treemaps* can be a very effective and flexible interactive tool for producing visual representation of large data structures as well as emphasizing substructures using angular partitioning approach. Users can make own preference over layouts and change any time during the analysis process. Relaxing the rectangular limitation, Tangram Treemaps can be potentially adopted into a wider range of applications, within different boundary of polygonal shape of containers. Application designers can combine it with rectangular treemaps for diverse domains purpose. In future, to ensure information visualization as lynchpin in data analysis and decision making, we should continue to fulfil the theory development and extend practices of highlighting and advance the user-centre evaluation for visualization in each layer of services.

PUBLICATION LIST

- Liang, J., Nguyen, Q.V., Simoff, S. & Huang, M.L., 2013, 'Angular Polygonal Enclosure approach', in Banissi E.(ed.), Information Visualization- Techniques, Usability &Evaluation, Cambridge Scholar, accepted. [IVTUE13]

- Huang, M.L. &Liang, J., 2013, 'Highlighting in Visual Data Analytics', in Huang M.L. &Huang W.D (ed.) , Innovative Approaches of Data Visualization and Visual Analytics, IGI Global, accepted. [DVVA13]

- Liang, J., Huang, M.L. & Nguyen Q.V., 2012, 'Perceptual User Study For Combined Treemaps', IEEE Machine Learning and Applications, pp. 300 – 305. [ICMLA12]

- Liang, J., Hua J., Huang, M.L., Nguyen Q.V. & Simoff, S., 2012, Rectangle orientation in area judgment task for treemap design. Proceedings of the 24th Australian Computer-Human Interaction Conference, 2012. ACM, 349-352. [OzCHI12]

- Liang, J., Nguyen, Q.V., Simoff, S. & Huang, M.L., 2012, 'Angular Treemaps – A New Technique for Visualizing and Emphasizing Hierarchical Structures', IEEE Information visualization Conference, pp. 74-80. [IV12a]

- Huang, M.L., Liang F.L., Chen Y.W., Liang, J.& Nguyen, Q.V.2012, 'Clutter Reduction in Multi-Dimensional Visualization of Incomplete Data Using Sugiyama Algorithm', IEEE Information visualization Conference, pp. 93-99. [IV12b]

- Huang, M.L. & Liang, J., 2010, 'Highlighting in Information Visualization: a Survey", IEEE Information visualization Conference,pp.79-85.[IV10]

- Huang, M.L., Liang, J. & Nguyen, Q.V. 2009, 'A Visualization Approach for Frauds Detection in Financial Market'. IEEE Information visualization Conference, pp197-202.[IV09]

- Huang, M.L., Liang, J. & Nguyen, Q.V., 2008, 'A Usability Study on the use of Multi-Context Visualisation', IEEE Fifth International conference on Computer Graphics, Imaging and Visualization, pp. 311- 316. [CGIV08]

REFERENCES

Ahlers, J. & Weimer, H. 2002, 'Challenges in interactive visualization for knowledge management', *Sixth International Conference on Information Visualisation, 2002*, pp. 367-71.

Ahmed, A., Fu, X., Hong, S.-H., Nguyen, Q. & Xu, K. 2010, 'Visual Analysis of History of World Cup: Dynamic Network with Dynamic Hierarchy and Geographic Clustering', in M.L. Huang, Q.V. Nguyen & K. Zhang (eds), *Visual Information Communication*, Springer US, pp. 25-39.

Anderson, J.W. 2005, *Hyperbolic geometry*, 2nd edn, Springer London, <http://www.lib.uts.edu.au/sso/goto.php?url=http://dx.doi.org/10.1007/1-84628-220-9>.

Andrews, K. & Heidegger, H. 1998, 'Information slices: Visualising and exploring large hierarchies using cascading, semi-circular discs', *Proceedings of IEEE Symposium on Information Visualziation*, IEEE, pp. 9-12.

Archambault, D., Purchase, H. & Pinaud, B. 2010a, 'Difference map readability for dynamic graphs', *18th International Symposium on Graph Drawing*, Springer-Verlag, Konstanz, Germany, pp. 50-61.

Archambault, D., Purchase, H. & Pinaud, B. 2010b, 'The Readability of Path Preserving Clusterings of Graphs', *12th annual Eurographics/IEEE Symposium on Visualization*, vol. 29, Wiley Online Library, pp. 1173-82.

Asano, T., Ranjan, D., Roos, T., Welzl, E. & Widmayer, P. 1997, 'Space-filling curves and their use in the design of geometric data structures', *Theoretical Computer Science*, vol. 181, no. 1, pp. 3-15.

Aurenhammer, R.K., R. 1999, 'Voronoi Diagrams', *Handbook of Computational Geometry*, Elsevier Science B. V., pp. 201-90.

Baker, M.J. & Eick, S.G. 1995, 'Space-filling software visualization', *Journal of Visual Languages and Computing*, vol. 6, no. 2, pp. 119-33.

Baldonado, M.Q.W., Woodruff, A. & Kuchinsky, A. 2000, 'Guidelines for using multiple views in information visualization', *Proceedings of the working conference on Advanced visual interfaces*, ACM, Palermo, Italy, pp. 110-9.

Balzer, M. & Deussen, O. 2005, 'Voronoi treemaps', *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005*, pp. 49-56.

Balzer, M., Deussen, O. & Lewerentz, C. 2005, 'Voronoi treemaps for the visualization of software metrics', *2005 ACM Symposium on Software Visualization*, ACM, St. Louis, Missouri, pp. 165-72.

Bartram, L. & Ware, C. 2002, 'Filtering and brushing with motion', *Information Visualization*, vol. 1, no. 1, pp. 66-79.

Battista, G.D., Eades, P., Tamassia, R. & Tollis, I.G. 1994, 'Algorithms for drawing graphs: an annotated bibliography', *Computational Geometry*, vol. 4, no. 5, pp. 235-82.

Bautista, J. & Carenini, G. 2006, 'An integrated task-based framework for the design and evaluation of visualizations to support preferential choice', *Proceedings of the working conference on Advanced visual interfaces*, ACM, pp. 217-24.

