Mateusz Piwowarski - ERASMUS student
Student ID: 1002440068
E-mail: mateusz.piwowarski@student.um.si
University of Maribor, FERI
30.05.2020 Maribor

Data visualization

ALGORITHMS FOR BIGDATA ANALYSIS

Today in the big data era when companies are overfilled with data of various types, searching information to understand what is important and what is not getting more difficult. Visualizations accelerate the analysis and make it easier. Visualizations offer the ability to quickly see what is important. Most people respond much better to visualizations than to text. Good data visualization is necessary for data analysis. When visualization is clear it helps to make decisions and understand patterns, relationships and emerging trends. In most cases it is much easier to understand and read data from a graph than from raw data in a table, spreadsheet or a file. In most cases, no specialized skills are required to interpret and understand what is presented in the graphic. Now we actually come to the question what is data visualization and what is it about?

Visualization is a process that transforms (abstract) data into an interactive graphical representation for the purpose of exploration, confirmation, or representation[1]. Nowadays we can visualize data in many ways. Some techniques display data in more continuous way (e.g. heatmap), while others display it in more discrete way (e.g. Venn diagram). The most important trade-off during choosing the visualization technique is that the visualization must be informative but readable. Visualization technique depends on the type, dimensionality, complexity of the data.

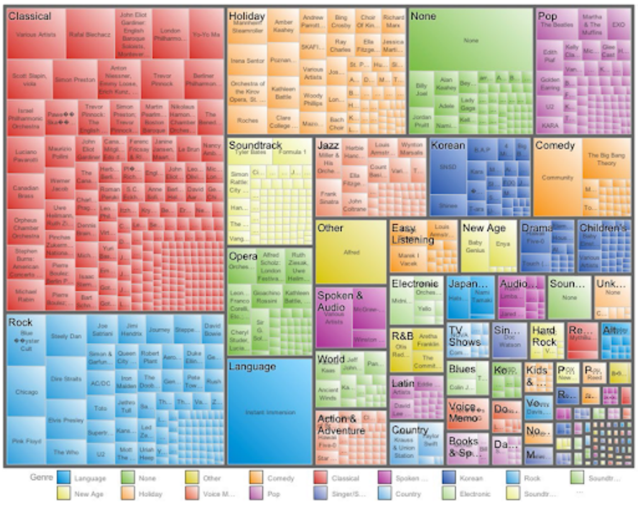
The visualization process consists of 5 steps:

- Acquire the data from various sources
- Parse collected data into structured format
- Filter data (dispose of unimportant data)
- Refinement, patterns usage
- Data Visualization

[1] definition of visualization taken from Human-Computer Interaction lectures (doc. dr. Niko Lukač)

In this article I will focus on data visualization from the point of big data. I will describe techniques which are useful for dealing with problems of volume, variety and velocity of the data.

Tree map



(Source: https://lh3.googleusercontent.com/proxy/AUddmQquk4ED4Ph4U2jYDAgJczngFjA9EMgxCZ4tG7Y3_jdHvRtOU5u_Q1muKDqj-GxcqBbaI9nY2n6D_otN9v0bt2LzO3K6H7A)

Tree maps are used to express a variety of nested and hierarchical data and data structures. Tree maps are an alternative way of visualizing the hierarchical structure of a Tree diagram. Tree maps consist of nested rectangles. The size of the rectangles plays an important role in visualizing the tree map because it shows how big that data element is. We use tree maps because they guarantee easy identification of patterns and efficient usage of space. Data used in tree map is organized as branches and sub-branches.

