Hierarchical Data Analysis

Tree Map Algorithms

Ira A. Fulton Schools of Engineering
Arizona State University

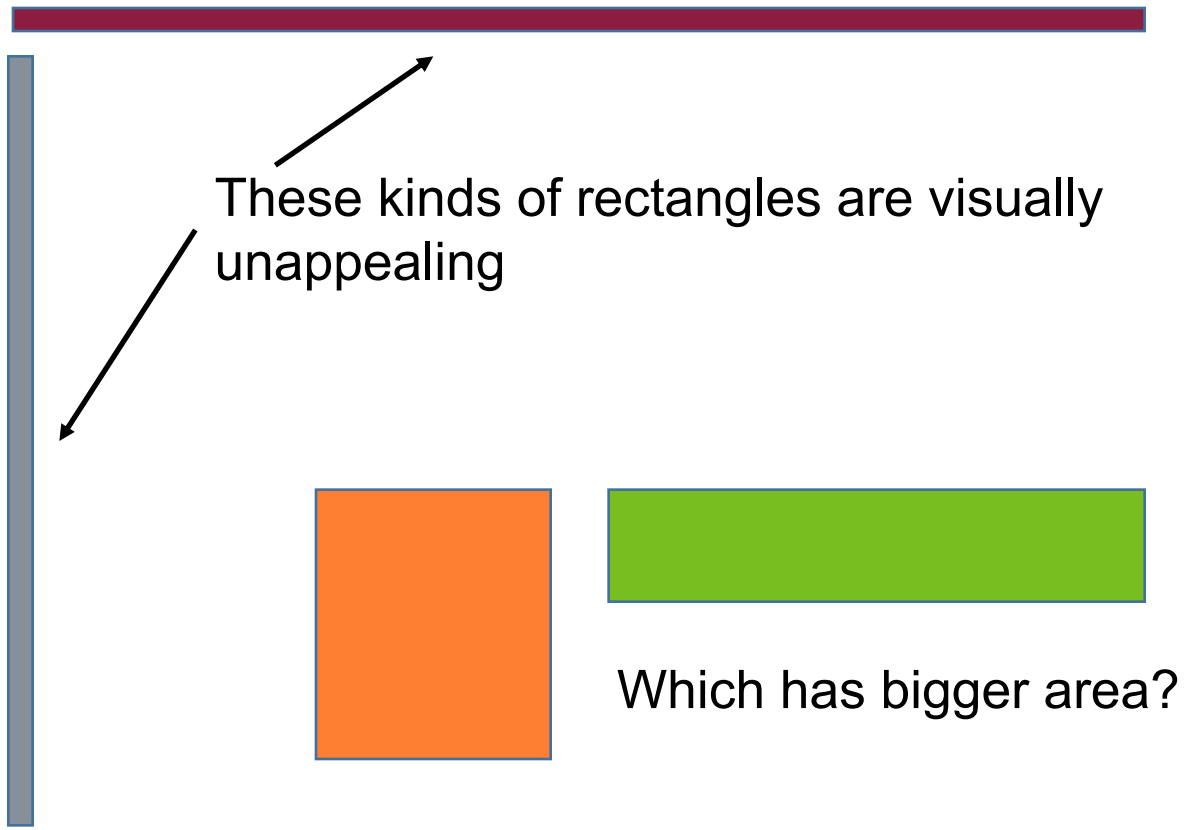
Objective

Objective

Explain hierarchical representation schemes

Aspect Ratios

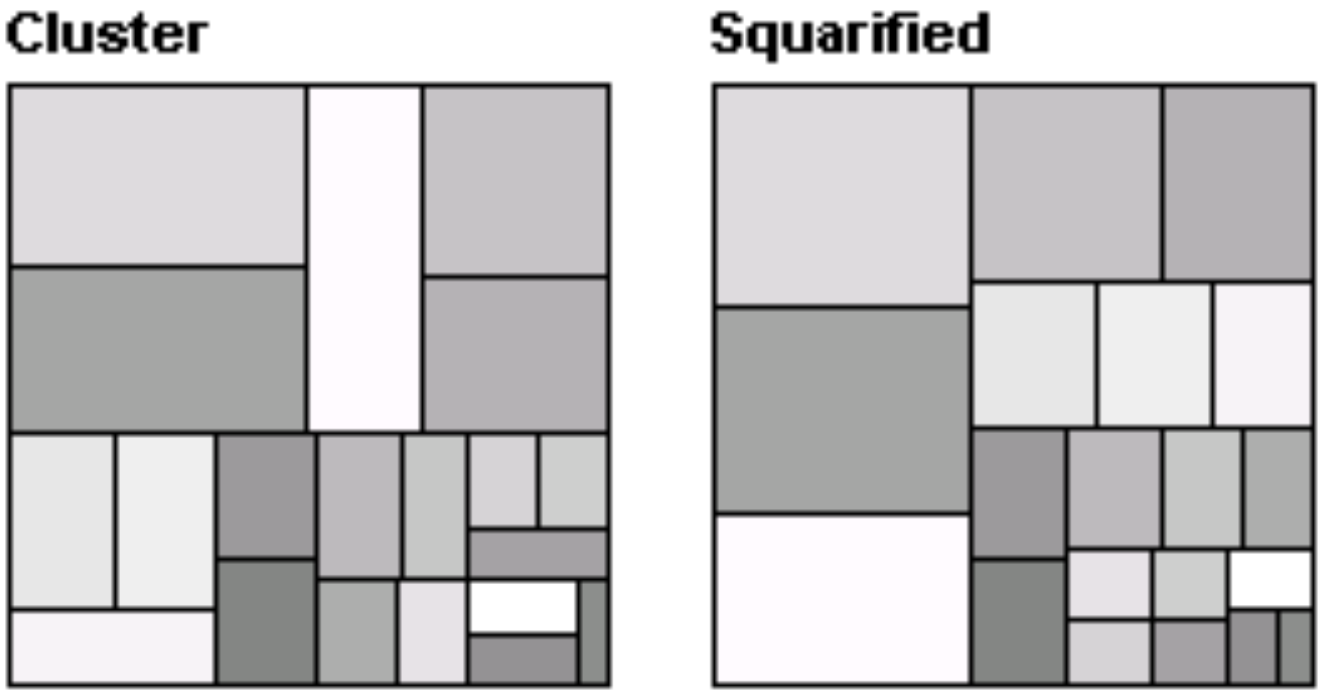
- Cleveland's "Banking to 45"
- Here, the aspect ratio will drastically affect the visualization



Parts of this slide borrowed from John Stasko's Information Visualization Course Lecture: Hierarchies and Trees 2(Space Filling): http://www.cc.gatech.edu/~stasko/7450/Notes/tree2.pdf

Clustered and Squarified Treemaps

- Simple recursive algorithm to reduce overall aspect ratio
- Bruls et al. introduced squarified treemap



1 -Wattenberg, M. “Visualizing the Stock Market,” Proceedings of ACM CHI 99, Extended Abstracts, pp.188-189, 1999.