Beaudoin, L., Parent, M.A. & Vroomen, L.C. 1996, 'Cheops: a compact explorer for complex hierarchies', *Visualization '96*, IEEE, pp. 87-92.

Becker, R. & Cleveland, W. 1987, 'Brushing scatterplots', *Technometrics*, vol. 29, no. 2, pp. 127-42.

Becker, R.A., Eick, S.G. & Wilks, A.R. 1995, 'Visualizing network data', *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, no. 1, pp. 16-28.

Bederson, B. 2000, 'Fisheye menus', *13th annual ACM symposium on User interface software and technology, UIST '00* ACM, pp. 217-25.

Bederson, B.B. 2001, 'Quantum treemaps and bubblemaps for a zoomable image browser', *Proc. User Interface Systems and Technology*, pp. 71-80.

Bederson, B.B., Shneiderman, B. & Wattenberg, M. 2002, 'Ordered and quantum treemaps: Making effective use of 2D space to display hierarchies', *ACM Transactions on graphics (TOG)*, vol. 21, no. 4, pp. 833-54.

Beroggi, G.E.G. 2001, 'Visual-interactive decision modeling (VIDEMO) in policy management: Bridging the gap between analytic and conceptual decision modeling', *European Journal of Operational Research*, vol. 128, no. 2, pp. 338-50.

Bially, T. 1969, 'Space-filling curves: Their generation and their application to bandwidth reduction', *IEEE Transactions on Information Theory*, vol. 15, no. 6, pp. 658-64.

Blanch, R. & Lecolinet, E. 2006, 'Navigation techniques for zoomable treemaps', *UIST 2006 Adjunct Proceedings: Demonstrations*, Montreux, Switzerland, pp. 49-50.

Bottger, J., Brandes, U., Deussen, O. & Ziezold, H. 2008, 'Map Warping for the Annotation of Metro Maps', *IEEE Pacific Visualization Symposium, 2008*, pp. 199-206.

Brath, R. & Peters, M. 2005, *Information visualization for business-past & future*, DM Review Magazine, viewed 5/5/2008, <http://www.dmreview.com/issues/20050101/1016489-1.html>.

Brath, R., Peters, M. & Senior, R. 2005, 'Visualization for communication: the importance of aesthetic sizzle', *Ninth International Conference on Information Visualisation, 2005*, pp. 724-9.

Brennan, S.E., Mueller, K., Zelinsky, G., Ramakrishnan, I., Warren, D.S. & Kaufman, A. 2006, 'Toward a Multi-Analyst, Collaborative Framework for Visual Analytics', *IEEE Symposium On Visual Analytics Science And Technology, 2006*, pp. 129-36.

The Broken Heart Tangram 2008, viewed 19 June 2010 <http://www.walkingrandomly.com/?p=65>.

Bruls, M., Huizing, K. & Van Wijk, J.J. 2000, 'Squarified treemaps', *Joint Eurographics and IEEE TCVG Symposium on Visualization*, Springer, Vienna, Austria, pp. 33-42.

Caat, M., Maurits, N.M. & Roerdink, J.B.T.M. 2007, 'Design and Evaluation of Tiled Parallel Coordinate Visualization of Multichannel EEG Data', *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 1, pp. 70-9.

Cañas, A., Carff, R., Hill, G., Carvalho, M., Arguedas, M., Eskridge, T., Lott, J. & Carvajal, R. 2005, 'Concept Maps: Integrating Knowledge and Information Visualization', in S.-O. Tergan & T. Keller (eds), *Knowledge and information visualization*, vol. 3426, Springer Berlin Heidelberg, pp. 205-19.

Card, S.K. & Mackinlay, J. 1997, 'The structure of the information visualization design space', *IEEE Symposium on Information Visualization, 1997*, IEEE Computer Society, pp. 92-9.

Carenini, G. & Loyd, J. 2004, 'ValueCharts: analyzing linear models expressing preferences and evaluations', *Proceedings of the working conference on Advanced visual interfaces*, ACM, Gallipoli, Italy, pp. 150-7.

Carpano, M.J. 1980, 'Automatic display of hierarchized graphs for computer-aided decision analysis', *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 10, no. 11, pp. 705-15.

Chen, C. 2004, *Information visualization: Beyond the horizon*, Springer.

Chen, C., Ibekwe-SanJuan, F., SanJuan, E. & Weaver, C. 2006, 'Visual Analysis of Conflicting Opinions', *IEEE Symposium On Visual Analytics Science And Technology, 2006*, pp. 59-66.

Chen, K. & Liu, L. 2006, 'iVIBRATE: Interactive visualization-based framework for clustering large datasets', *ACM Transactions on Information Systems*, vol. 24, no. 2, pp. 245-94.

Chhetri, A. and Zhang, K. 2012, 'Modified RELT for Display and Navigation of Large Hierarchy on Handheld Touch-Screen Device's, Proc. 11th IEEE/ACIS *International Conference on computer and information Science* (ICIS'2012), Shanghai, China, 30 May – 1 June 2012, 147-152.

Chi, E.H., Pitkow, J., Mackinlay, J., Pirolli, P., Gossweiler, R. & Card, S.K. 1998, 'Visualizing the evolution of Web ecologies', *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM Press/Addison-Wesley Publishing Co., Los Angeles, California, United States, pp. 400-7.

Chittaro, L., Ranon, R. & Ieronutti, L. 2006, 'VU-Flow: A Visualization Tool for Analyzing Navigation in Virtual Environments', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1475-85.

Chuah, M.C. 1998, 'Dynamic aggregation with circular visual designs', *IEEE Symposium on Information Visualization, 1998*, IEEE, pp. 35-43, 151.

Cleveland, W.S. & McGill, R. 1984, 'Graphical perception: Theory, experimentation, and application to the development of graphical methods', *Journal of the American Statistical Association*, vol. 79, no. 387, pp. 531-54.

Cook, K.A. & Thomas, J.J. 2005, *Illuminating the Path: The Research and Development Agenda for Visual Analytics*, IEEE computer Society.