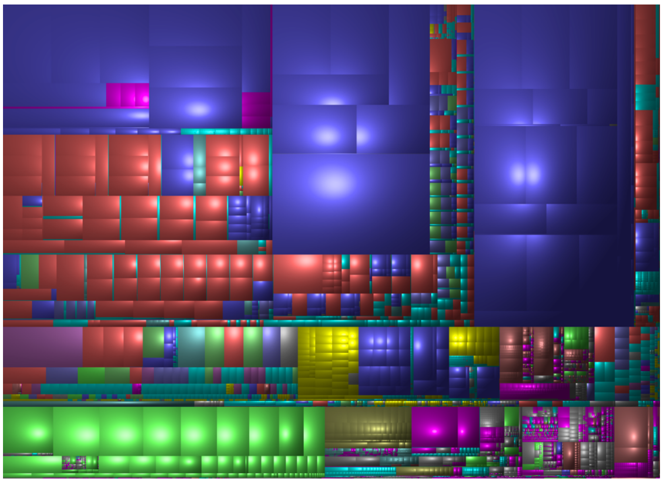
Interaction with the map makes searching and browsing data easier. The interaction is based on tree levels: user can zoom smaller areas of hierarchy (if this feature is implemented).

Good idea is to use tree map technique if you need query for a large set of data or find out patterns in large data set. For user intuitiveness radio buttons, buttons and sliders are most commonly used controls for dynamic queries implementation. By those controls user can jump from one rectangle to another to find needed information without typing anything.

Tree maps have also limitations. There is no option for sorting: the nodes are automatically ordered by area within the parent node. It is complicated to display tree map with large number of data points on a single level. If in tree map size of the rectangle represents the quantitative value it's impossible to print negative value. Considering the main tree map advantage (identifying objects by size) it is poor choice to pick tree map for data sets with similar data size.

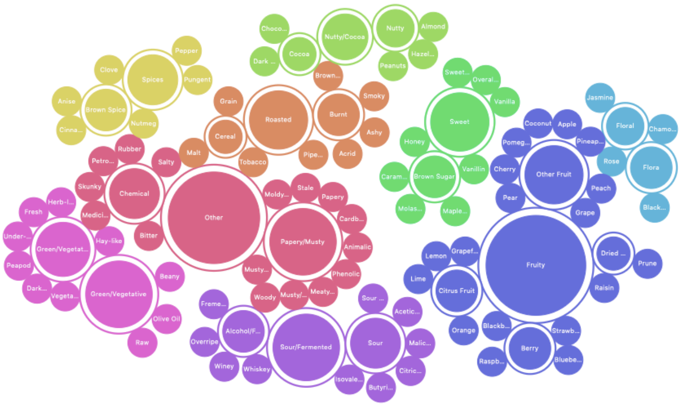
Comparisons in tree maps are difficult because 2 different nodes may be located at different levels in different categories. Sometimes it may be impossible to display two strongly nested nodes at the same time.

If we encounter a huge amount of data sets in our tree map, the user may have trouble noticing the smallest rectangles. To facilitate the analysis and eliminate the above-mentioned problem we can use **cushion tree map**. In this technique we use slight gradient inside every rectangle. The gradient goes from the edges to the center. It gives the impression that rectangles are raised in the center and tapering off to the edges. It helps to notice and identify adjacent rectangles. The effect is shown below:



(Source: https://media.nngroup.com/media/editor/2019/09/16/cushiontreemap.png)

Circle packing



(Source: https://www.amcharts.com/demos/packed-circle-chart/)

Circle packing technique is another method to visualize huge amount of data. Circle packing like tree map technique is intended mainly for hierarchically and structured data. Data in circle packing is packed and displayed in the form of circles. Brother nodes at the same level are represented by tangent circles. Size of the circle plays an important role in circle packing like size of rectangles in tree map technique: it may represent value of specific property or amount/size of elements in specific group.

In specific circles we can pack another circles that represent nodes from one level below. This is the way we can visualize hierarchy in circle packing method.

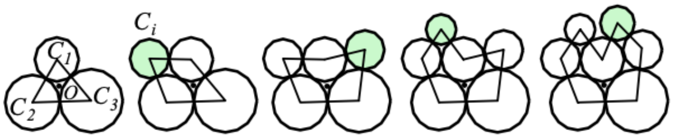
So what are the differences between circle packing and tree map? In which cases should we choose circle packing instead of tree map?

In tree maps it is harder to see the hierarchical structure: „Most of the space is used for the display of leaf nodes and the branches are encoded implicitly”[2]. Usage of circles instead of rectangles allows to show in better way structural relationships and groupings. On the other hand circle packing method is not space efficient, in that case tree map is better choice.