2 -Bruls, D.M., C. Huizing, J.J. van Wijk. “Squarified Treemaps”. In: W. de Leeuw, R. van Liere (eds.), Data Visualization 2000, Proceedings of the joint Eurographics and IEEE TCVG Symposium on Visualization, 2000, pp. 33-42.

Clustered and Squarified Treemaps

Methods had two major drawbacks

- Changes in the set can cause discontinuities in the layout
  - If treemap data is dynamic large visual changes make data hard to track
- If the data has ordering information this is not preserved

1 -Wattenberg, M. “Visualizing the Stock Market,” Proceedings of ACM CHI 99, Extended Abstracts, pp.188-189, 1999. 2 - Bruls, D.M., C. Huizing, J.J. van Wijk. “Squarified Treemaps”. In: W. de Leeuw, R. van Liere (eds.), Data Visualization 2000, Proceedings of the joint Eurographics and IEEE TCVG Symposium on Visualization, 2000, pp. 33-42.

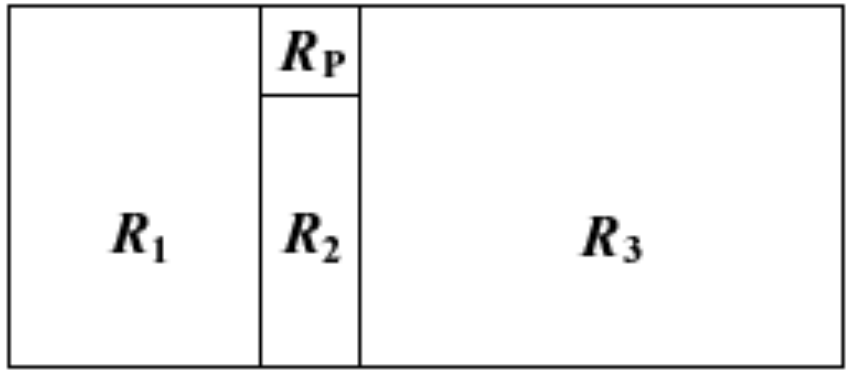
Ordered Treemap

- Shneiderman and Wattenberg introduced the ordered treemap to try and overcome these limitations
- Possible to create a layout in which items that are next to each other in given order are adjacent in the tree map
- Presented two algorithms for ordering a treemap

Ordered Treemap Algorithm

Starting with a rectangle R to be subdivided, first algorithm pivot-by-size, the pivot is item with largest area

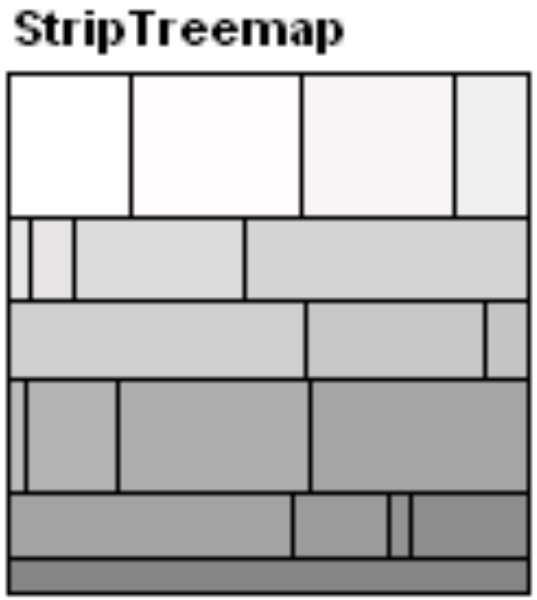
1. Let P, the pivot, be the item with largest area in list of items
2. If width R is greater than or equal to the height, divide R into four rectangle
3. Place P in the rectangle R_p
4. Divide the items in the list, other than P, into three lists, L_1, L_2, L_3 to be laid out in R_1, R_2 and R_3.
5. Recursively lay out L_1, L_2 and L_3 in R_1, R_2 and R_3



Ben Shneiderman, Martin Wattenberg: Ordered Treemap Layouts. INFOVIS 2001: 73-78

Strip Treemaps

1. Scale the area of all rectangles so total area of input rectangles equals that of layout rectangle
2. Create a new empty strip, the current strip
3. Add next rectangle to current strip, recomputing height of strip based on area of all rectangles within the strip and recomputing width of each rectangle
4. If average aspect ratio of the current strip has increased as a result of adding rectangle in step 3, remove rectangle pushing it back onto list of rectangles and go to step 2
5. If all rectangles have been processed stop, else step 3



Benjamin B. Bederson, Ben Shneiderman, Martin Wattenberg: Ordered and quantum treemaps: Making effective use of 2D space to display hierarchies. ACM Trans. Graph. 21(4): 833-854 (2002)