Costello, L., Grinstein, G., Plaisant, C. & Scholtz, J. 2009, 'Advancing user-centered evaluation of visual analytic environments through contests', *Information Visualization*, vol. 8, no. 3, p. 230.

Czerwinski, M., Dumais, S., Robertson, G., Dziadosz, S., Tiernan, S. & Van Dantzich, M. 1999, 'Visualizing implicit queries for information management and retrieval', *SIGCHI conference on Human Factors in Computing Systems*, ACM, pp. 560-7.

D'Amico, A.D., Goodall, J.R., Tesone, D.R. & Kopylec, J.K. 2007, 'Visual Discovery in Computer Network Defense', *IEEE Computer Graphics and Applications*, vol. 27, no. 5, pp. 20-7.

Dasgupta, S., Papadimitriou, C.H. & Vazirani, U.V. 2006, 'Algorithms'.

de Berg, M., Onak, K. & Sidiropoulos, A. 2010, 'Fat polygonal partitions with applications to visualization and embeddings', *CoRR*.

de Berg, M., Speckmann, B. & van der Weele, V. 2011, 'Treemaps with Bounded Aspect Ratio', in T. Asano, S.-i. Nakano, Y. Okamoto & O. Watanabe (eds), *Algorithms and Computation*, vol. 7074, Springer Berlin Heidelberg, pp. 260-70.

Dean, A.M., Ellis-Monaghan, J.A., Hamilton, S. & Pangborn, G. 2008, 'Unit Rectangle Visibility Graphs', *The Electronic Journal of Combinatorics*, vol. 15, no. R79, pp. 1-24.

Di Battista, G., Eades, P, Tamassia, R & Tollis, IG 1999, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall, New Jersey.

Doleisch, H. & Hauser, H. 2002, 'Smooth brushing for focus+ context visualization of simulation data in 3D', *Journal of WSCG*, vol. 10, no. 1, pp. 147-54.

Du, Q. & Wang, X. 2004, 'Centroidal Voronoi tessellation based algorithms for vector fields visualization and segmentation', *IEEE Visualization, 2004*, IEEE Computer Society, pp. 43-50.

Dwyer, T., Lee, B., Fisher, D., Quinn, K., Isenberg, P., Robertson, G., North, C. & Inkpen, K. 2009, 'A Comparison of User-Generated and Automatic Graph Layouts', *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 961-8.

Dwyer, T., Lee, B., Fisher, D., Quinn, K.I., Isenberg, P., Robertson, G. & North, C. 2009, 'A comparison of user-generated and automatic graph layouts', *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 961-8.

Eades, P. 1992, *Drawing free trees*, Bulleting of the Institute of Combinatorics and its Applications.

Eades, P., Feng, Q.-W. & Lin, X. 1997, 'Straight-line drawing algorithms for hierarchical graphs and clustered graphs', in S. North (ed.), *Graph Drawing*, vol. 1190, Springer Berlin Heidelberg, pp. 113-28.

Eiglsperger, M., Fekete, S. & Klau, G. 2001, 'Orthogonal Graph Drawing', in M. Kaufmann & D. Wagner (eds), *Drawing Graphs*, vol. 2025, Springer Berlin Heidelberg, pp. 121-71.

Ekpar, F., Yoneda, M. & Hase, H. 2007, 'On the Interactive Visualization of Very Large Image Data Sets', *7th IEEE International Conference on Computer and Information Technology, 2007. CIT 2007*, pp. 627-32.

Elmqvist, N. & Fekete, J. 2010, 'Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines', *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 3, pp. 439-54.

Fan, M., Stallaert, J. & Whinston, A.B. 2000, 'The Internet and the future of financial markets', *Communications of the ACM*, vol. 43, no. 11, pp. 82-8.

Fayyad, U., Grinstein, G. & Wierse, A. 2002, *Information visualization in data mining and knowledge discovery*, Morgan Kaufmann Publication.

File: WorldWideWebAroundWikipedia.png, Wikipedia, the free encyclopedia, <http://en.wikipedia.org/wiki/File:WorldWideWebAroundWikipedia.png>.

Friedman, V. 2008, *Data Visualization and Infographics*, Graphics, Monday Inspiration, <http://www.smashingmagazine.com/2008/01/14/monday-inspiration-data-visualization-and-infographics/>.

Friendly, M. & Denis, D.J. 2001, 'Milestones in the history of thematic cartography, statistical graphics, and data visualization', *Web document, http://www.math.yorku.ca/SCS/Gallery/milestone/*.

Fruchterman, T.M.J. & Reingold, E.M. 1991, 'Graph drawing by force-directed placement', *Software: Practice and Experience*, vol. 21, no. 11, pp. 1129-64.

Furnas, G.W. 1999, 'The FISHEYE view: a new look at structured files', in K.C. Stuart, D.M. Jock & S. Ben (eds), *Readings in information visualization*, Morgan Kaufmann Publishers Inc., pp. 312-30.

Gibson, C.G. 2001, *Elementary geometry of differential curves : an undergraduate introduction*, Cambridge University Press, Cambridge.

Golemati, M., Katifori, A., Giannopoulou, E.G., Daradimos, I., Vassilakis, C., Lepouras, G. & Halatsis, C. 2008, 'An Interview-Based User Study on the use of Visualizations for

Folder Browsing', *12th International Conference on Information Visualisation, 2008. IV '08*, pp. 106-12.

Golub, E. & Shneiderman, B. 2003, 'Dynamic query visualisations on world wide web clients: A DHTML solution for maps and scattergrams', *International journal of Web engineering and technology*, vol. 1, no. 1, pp. 63-78.

Gotz, D. 2011, 'Dynamic Voronoi Treemaps: A Visualization Technique for Time-Varying Hierarchical Data', *IBM Research Report*.

Gotz, D., Zhou, M.X. & Aggarwal, V. 2006, 'Interactive Visual Synthesis of Analytic Knowledge', *IEEE Symposium On Visual Analytics Science And Technology, 2006*, Baltimore, MD, USA, pp. 51-8.

