Ordered and Quantum Treemaps: Making effective use of 2D space to display hierarchies

B. Bederson, B. Shneiderman, M. Wattenberg

Presented by: Neeti Ogale and Rachael Bradley
Treemaps

- Space filling technique to visualize hierarchical data
Issues of concern for treemaps

- Minimize the aspect ratio of the rectangles
- Maintain Order in the data
- Accommodate changes in the data
Original algorithm for Treemaps

Slice and Dice Treemap:
(Shneiderman, B. 1992).

- Preserves order
- Changes to the data cause continuous changes in the output
- Aspect ratio of rectangles can be large – so can give long skinny rectangles

Gray shades: Order

Slice and Dice
Treemaps by the other two algorithms

**Cluster Treemap** (Wattenberg, M. 99’)

**Squarified Treemap** (Wijk et al. 2000)

- Overall aspect ratio is less – no skinny long rectangles
- Changes to the data cause dramatic discontinuous changes
- Does not maintain order
Demo
Ordered treemap algorithm

- Inputs: Rectangle R and ordered list of items having specific areas
- Recursive algorithm. At each step:
- Select a pivot item (P)
- Divide the list into 3 parts L1, L2, and L3 and R into four rectangles, \( R_1, R_P, R_2, \) and \( R_3 \)
- If the number of items is \( \leq 4 \), lay them out in either a pivot, quad, or snake layout, Stop

If width \( \geq \) height
Selection of Pivot

pivot-by-size

pivot-by-split-size

pivot-by-middle
Dividing the list $L$

Divide the items in the list, other than $P$, into three lists, $L_1$, $L_2$, and $L_3$, such that $L_1$ consist of items whose index is less than $P$ and $L_2$ have items having index less than those in $L_3$, and the aspect ratio of $RP$ is as close to 1 as possible.
Stopping condition

If the number of items is $\leq 4$, lay them either in a **pivot**, **quad**, or **snake** layout.

- **Pivot**
- **Quad**
- **Snake**

Pick the layout whose average aspect ratio is closest to 1.
Strip treemap algorithm

- Rectangles are processed in order and inserted in the current strip if doing so does not increase the average aspect ratio of all the rectangles in the strip ($O(\sqrt{n})$ time on average)
Quantum Treemaps

- To display items of fixed size – Images.
Quantum tree maps algorithm

- Rectangles are quantized when they are generated in the following manner –
  - Dimensions are integer multiple of the given element size.
  - Total area is at least equal to the total area of all the elements to be fitted in it.
- Grids of elements are aligned with rows and columns of elements running across the entire series of rectangles.
Experimental Results

100 Monte Carlo trials with 100 steps - simulate continuously changing data.

- A tradeoff between low aspect ratios and smooth updates.
- The slice-and-dice method - high aspect ratios, but small changes with data updates.
- Squarified and Cluster treemaps - low aspect ratios and large changes in layouts.
- The ordered and strip treemaps fall in the middle of the spectrum. Aspect ratios better than slice-and-dice method, update more smoothly than cluster or squarified treemaps.
Readability study

Readability: ease of visually scanning a layout

Static stock market data.
Quantum Treemap Analysis: Aspect Ratios

Average Aspect Ratios

- Basic Ordered Treemap
- Quantum Basic Ordered Treemap
- Strip Treemap
- Quantum Strip Treemap

Maximum # elements per rectangle
Quantum Treemap Analysis: Space

Wasted Space

- Basic Ordered Treemap
- Quantum Basic Ordered Treemap
- Strip Treemap
- Quantum Strip Treemap

Maximum # elements per rectangle

- 10
- 20
- 50
- 100
- 1000

Wasted Space

- 0%
- 5%
- 10%
- 15%
- 20%
- 25%
- 30%
- 35%
- 40%
- 45%
- 50%
Future Work

- Improve the overall aspect ratio and optimize the algorithms used by cluster treemaps and squarified treemaps to improve stability under dynamic updates.

- Mixing different algorithms to combine their strengths

- Look for a mathematical theorem that makes the tradeoff between the aspect ratio and smoothness of layout changes precise.

- Accommodation of trees with millions of nodes

- Show more than two attribute values for each leaf node