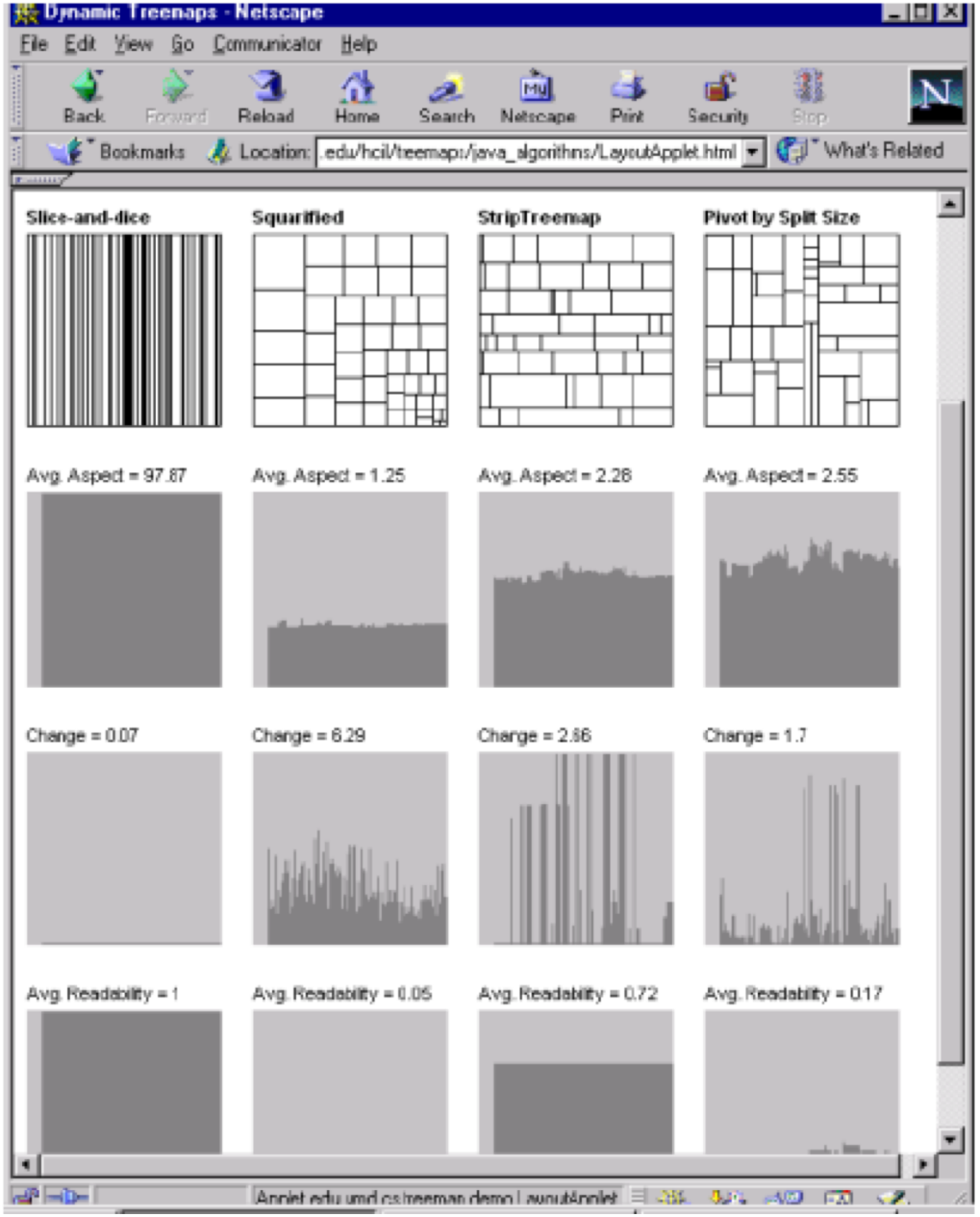
Metrics For Treemaps

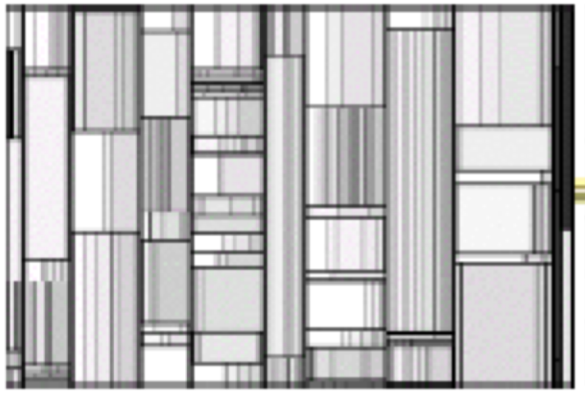
- In order to assess all these different treemap algorithms, we need metrics to define how “good” they are

- Use two metrics:
  - Average aspect ratio of treemap layout
  - Layout distance change function

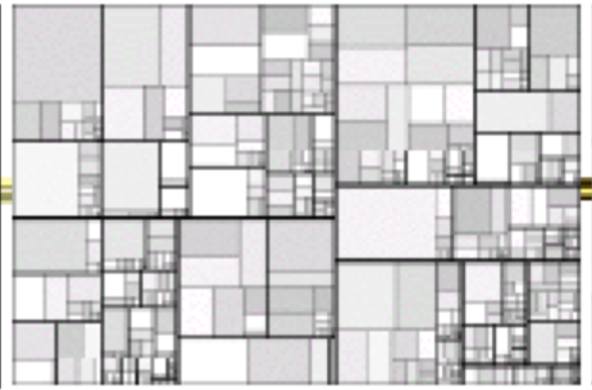
Metrics For Treemaps

- Goal is to have low average aspect ratio and a low distance change as data is updated
- Average aspect ratio is the unweighted average
- Distance change is Euclidian distance of change in width height and corner location of rectangles

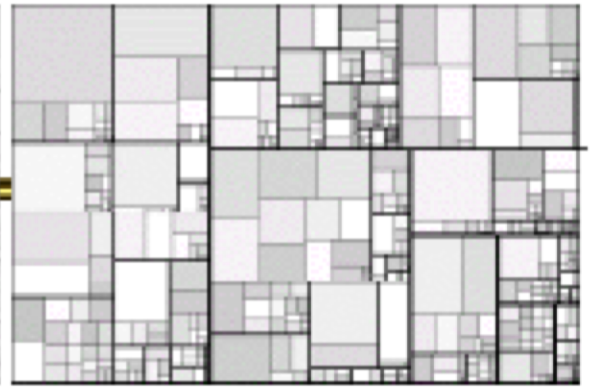

Dynamic Treemaps - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: .edu/hcil/treemap/java_algorithms/LayoutApplet.html What's Related
Slice-and-dice
Squarified
StripTreemap
Pivot by Split Size
Avg. Aspect = 97.87
Avg. Aspect = 1.25
Avg. Aspect = 2.28
Avg. Aspect = 2.55
Change = 0.07
Change = 6.29
Change = 2.86
Change = 1.7
Avg. Readability = 1
Avg. Readability = 0.05
Avg. Readability = 0.72
Avg. Readability = 0.17