How does circle packing work?

For every circle we save information like: index of the circle, center of the circle (position) and radius (node size). At the beginning of we put 3 circles tangent to each other around specific point in the visualization space. Link of the circles centers is **front-chain**. To add every next circle we have to put it externally tangent to two selected circles on the front-chain. We have to remember to update front-chain after every insertion of new circle.

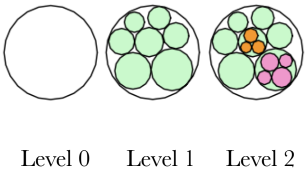
Front-chain is saved in background as a list which saves information about relationships between nodes (which circle borders with which) e.g. {C1<->C2<->C3<->C1}. On the graphic below you can see example how inserting new circles into structure looks like:



(Source: Visualization of Large Hierarchical Data by Circle Packing (W.Wang, Hui Wang, G.Dai, Hongan Wang))

After describing idea how to pack brother nodes we have to meet with idea how nesting of the circles work:

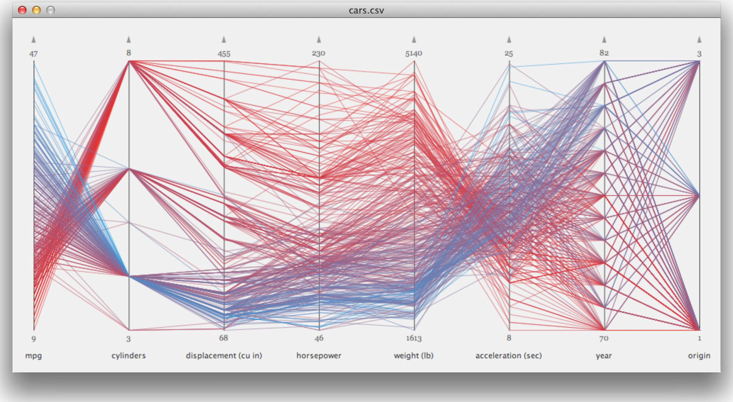
For nesting circles we use recursive function. In the first step we calculate size of every circle by sum of every nested circle inside specific circle. Then we are creating root node and pack all its nodes inside of it. Nested circles are on one level higher. We pack child nodes with the idea described above (how to pack brother nodes). We repeat packing child nodes until we reach the most nested circle (without children). On the graphic below you can see example how nesting circles looks like:



(Source: Visualization of Large Hierarchical Data by Circle Packing (W.Wang, Hui Wang, G.Dai, Hongan Wang))

[2] Visualization of Large Hierarchical Data by Circle Packing (W.Wang, Hui Wang, G.Dai, Hongan Wang)

Parallel Coordinates



(Source: https://i.stack.imgur.com/rrEMX.jpg)

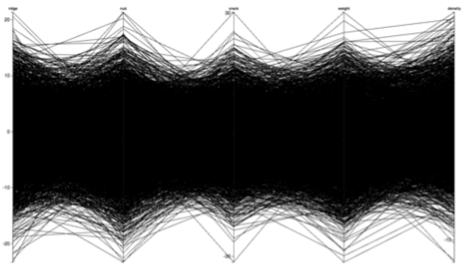
Parallel coordinates is visualization technique to visualize high-dimensional data. This method is not recommended for categorical data. Main feature of parallel coordinates is showing many variables in the same time. Every point in our data set is represented on parallel coordinates graph as connected lines passing through variable axes showing variable values.

If we use parallel coordinates for big data, commonly known actions before final visualization are: dimension reordering, dimension ordering, spacing, filtering. Those actions lead to present data in a form that makes them the most clear and easiest to analyze. Clustering is recommended action to deal with excessive number of lines. It leads to minimize the number of lines to be shown by grouping the lines and centering them.