Graham, J.H., Karachiwala, I.S. & Elmaghraby, A.S. 1998, 'Evaluation of a prototype visualization for distributed simulations', *Winter Simulation Conference, 1998*, vol. 2, pp. 1469-77 vol.2.

Gresh, D.L., Rogowitz, B.E., Tignor, M.S. & Mayland, E.J. 1999, 'An interactive framework for visualizing foreign currency exchange options', *Visualization '99*, IEEE Computer Society Press, pp. 453-562.

Grinstein, G., O'Connell, T., Laskowski, S., Plaisant, C., Scholtz, J. & Whiting, M. 2006, 'VAST 2006 Contest - A Tale of Alderwood', *2006 IEEE Symposium On Visual Analytics Science And Technology*, pp. 215-6.

Gross, J. & Hayne, H. 1999, 'Young children's recognition and description of their own and others' drawings', *Developmental Science*, vol. 2, no. 4, pp. 476-89.

Gutwin, C. & Fedak, C. 2004, 'Interacting with big interfaces on small screens: a comparison of fisheye, zoom, and panning techniques', *Conference on Graphics Interface 2004*, Canadian Human-Computer Communications Society, Waterloo, Ontario, Canada, pp. 145-52.

GVU Center, Georgia I. T. , <http://gvu.gatech.edu/>.

Hao, J., Zhang, K. & Huang, M. 2007, 'RELT–visualizing trees on mobile devices', *Advances in Visual Information Systems*, pp. 344-57.

Hao, J. Zhang K., Gabrysch C.A. and Zu Q.M. 2009, 'Managing hierarchical information on small screen', Proc. Joint International conferences on Asia – *Pacific Web conference (APWeb) and web-Age information management(WAIM)*, Suzhou, China, 1-4 April 2009, LNCS, Springer, 429 – 441.

Hauser, H., Ledermann, F. & Doleisch, H. 2002, 'Angular brushing of extended parallel coordinates', *IEEE Symposium on Information Visualization, 2002. INFOVIS 2002.*, pp. 127-30.

Hearst, M.A. 1995, 'TileBars: visualization of term distribution information in full text information access', *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM Press/Addison-Wesley Publishing Co., Denver, Colourado, United States, pp. 59-66.

Heer, J. & Bostock, M. 2010, 'Crowdsourcing graphical perception: using mechanical turk to assess visualization design', *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, Atlanta, Georgia, USA, pp. 203-12.

Hendley, R.J., Drew, N.S., Wood, A.M. & Beale, R. 1995, 'Case study. Narcissus: visualising information', *Information Visualization, 1995*, pp. 90-6.

Herman, I., Melancon, G. & Marshall, M.S. 2000, 'Graph visualization and navigation in information visualization: A survey', *IEEE Transactions on Visualization and Computer Graphics*, vol. 6, no. 1, pp. 24-43.

Hetzler, E. & Turner, A. 2004, 'Analysis experiences using information visualization', *IEEE Computer Graphics and Applications*, vol. 24, no. 5, pp. 22-6.

Holten, D., Isenberg, P., van Wijk, J.J. & Fekete, J. 2011, 'An extended evaluation of the readability of tapered, animated, and textured directed-edge representations in node-link graphs', *IEEE Pacific Visualization Symposium (PacificVis), 2011*, pp. 195-202.

Holten, D. & van Wijk, J.J. 2009, 'A user study on visualizing directed edges in graphs', *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, Boston, MA, USA, pp. 2299-308.

Hong, S.H. & Eades, P. 2005, 'Symmetric Graph Drawing'.

Huang, M. & Nguyen, Q. 2008, 'Context Visualization for Visual Data Mining', in S. Simoff, M. Böhlen & A. Mazeika (eds), *Visual Data Mining*, vol. 4404, Springer Berlin Heidelberg, pp. 248-63.

Huang, M.L., Huang, T.-H. & Zhang, J. 2009, 'TreemapBar: Visualizing Additional Dimensions of Data in Bar Chart', *13th International Conference on Information Visualisation, 2009*, pp. 98-103.

Huang, M.L., Liang, J. & Nguyen, Q.V. 2009, 'A Visualization Approach for Frauds Detection in Financial Market', *13th International Conference on Information Visualisation, 2009*, pp. 197-202.

Huang, W., Eades, P. & Hong, S.H. 2009, 'Measuring effectiveness of graph visualizations: A cognitive load perspective', *Information Visualization*, vol. 8, no. 3, pp. 139-52.

Itoh, T., Takakura, H., Sawada, A. & Koyamada, K. 2006, 'Hierarchical visualization of network intrusion detection data', *IEEE Computer Graphics and Applications*, vol. 26, no. 2, pp. 40-7.

Jeong, C.S. & Pang, A. 1998, 'Reconfigurable disc trees for visualizing large hierarchical information space', *Information Visualization, 1998. Proceedings. IEEE Symposium on*, IEEE, pp. 19-25, 149.

Johnson, B. & Shneiderman, B. 1991, 'Tree-maps: a space-filling approach to the visualization of hierarchical information structures', *IEEE Conference on Visualization, 1991. Visualization '91*, pp. 284-91.

Johnson, C., Moorhead, R., Munzner, T., Pfister, H., Rheingans, P. & Yoo, T. 2006, 'NIH-NSF Visualization Research Challenges Report', IEEE Press, Los Alamitos, CA, USA.

Jungmeister, W.A. & Turo, D. 1992, *Adapting treemaps to stock portfolio visualization*, Tech Report CS-TR-2996, Computer Science Department, University of Maryland, College Park, MD.

Ka-Ping, Y., Fisher, D., Dhamija, R. & Hearst, M. 2001, 'Animated exploration of dynamic graphs with radial layout', *IEEE Symposium on Information Visualization, 2001. INFOVIS 2001*, pp. 43-50.

Kapler, T., Harper, R. & Wright, W. 2005, 'Correlating events with tracked movement in time and space: A geotime case study', *Intelligence Analysis Conference*, Washington, DC, pp. 1-6.