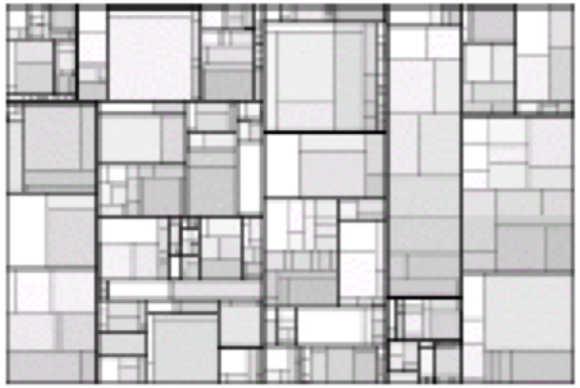
Slice-and-dice



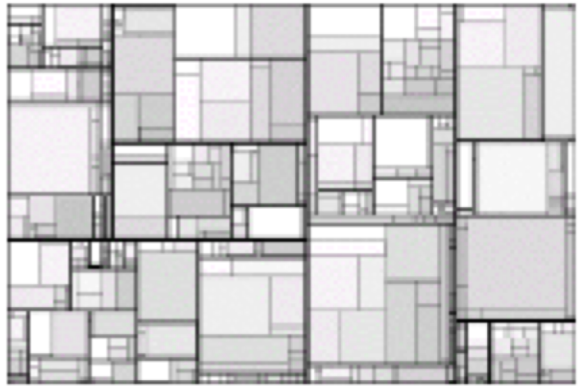
Cluster



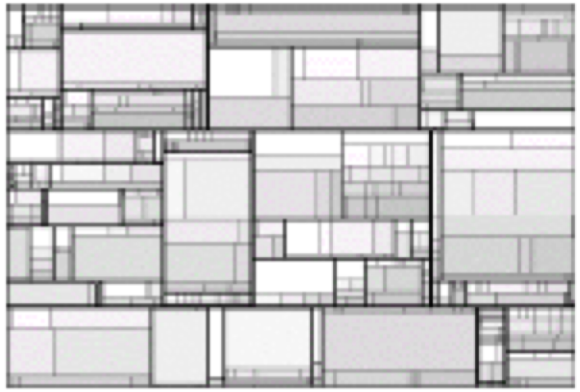
Squarified



Pivot-by-middle



Pivot-by-size



Strip

Showing Structure

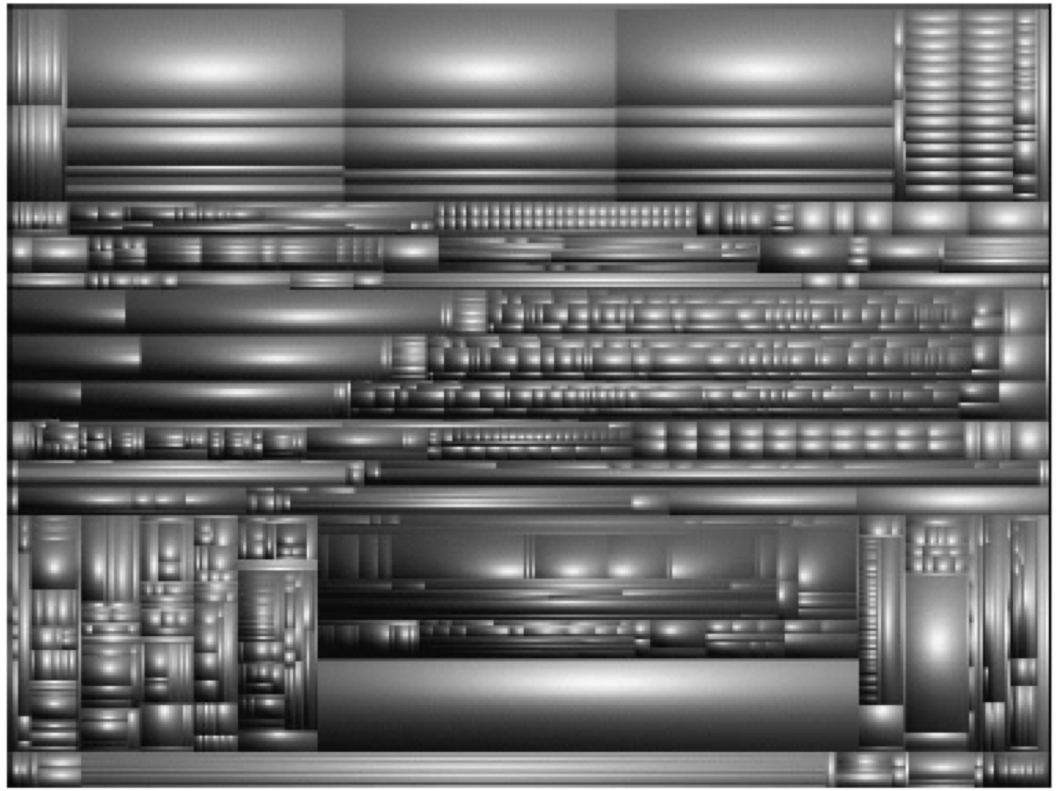
- Regular borderless treemap makes it challenging to discern structure of hierarchy, particularly large ones
  - Supplement treemap view
  - Change rectangles to other forms

newsmap

Parts of this slide borrowed from John Stasko's Information Visualization Course Lecture: Hierarchies and Trees 2(Space Filling): http://www.cc.gatech.edu/~stasko/7450/Notes/tree2.pdf

Cushion Treemap

- Use shading and texture to help convey structure of hierarchy



Van Wijk, J.J.; Van de Wetering, H. (1999), "Cushion treemaps: visualization of hierarchical information", Proc. IEEE Symp. Information Visualization (Info Vis '99), pp. 73–78

Another Problem

- What if nodes with zero value are very important?
- If we're mapping areas, how do we map to zero?
  - Example: Stocks portfolios

Parts of this slide borrowed from John Stasko's Information Visualization Course Lecture: Hierarchies and Trees 2(Space Filling): http://www.cc.gatech.edu/~stasko/7450/Notes/tree2.pdf

Context Treemap

- One way to overcome this is to distort classic treemap visualization

- Distorted treemap can show one more attribute than a classic treemap
  - node area is no longer proportional to attribute being visualized