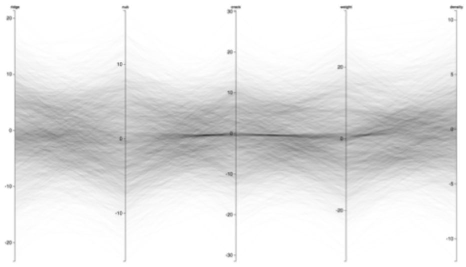
Overplotting is common problem in visualization big data by parallel coordinates. In the graphic below (1) we can see overplotting example when we have too many lines on the screen overplotting with the same color and the final graph is really hard to be interpreted.

One of the solution for overplotting is to visualize density of lines instead of individual lines. You can see applied this method on the graphic below (2).

1)

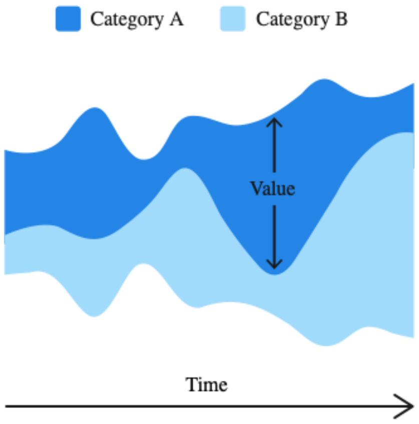


2)



(Source: Big Data Visual Analytics with Parallel Coordinates (J.Heinrich, B.Broeksema))

Stream Graph



(Source: https://datavizcatalogue.com/methods/images/anatomy/SVG/stream_graph.svg)

Stream graph is visualization technique to display changes in data over time. Stream graph uses categories (most likely using different colors). Shapes of categories (layers) used in stream graph look like river stream. Value in each category is represented by the size of the stream. High-volume datasets can be visualized by this method. Use of stream graph makes easier recognizing/discovering trends and patterns in specific time in specific categories.

Layers must be sorted before displaying them. Layers with the biggest and the most frequently appearing differences in size (value) should be put outside. Layers with the smallest differences in value (the most consistent) should be put closer to the center.

After addition of new data, layers must be reordered what leads to creation of the new graph. It is not the problem in performance case but for user it may be confusing if the graph has changed significantly.

Contrary to appearances it is not always good idea to use stream graph for visualization of streaming data. Stream graph works best if: there are not many layers, burst in layers are not to big and not too often occurring. Problem with graph readability for problem mentioned before occurs because layers with smaller values free space for biggest layers.

To avoid user confusion it is recommended to use stream graph if data doesn't have to be updated frequently.

Conclusions

This article shows commonly used and popular visualization techniques for big data. I tried to describe 4 techniques and their usage, purpose and how to deal with most common problems occurring in visualization by specific method.

Before choice of the data visualization method you should have information about your data. Every technique is intended for different type of data. It depends if we have categorical data, what is the structure, how many data sets or categories we have.

Visualization of big data may be problematic and many scientists still work to discover/create new tools, techniques and they solve problems related to big data visualization. Big data is quite new topic and many new solutions in this topic can be discovered at any moment.

Sources

1. Human-Computer Interaction lectures [Introduction to 2D graphics and visualization] (doc. dr. Niko Lukač)
2. Big Data Analytics for Data Visualization: Review of Techniques (G.Chawla, S.Bamal, R.Khatana)
3. Tree-map: A visualization tool for large data (M. Jadeja, K.Shah)
4. Visualization of Large Hierarchical Data by Circle Packing (W.Wang, Hui Wang, G.Dai, Hongan Wang)
5. Visual Clustering in Parallel Coordinates (H. Zhou, X. Yuan, H. Qu, W. Cui, B. Chen)
6. Big Data Visual Analytics with Parallel Coordinates (J.Heinrich, B.Broeksema)
7. Visualization of Streaming Data: Observing Change and Context in Information Visualization Techniques (M. Krstajić, D. A. Keim)
8. https://www.oracle.com/business-analytics/what-is-data-visualization/
9. https://www.fusioncharts.com/resources/chart-primers/treemap-chart
10. https://www.nngroup.com/articles/treemaps/
11. https://datavizproject.com/data-type/packed-circle-chart/
12. https://datavizcatalogue.com/methods/stream_graph.html