Kaski, S., Sinkkonen, J. & Peltonen, J. 2002, 'Bankruptcy analysis with self-organizing maps in learning metrics', *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 936-47.

Keim, D. 2002, 'Information visualization and visual data mining', *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 1-8.

Keim, D.A., Mansmann, F., Schneidewind, J. & Ziegler, H. 2006, 'Challenges in Visual Data Analysis', *Tenth International Conference on Information Visualization, 2006. IV 2006*, pp. 9-16.

Kerr, B. 2003, 'Thread arcs: an email thread visualization', *Proceedings of the Ninth annual IEEE conference on Information visualization*, IEEE Computer Society, Seattle, Washington, pp. 211-8.

Kim, N.W., Card, S.K. & Heer, J. 2010, 'Tracing genealogical data with TimeNets', *Proceedings of the International Conference on Advanced Visual Interfaces*, ACM, Roma, Italy, pp. 241-8.

Kincaid, R. & Lam, H. 2006, 'Line graph explorer: scalable display of line graphs using Focus+Context', *Proceedings of the working conference on Advanced visual interfaces*, ACM, Venezia, Italy, pp. 404-11.

Kobsa, A. 2004, 'User Experiments with Tree Visualization Systems', *IEEE Symposium on Information Visualization, 2004. INFOVIS 2004*, pp. 9-16.

Kolatch, E. & Weinstein, B. 2001, 'Cattrees: Dynamic visualization of categorical data using treemaps', *Project report*.

Komlodi, A., Rheingans, P., Utkarsha, A., Goodall, J.R. & Amit, J. 2005, 'A user-centered look at glyph-based security visualization', *IEEE Workshop on Visualization for Computer Security, 2005. (VizSEC 05)*, pp. 21-8.

Kong, N., Heer, J. & Agrawala, M. 2010, 'Perceptual guidelines for creating rectangular treemaps', *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 990-8.

Kosara, R., Healey, C., Interrante, V., Laidlaw, D. & Ware, C. 2003, 'Thoughts on user studies: Why, how, and when', *IEEE Computer Graphics and Applications*, vol. 23, no. 4, pp. 20-5.

Kosara, R., Miksch, S. & Hauser, H. 2001, 'Semantic depth of field', *IEEE Symposium on Information Visualization, 2001. INFOVIS 2001*, pp. 97-104.

Kreuseler, M. & Schumann, H. 1999, 'Information visualization using a new focus+context technique in combination with dynamic clustering of information space', *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*, ACM, Kansas City, Missouri, United States, pp. 1-5.

Kruger, J., Kipfer, P., Konclratieva, P. & Westermann, R. 2005, 'A particle system for interactive visualization of 3D flows', *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 6, pp. 744-56.

Lai, P.-S. & Fu, H.-C. 2007, 'A polygon description based similarity measurement of stock market behavior', *IEEE Congress on Evolutionary Computation, 2007. CEC 2007*, pp. 806-12.

Lamping, J. & Rao, R. 1996, 'The Hyperbolic Browser: A Focus+Context Technique for Visualizing Large Hierarchies', *Journal of Visual Languages & Computing*, vol. 7, no. 1, pp. 33-55.

Lee, B., Parr, C.S., Plaisant, C., Bederson, B.B., Veksler, V.D., Gray, W.D. & Kotfila, C. 2006, 'TreePlus: Interactive Exploration of Networks with Enhanced Tree Layouts', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1414-26.

Lehmann, D., Stuart, J., Johar, G. & Thozhur, A. 2007, 'Spontaneous visualization and concept evaluation', *Journal of the Academy of Marketing Science*, vol. 35, no. 3, pp. 309-16.

Liang, J. & Huang, M.L. 2010, 'Highlighting in Information Visualization: A Survey', *14th International Conference on Information Visualisation (IV), 2010*, pp. 79-85.

Liqun, J. & Banks, D.G. 1997, 'TennisViewer: a browser for competition trees', *Computer Graphics and Applications, IEEE*, vol. 17, no. 4, pp. 63-5.

Liston, K., Fischer, M. & Kunz, J. 2000, 'Designing and evaluating visualization techniques for construction planning', *Eighth International Conference on Computing in Civil and Building Engineering (ICCCBE-VIII)*, Stanford, California, United States, pp. 1293-300.

Liu, S., Cao, N. & Lv, H. 2008, 'Interactive Visual Analysis of the NSF Funding Information', *IEEE Pacific Visualization Symposium, 2008. PacificVIS '08*, pp. 183-90.

Lu, L.F., Zhang, J.W., Huang, M.L. & Fu, L. 2010, 'A new concentric-circle visualization of multi-dimensional data and its application in network security', *Journal of Visual Languages & Computing*, vol. 21, no. 4, pp. 194-208.

MacEachren, A.M., Hardisty, F., Dai, X. & Pickle, L. 2003, 'Supporting visual analysis of federal geospatial statistics', *Communications of the ACM*, vol. 46, no. 1, pp. 59-60.

Mann, T.M. & Reiterer, H. 2000, 'Evaluation of different visualizations of Web search results', *11th International Workshop on Database and Expert Systems Applications, 2000*, pp. 586-90.

Mansmann, F. & Vinnik, S. 2006, 'Interactive Exploration of Data Traffic with Hierarchical Network Maps', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1440-9.

Mao Lin, H., Quang Vinh, N., Wei, L. & Xiaodi, H. 2007, 'Three-Dimensional EncCon Tree', *Computer Graphics, Imaging and Visualisation, 2007. CGIV '07*, pp. 429-33.

Marshall, S., Biddle, R. & Noble, J. 2004, 'Using software visualisation to enhance online component markets', *Proceedings of the 2004 Australasian symposium on Information Visualisation - Volume 35*, Australian Computer Society, Inc., Christchurch, New Zealand, pp. 35-41.

Martin, A.R. & Ward, M.O. 1995, 'High Dimensional Brushing for Interactive Exploration of Multivariate Data', *IEEE Conference on Visualization, 1995. Visualization '95*, pp. 271-8.

Martin, G.E. 1982, *Transformation geometry : an introduction to symmetry*, Springer-Verlag, New York.