Christoph Csallner, Marcus Handte, Othmar Lehmann, John T. Stasko: FundExplorer: Supporting the Diversification of Mutual Fund Portfolios Using Context Treemaps. INFOVIS 2003: 203-208

Context Treemap

- Several different implementation strategies for this
  1. Use a regular tree map but add some epsilon to zero value items
  2. Use an exponential mapping area(node)=2^(value(node))
  3. Assign some minimal screen space size to zero nodes

Christoph Csallner, Marcus Handte, Othmar Lehmann, John T. Stasko: FundExplorer: Supporting the Diversification of Mutual Fund Portfolios Using Context Treemaps. INFOVIS 2003: 203-208

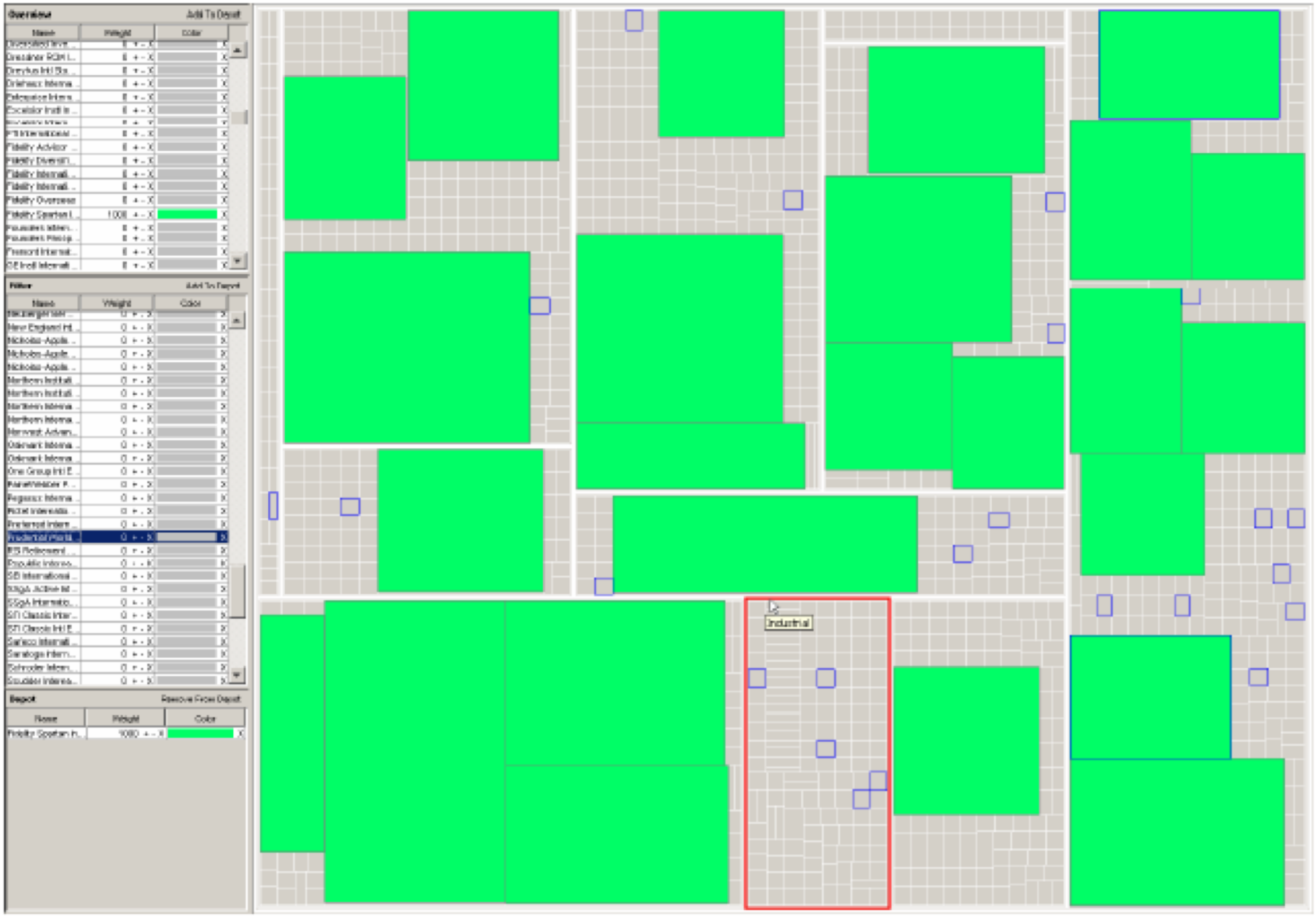
Context Treemap

Final solution was to calculate intermediate values

1. Calculate the total (in this paper it was total invest money)
   - Value(total)
2. Create an additional total with respect to the context
   - Value(total)*v, where v can be modified as a scale factor
3. Split context screen real estate among all empty nodes
   - $Value_c$ = value(total)*v/#empty

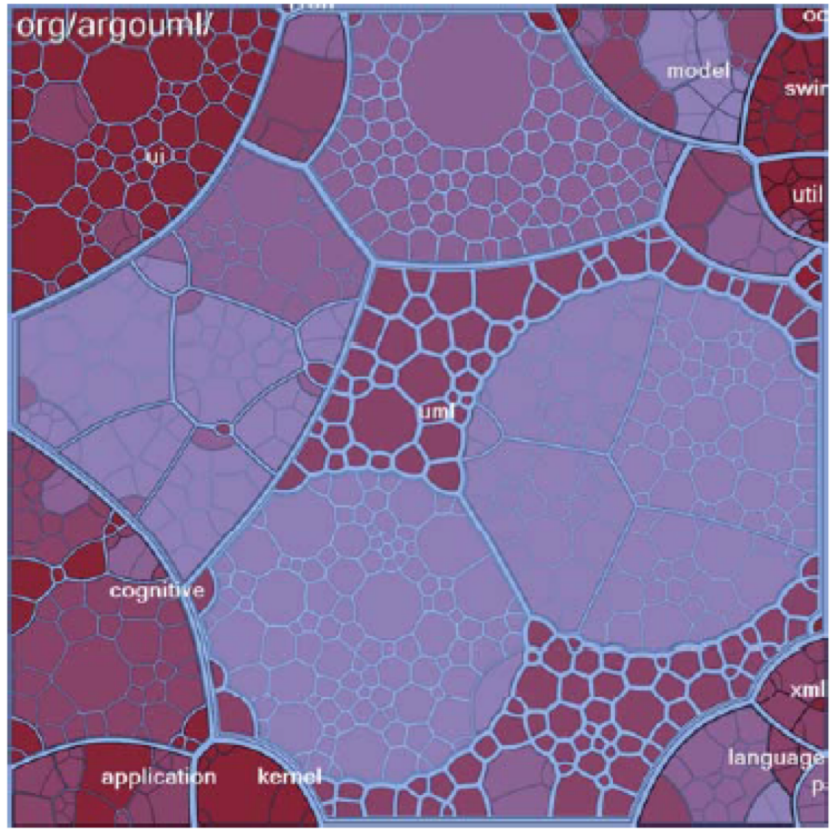
$$value`(node) = \begin{cases} value_c & \text{if } value(node) = 0 \\ value(node) & \text{otherwise} \end{cases}$$