Mascoet, M. 2001, 'Interaction and visualization supporting Web browsing patterns', *Fifth International Conference on Information Visualisation, 2001*, pp. 413-8.

McGrath, C., Blythe, J. & Krackhardt, D. 1996, 'Seeing Groups in Graph Layouts1', *Connections*, vol. 19, no. 2, pp. 22-9.

McGuffin, M. & Robert, J. 2009, 'Quantifying the space-efficiency of 2D graphical representations of trees', *Information Visualization*, vol. 9, no. 2, pp. 115-40.

McGuffin, M.J. & Balakrishnan, R. 2005, 'Interactive visualization of genealogical graphs', *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005*, pp. 16-23.

McGuffin, M.J., Davison, G. & Ravin, B. 2004, 'Expand-Ahead: A Space-Filling Strategy for Browsing Trees', *IEEE Symposium on Information Visualization, 2004. INFOVIS 2004*, pp. 119-26.

Melancon, G. & Herman, I. 1998, *Circular drawings of rooted trees*, Technical Report of the Centre for Mathematics and Computer Sciences, INS-R9817, Amsterdam, The Netherlands.

Merino, C.S., Sips, M., Keim, D.A., Panse, C. & Spence, R. 2006, 'Task-at-hand interface for change detection in stock market data', *Proceedings of the working conference on Advanced visual interfaces*, ACM, Venezia, Italy, pp. 420-7.

Misue, K., Eades, P., Lai, W. & Sugiyama, K. 1995, 'Layout Adjustment and the Mental Map', *Journal of Visual Languages & Computing*, vol. 6, no. 2, pp. 183-210.

Nesbitt, K.V. & Barrass, S. 2004, 'Finding trading patterns in stock market data', *IEEE Computer Graphics and Applications*, vol. 24, no. 5, pp. 45-55.

Nguyen, Q. & Huang, M. 2005, 'EncCon: an approach to constructing interactive visualization of large hierarchical data', *Information Visualization*, vol. 4, no. 1, pp. 1-21.

Nguyen, Q.V. & Huang, M.L. 2003, 'Space-optimized tree: a connection+enclosure approach for the visualization of large hierarchies', *Information Visualization*, vol. 2, no. 1, pp. 3-15.

Nguyen, Q.V. & Huang, M.L. 2004, 'A focus+context visualization technique using semi-transparency', *The Fourth International Conference on Computer and Information Technology, 2004. CIT '04*, IEEE Computer Society, pp. 101-8.

Nguyen, T.N. & Zhang, J. 2006, 'A Novel Visualization Model for Web Search Results', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 981-8.

Nowell, L., Schulman, R. & Hix, D. 2002, 'Graphical encoding for information visualization: an empirical study', *IEEE Symposium on Information Visualization, 2002. INFOVIS 2002*, pp. 43-50.

Onak, K. & Sidiropoulos, A. 2008, 'Circular partitions with applications to visualization and embeddings', *Proceedings of the twenty-fourth annual symposium on Computational geometry*, ACM, College Park, MD, USA, pp. 28-37.

Parrish, E. 2000, 'StockVis: An Internet-Based System for Visualizing Stock Market Data', Master thesis, UC Santa Cruz, Department of Computer Science.

Plaisant, C., Grosjean, J. & Bederson, B.B. 2002, 'SpaceTree: supporting exploration in large node link tree, design evolution and empirical evaluation', *IEEE Symposium on Information Visualization, 2002. INFOVIS 2002*, pp. 57-64.

Plumlee, M. & Ware, C. 2006, 'Zooming versus multiple window interfaces: Cognitive costs of visual comparisons', *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 13, no. 2, p. 209.

Purchase, H., Carrington, D. & Allder, J.-A. 2002, 'Empirical Evaluation of Aesthetics-based Graph Layout', *Empirical Software Engineering*, vol. 7, no. 3, pp. 233-55.

Purchase, H.C., Pilcher, C. & Plimmer, B., 'Graph Drawing Aesthetics - Created by Users, Not Algorithms', *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 1, pp. 81-92.

Ramey, N.A., Corso, J.J., Lau, W.W., Burschka, D. & Hager, G.D. 2004, 'Real-time 3D Surface Tracking and Its Applications', *Conference on Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04*, p. 34.

Ramos, E. & Donoho, D. 1983, *The 1983 ASA data exposition dataset: Cars*, <http://lib.stat.cmu.edu/datasets/cars/data>.

Ramsay, A. & Richtmyer, R.D. 1995, *Introduction to hyperbolic geometry*, Springer-Verlag, New York.

Redish, J. 2007, 'Expanding usability testing to evaluate complex systems', *Journal of Usability Studies*, vol. 2, no. 3, pp. 102-11.

Reingold, E.M. & Tilford, J.S. 1981, 'Tidier Drawings of Trees', *IEEE Transactions on Software Engineering*, vol. SE-7, no. 2, pp. 223-8.

Robinson, A.C. 2009, 'Visual highlighting methods for geovisualization', *24th International Cartographic Conference*.

Robinson, A.C. & Center, G. 2006, 'Highlighting techniques to support geovisualization', *Proceedings of the ICA Workshop on Geovisualization and Visual Analytics*.

Roe, J. 1993, *Elementary geometry*, Oxford University Press, Oxford, UK.

Saraiya, P., North, C. & Duca, K. 2005, 'An insight-based methodology for evaluating bioinformatics visualizations', *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 4, pp. 443-56.

Saraiya, P., North, C., Lam, V. & Duca, K.A. 2006, 'An Insight-Based Longitudinal Study of Visual Analytics', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1511-22.

Schaffer, D., Zuo, Z., Greenberg, S., Bartram, L., Dill, J., Dubs, S. & Roseman, M. 1996, 'Navigating hierarchically clustered networks through fisheye and full-zoom methods', *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 3, no. 2, pp. 162-88.

Scholtz, J. 2006, 'Beyond Usability: Evaluation Aspects of Visual Analytic Environments', *IEEE Symposium On Visual Analytics Science And Technology, 2006*, pp. 145-50.