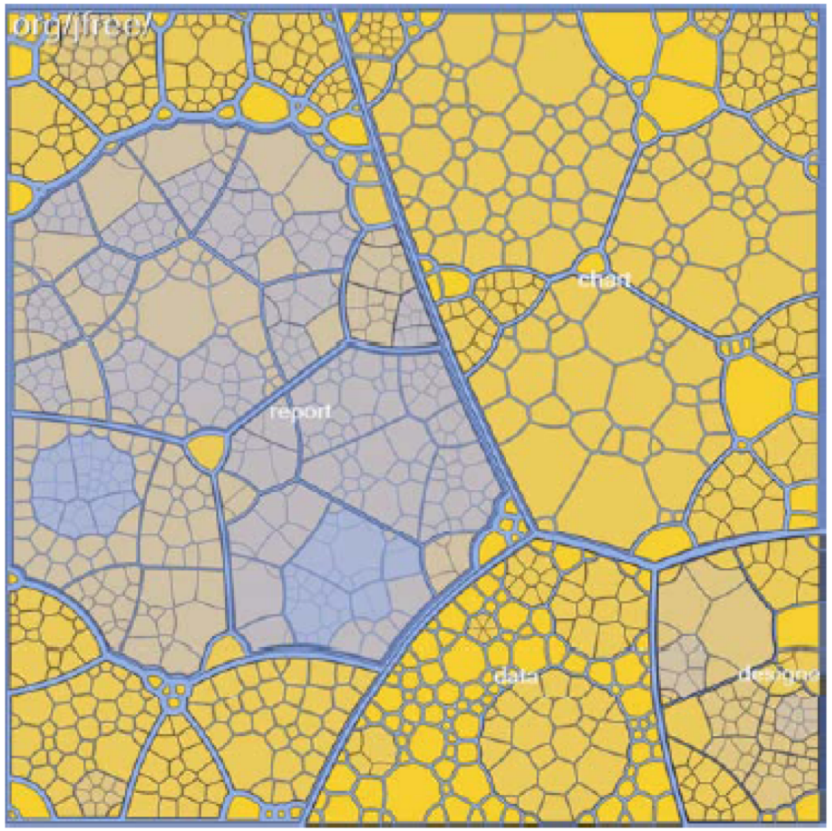
Christoph Csallner, Marcus Handte, Othmar Lehmann, John T. Stasko: FundExplorer: Supporting the Diversification of Mutual Fund Portfolios Using Context Treemaps. INFOVIS 2003: 203-208



- Christoph Csallner, Marcus Handte, Othmar Lehmann, John T. Stasko: FundExplorer: Supporting the Diversification of Mutual Fund Portfolios Using Context Treemaps. INFOVIS 2003: 203-208

Voronoi Treemaps





Michael Balzer, Oliver Deussen, and Claus Lewerentz. 2005. Voronoi treemaps for the visualization of software metrics. In Proceedings of the 2005 ACM symposium on Software visualization (SoftVis '05)

Definition of Voronoi Diagram

- Let P be a set of n distinct points (sites) in the plane.
- The Voronoi diagram of P is the subdivision of the plane into n cells, one for each site

Steven Fortune. A sweepline algorithm for Voronoi diagrams. Proceedings of the second annual symposium on Computational geometry. Yorktown Heights, New York, United States, pp.313–322. 1986

Definition of Voronoi Diagram

- A point q lies in the cell corresponding to a site $p_i \in P$ *iff*

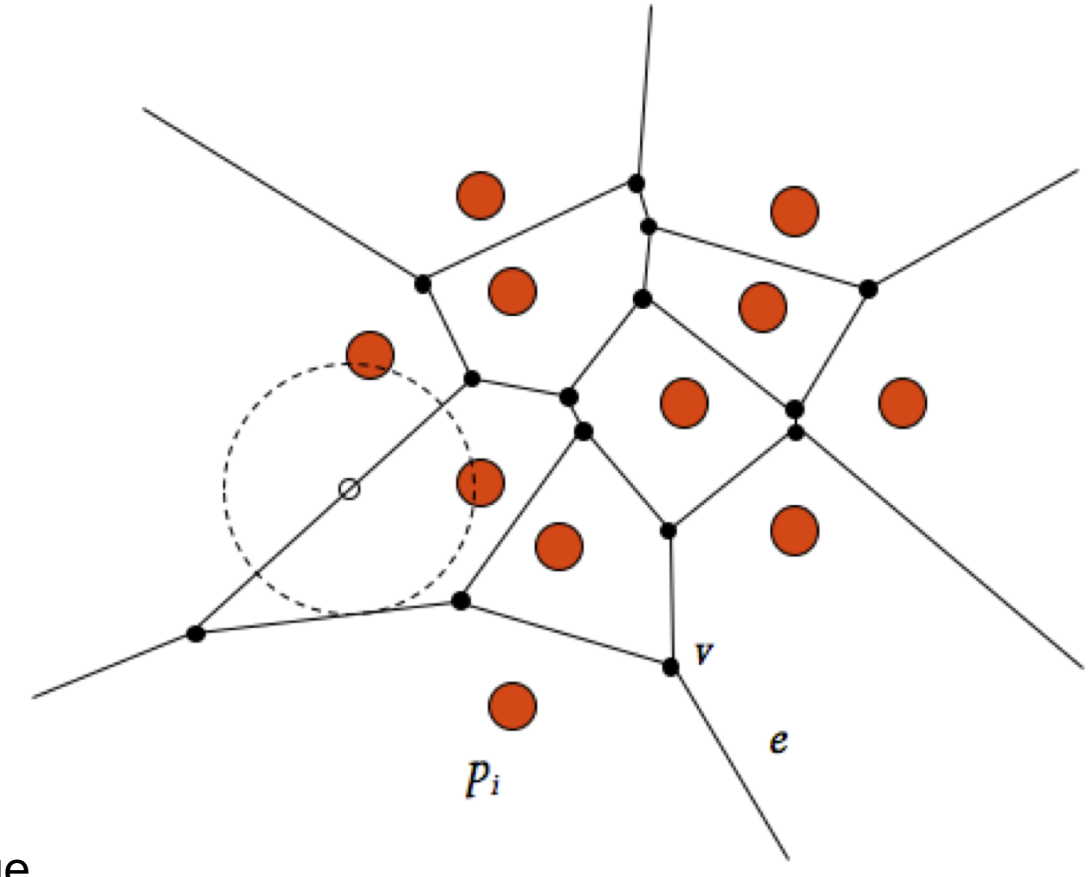
- Main Algorithm is Fortune's Algorithm

Euclidean_Distance(q, p_i) < Euclidean_distance(q, p_j), for each $p_i \in P$, $j \neq i$.

Steven Fortune. A sweepline algorithm for Voronoi diagrams. Proceedings of the second annual symposium on Computational geometry. Yorktown Heights, New York, United States, pp.313–322. 1986

Summary of Voronoi Properties

- A point q lies on a Voronoi edge between sites p_i and p_j *iff* the largest empty circle centered at q touches only p_i and p_j
  - A Voronoi edge is a subset of locus of points equidistant from p_i and p_j



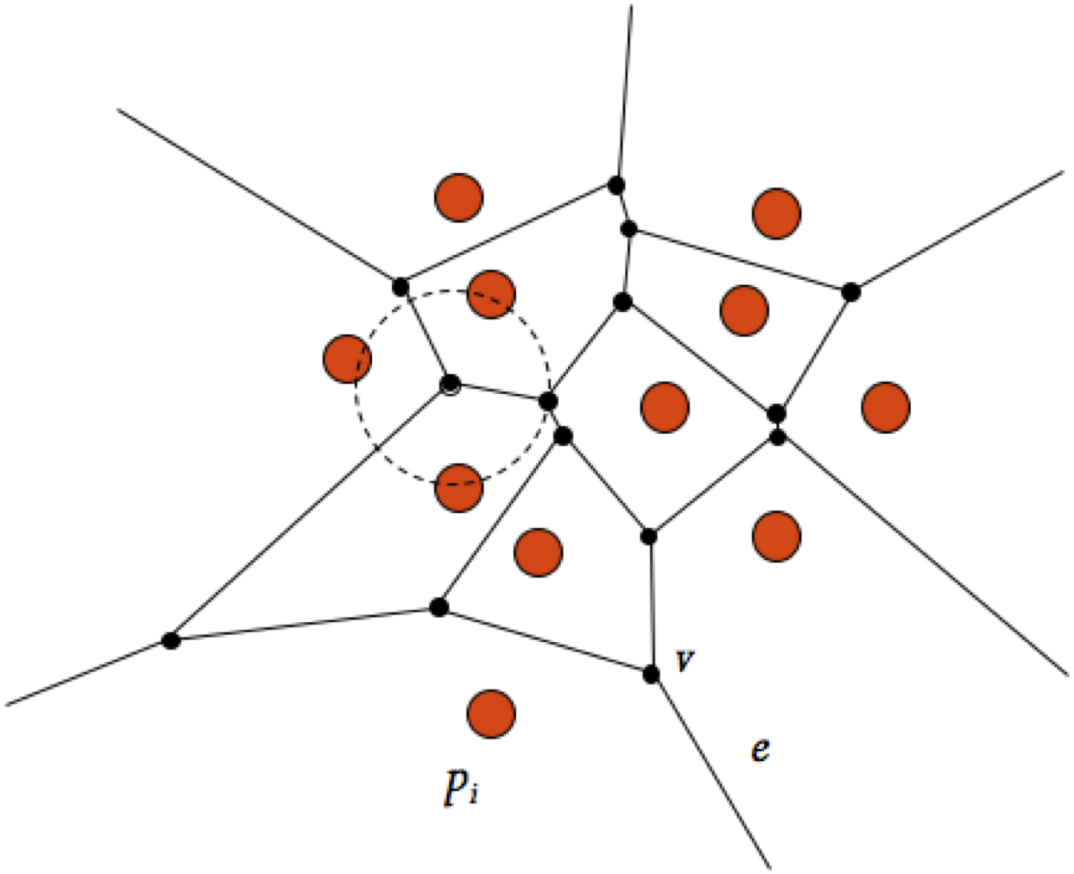
p_i : site points

e : Voronoi edge

v : Voronoi vertex

Summary of Voronoi Properties

- A point *q* is a vertex *iff* the largest empty circle centered at *q* touches at least 3 sites
  - A Voronoi vertex is an intersection of 3 more segments, each equidistant from a pair of sites