Schulz, H., Hadlak, S. & Schumann, H. 2011a, 'The design space of implicit hierarchy visualization: A survey', *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 4, pp. 393-411.

Schulz, H.J. 2011, 'Treevis.net: A Tree Visualization Reference', *Computer Graphics and Applications, IEEE*, vol. 31, no. 6, pp. 11-5.

Schulz, H.J., Hadlak, S. & Schumann, H. 2011b, 'Point-based visualization for large hierarchies', *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 5, pp. 598-611.

Seo, J. & Shneiderman, B. 2005, 'A rank-by-feature framework for interactive exploration of multidimensional data', *Information Visualization*, vol. 4, no. 2, pp. 96-113.

Shaverdian, A.A., Zhou, H., Michailidis, G. & Jagadish, H. 2010, 'An Algebra for Visual Analysis'.

Shekhar, S. & Oliver, D. 2010, 'Computational modeling of spatio-temporal social networks: A time-aggregated graph approach', *Proceedings of Spatio-Temporal Constraints on Social Network*.

Shen, Z., Ma, K.L. & Eliassi-Rad, T. 2006, 'Visual Analysis of Large Heterogeneous Social Networks by Semantic and Structural Abstraction', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1427-39.

Shi, K., Irani, P. & Li, B. 2005, 'An evaluation of content browsing techniques for hierarchical space-filling visualizations', *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005*, pp. 81-8.

Shneiderman, B. 1992, 'Tree visualization with tree-maps: 2-d space-filling approach', *ACM Transactions on graphics (TOG)*, vol. 11, no. 1, pp. 92-9.

Shneiderman, B. 1998, 'Treemaps for space-constrained visualization of hierarchies', *ACM Transactions on Graphics (TOG)* vol. 11, pp. 92-9.

Shreck, T., Tekusova, T. & Kohlhammer, J. 2007, 'Trajectory-based visual analysis of large financial time series data', *SIGKDD Explorations*, vol. 9, no. 2, pp. 30-7.

Siirtola, H. & Räihä, K. 2006, 'Interacting with parallel coordinates', *Interacting with Computers*, vol. 18, no. 6, pp. 1278-309.

Simunic, K. 2003, 'Visualization of stock market charts', *Proceedings of the International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG)*.

Slocum, J. 2001, *The Tao of Tangram*, Barnes & Noble.

Stasko, J., Catrambone, R., Guzdial, M. & McDonald, K. 2000, 'An evaluation of space-filling information visualizations for depicting hierarchical structures', *International Journal of Human-Computer Studies*, vol. 53, no. 5, pp. 663-94.

Stasko, J. & Zhang, E. 2000, 'Focus+ context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations', *IEEE Symposium on Information Visualization, 2000. InfoVis 2000*, IEEE, pp. 57-65.

Strausfeld, L. 1995, 'Embodying Virtual Space to Enhance the Understanding of Information', Master thesis, Massachusetts Institute of Technology, Program in Media Art and Sciences.

Sud, A., Fisher, D. & Huai-Ping, L. 2010, 'Fast Dynamic Voronoi Treemaps', *2010 International Symposium on Voronoi Diagrams in Science and Engineering (ISVD)*, pp. 85-94.

Sugiyama, K. & Misue, K. 1995, 'Graph Drawing by the Magnetic Spring Model', *Journal of Visual Languages & Computing*, vol. 6, no. 3, pp. 217-31.

Sugiyama, K., Tagawa, S. & Toda, M. 1981, 'Methods for visual understanding of hierarchical system structures', *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 11, no. 2, pp. 109-25.

Tamassia, R., Di Battista, G. & Batini, C. 1988, 'Automatic graph drawing and readability of diagrams', *IEEE Transactions on Systems, Man and Cybernetics*, vol. 18, no. 1, pp. 61-79.

Tangram People Set, viewed 20 June 2010, <http://www.istockphoto.com/stock-illustration-3652877-tangram-people-set-003.php>.

Tatemura, J., Santini, S. & Jain, R. 1999, 'Social and content-based approach for visual recommendation of Web graphics', *IEEE Symposium on Visual Languages, 1999*, pp. 200-1.

Teoh, S. & Kwan-Liu, M. 2002, 'RINGS: A Technique for Visualizing Large Hierarchies', in M. Goodrich & S. Kobourov (eds), *Graph Drawing*, vol. 2528, Springer Berlin Heidelberg, pp. 268-75.

Teoh, S.T., Ma, K.-L., Wu, S.F. & Jankun-Kelly, T.J. 2004, 'Detecting flaws and intruders with visual data analysis', *IEEE Computer Graphics and Applications*, vol. 24, no. 5, pp. 27-35.

Theilhaber 2004, *BMC Bioinformatics*, <http://www.biomedcentral.com/1471-2105/5/195/figure/F8>.

Thomas, J.J. & Cook, K.A. 2006, 'A visual analytics agenda', *IEEE Computer Graphics and Applications*, vol. 26, no. 1, pp. 10-3.

Tseng, F., Tzeng, G., Yu, H. & Yuan, B. 2001, 'Fuzzy ARIMA model for forecasting the foreign exchange market', *Fuzzy sets and systems*, vol. 118, no. 1, pp. 9-19.

Tue, D.H., Bazinet, A., Berthier, R. & Shneiderman, B. 2008, 'NASDAQ Velocity and Forces: An Interactive Visualization of Activity and Change', *Journal of Universal Computer Science*, vol. 14, pp. 1391-410.

Turo, D. & Johnson, B. 1992, 'Improving the visualization of hierarchies with treemaps: design issues and experimentation', *IEEE Conference on Visualization, 1992. Visualization '92*, pp. 124-31.

van Ham, F. & Perer, A. 2009, '"Search, Show Context, Expand on Demand": Supporting Large Graph Exploration with Degree-of-Interest', *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 953-60.

van Ham, F. & Rogowitz, B. 2008, 'Perceptual organization in user-generated graph layouts', *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1333-9.

van Wijk, J.J. 2006, 'Bridging the Gaps', *IEEE Computer Graphics and Applications*, vol. 26, no. 6, pp. 6-9.