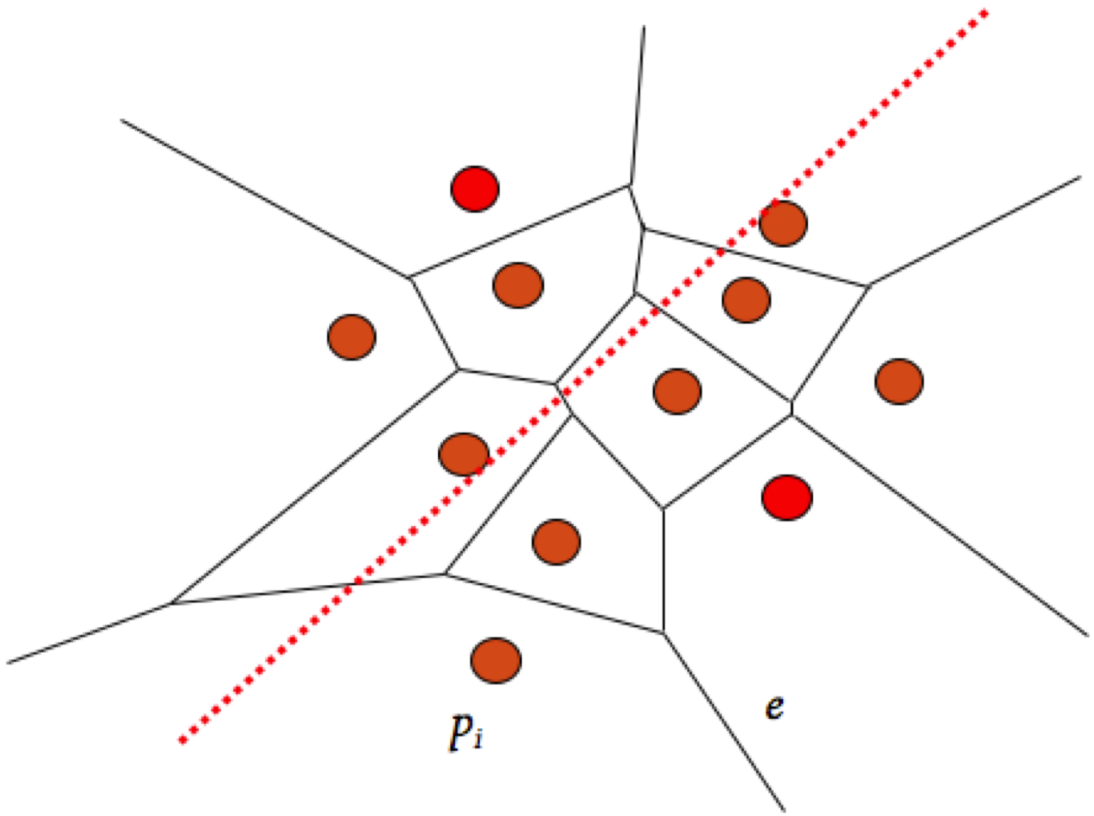
p_i : site points

e : Voronoi edge

v : Voronoi vertex

Summary of Voronoi Properties

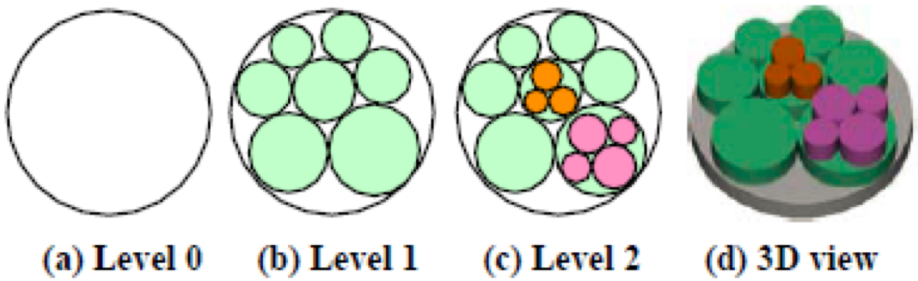
- Voronoi diagrams have linear complexity $\{|v|, |e| = O(n)\}$
  - Intuition: Not all bisectors are Voronoi edges!

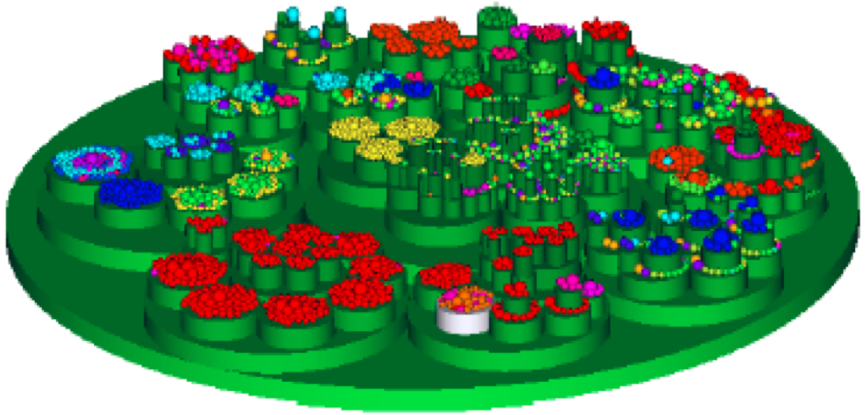


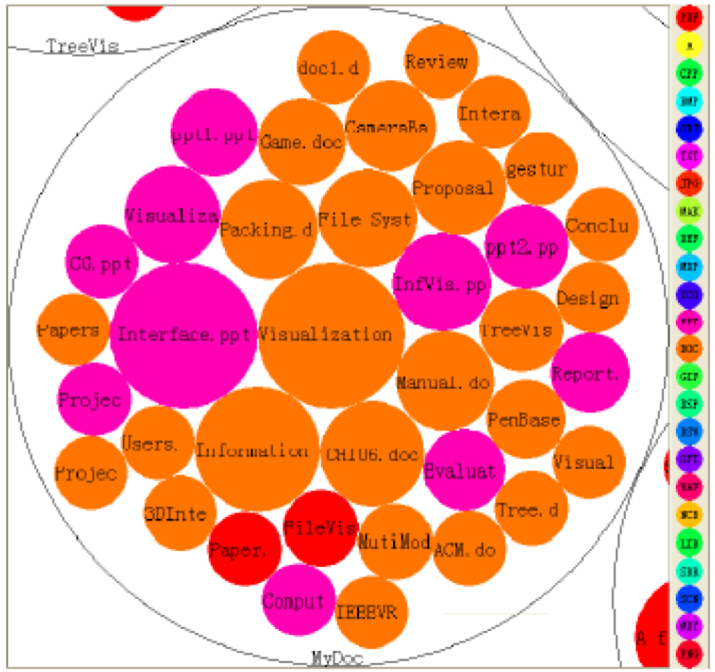
p_i : site points

e : Voronoi edge

Circle Packing







Weixin Wang, Hui Wang, Guozhong Dai, and Hongan Wang. 2006. Visualization of large hierarchical data by circle packing. In Proceedings of the SIGCHI conference on Human Factors in computing systems (CHI '06), pg. 517-520.