Van Wijk, J.J. & Van de Wetering, H. 1999, 'Cushion treemaps: visualization of hierarchical information', *1999 IEEE Symposium on Information Visualization, 1999. (Info Vis '99)*, pp. 73-8, 147.

Villarroel, M., de la Fuente, P., Pedrero, A. & Adiego, J. 2006, 'Visualizing Shared Highlighting Annotations', in R. Navarro-Prieto & J.L. Vidal (eds), *HCI related papers of Interacción 2004*, Springer Netherlands, pp. 195-204.

von Landesberger, T., Kuijper, A., Schreck, T., Kohlhammer, J., VAN WIJK, J., FEKETE, J. & Fellner, D. 2010, 'Visual analysis of large graphs', *Proceedings of Euro-Graphics: State of the Art Report*, vol. 2, no. 3.

Walter, J., Ontrup, J., Wessling, D. & Ritter, H. 2003, 'Interactive visualization and navigation in large data collections using the hyperbolic space', *Third IEEE International Conference on Data Mining, 2003. ICDM 2003*, pp. 355-62.

Ware, C. 2004, *Information visualization: perception for design*, Morgan Kaufmann.

Ware, C. & Bobrow, R. 2004, 'Motion to support rapid interactive queries on node--link diagrams', *ACM Transactions on Applied Perception (TAP)*, vol. 1, no. 1, pp. 3-18.

Ware, C. & Bobrow, R. 2005, 'Supporting visual queries on medium-sized node-link diagrams', *Information Visualization*, vol. 4, no. 1, pp. 49-58.

Ware, C., Gilman, A. & Bobrow, R. 2007, 'Visual Thinking with an Interactive Social Network Diagram', *Diagrams 2008*, pp. 118-26.

Ware, C., Gilman, A. & Bobrow, R. 2008, 'Visual Thinking with an Interactive Diagram', in G. Stapleton, J. Howse & J. Lee (eds), *Diagrammatic Representation and Inference*, vol. 5223, Springer Berlin Heidelberg, pp. 118-26.

Ware, C., Purchase, H., Colpoys, L. & McGill, M. 2002, 'Cognitive measurements of graph aesthetics', *Information Visualization*, vol. 1, no. 2, pp. 103-10.

Warfield, J.N. 1977, 'Crossing theory and hierarchy mapping', *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 7, no. 7, pp. 505-23.

Wattenberg, M. 1998, *Map of the Market*, SmartMoney. com, viewed 18/03/2010, <http://www.smartmoney.com/map-of-the-market/>.

Wattenberg, M. 1999, 'Visualizing the stock market', *CHI '99 Extended Abstracts on Human Factors in Computing Systems*, ACM, Pittsburgh, Pennsylvania, pp. 188-9.

Wehrend, S. & Lewis, C. 1990, 'A problem-oriented classification of visualization techniques', *Proceedings of the First IEEE Conference on Visualization, 1990. Visualization '90*, pp. 139-43, 469.

Wexelblat, A. & Maes, P. 1999, 'Footprints: history-rich tools for information foraging', *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, ACM, Pittsburgh, Pennsylvania, United States, pp. 270-7.

Wills, G.J. 1999, 'NicheWorks—Interactive Visualization of Very Large Graphs', *Journal of Computational and Graphical Statistics*, vol. 8, no. 2, pp. 190-212.

Winckler, M.A., Palanque, P. & Freitas, C.M.D.S. 2004, 'Tasks and scenario-based evaluation of information visualization techniques', *Proceedings of the 3rd annual conference on Task models and diagrams*, ACM, Prague, Czech Republic, pp. 165-72.

Wiss, U., Carr, D. & Jonsson, H. 1998, 'Evaluating three-dimensional information visualization designs: a case study of three designs', *IEEE Conference on Information Visualization, 1998*, pp. 137-44.

Wong, P., Foote, H., Chin, G., Mackey, P. & Perrine, K. 2006, 'Graph signatures for visual analytics', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1399-413.

Wong, P.C., Foote, H., Mackey, P., Perrine, K. & Chin, G. 2006, 'Generating Graphs for Visual Analytics through Interactive Sketching', *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 6, pp. 1386-98.

Xiong, F., Prakash, E. & Ho, K. 2002, 'ER modeling and visualization of large mutual fund data', *Journal of Visualization*, vol. 5, no. 2, pp. 197-204.

Yang, J., Ward, M.O. & Rundensteiner, E.A. 2002, 'InterRing: an interactive tool for visually navigating and manipulating hierarchical structures', *IEEE Symposium on Information Visualization, 2002. INFOVIS 2002*, pp. 77-84.

Yoo, H.Y. & Yoo, H.J. 2005, 'Visualizing hierachical information using a new focus+context method', *Fourth Annual ACIS International Conference on Computer and Information Science, 2005*, pp. 573-6.

Zhang, D. & Zhou, L. 2004, 'Discovering golden nuggets: data mining in financial application', *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 34, no. 4, pp. 513-22.

Zhou, N., Wu, J., Wang, B. & Zhang, S. 2008, 'A Visualization Model for Information Resources Management', *12th International Conference Information Visualisation, 2008. IV '08*, pp. 57-62.

Ziegler, H., Nietzschmann, T. & Keim, D.A. 2007, 'Visual Exploration and Discovery of Atypical Behavior in Financial Time Series Data using Two-Dimensional Colourmaps', *11th International Conference Information Visualization, 2007. IV '07*, pp. 308-15.

Ziemkiewicz, C. & Kosara, R. 2008, 'The shaping of information by visual metaphors', *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1269-76.

Tangram Treemaps 2012

This thesis proposes a new enclosure visualization method, named Tangram Treemaps that achieves the maximization of the computer screen utilization through the flexibility of display shapes. Breaking through the limitation of rectangular constraint, the new approach is able to partition various polygonal shapes. Furthermore, our algorithms also improve the efficiency of interactive tree visualization significantly, through the reduction of the computational cost.

An enclosure geometrical partitioning method with various shapes