GRAPH VISUALIZATION

Hans-Jörg Schulz

1. INTRODUCTION

2. VISUAL ENCODING

3. BASIC CHART TYPES

Basics

4. INTERACTION

5. VISUALIZATION DESIGN

6. DATA PREPROCESSING

7. RECAP 1st Half

Visualization Building Blocks & Processes

8. MULTIVARIATE DATAVIS

9. TEMPORAL DATAVIS

10. GEOSPATIAL DATAVIS

11. GRAPH DATAVIS

Visualization Techniques

12. 3D DATAVIS

13. VISUAL ANALYTICS

14. RECAP 2nd Half

Visualization Applications

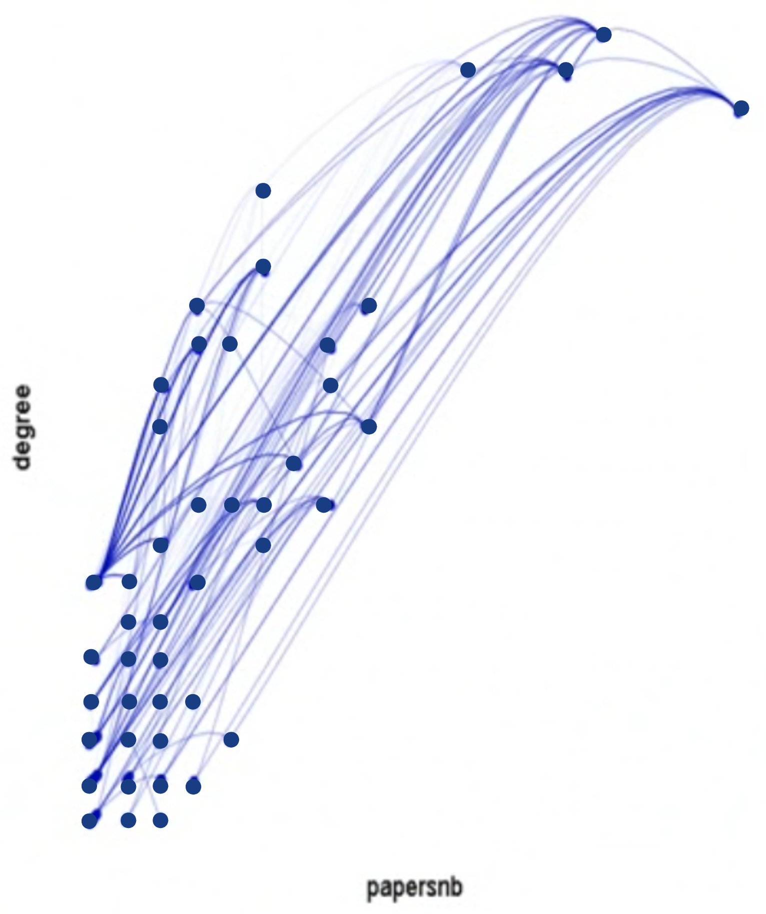
WHY TALK ABOUT GRAPH VISUALIZATION?

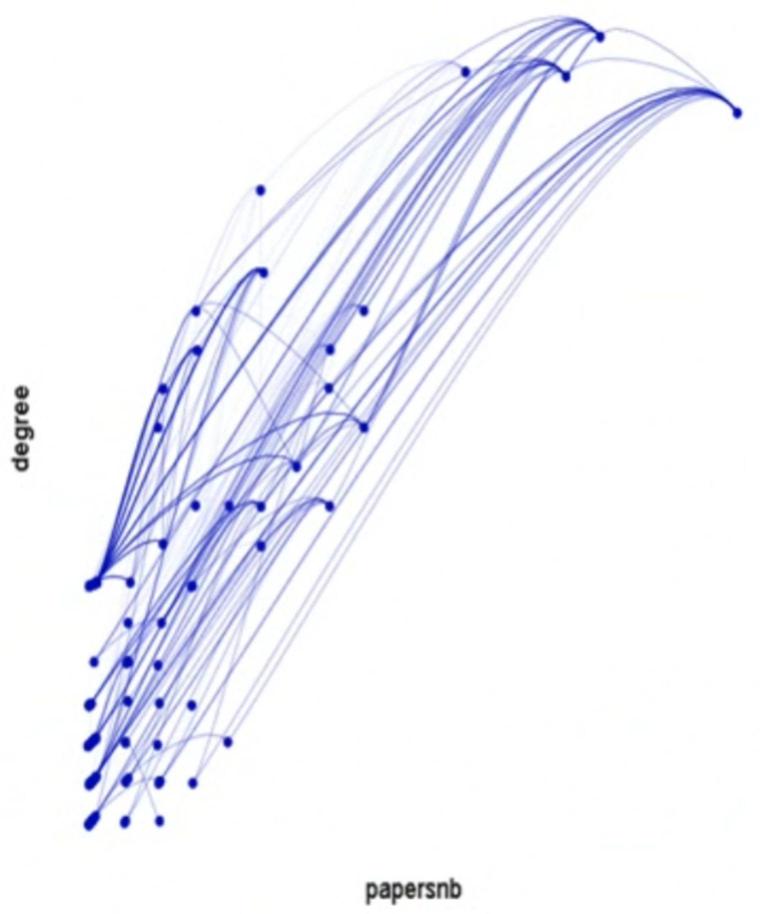
Graphs / networks put two data items into relation to each other.

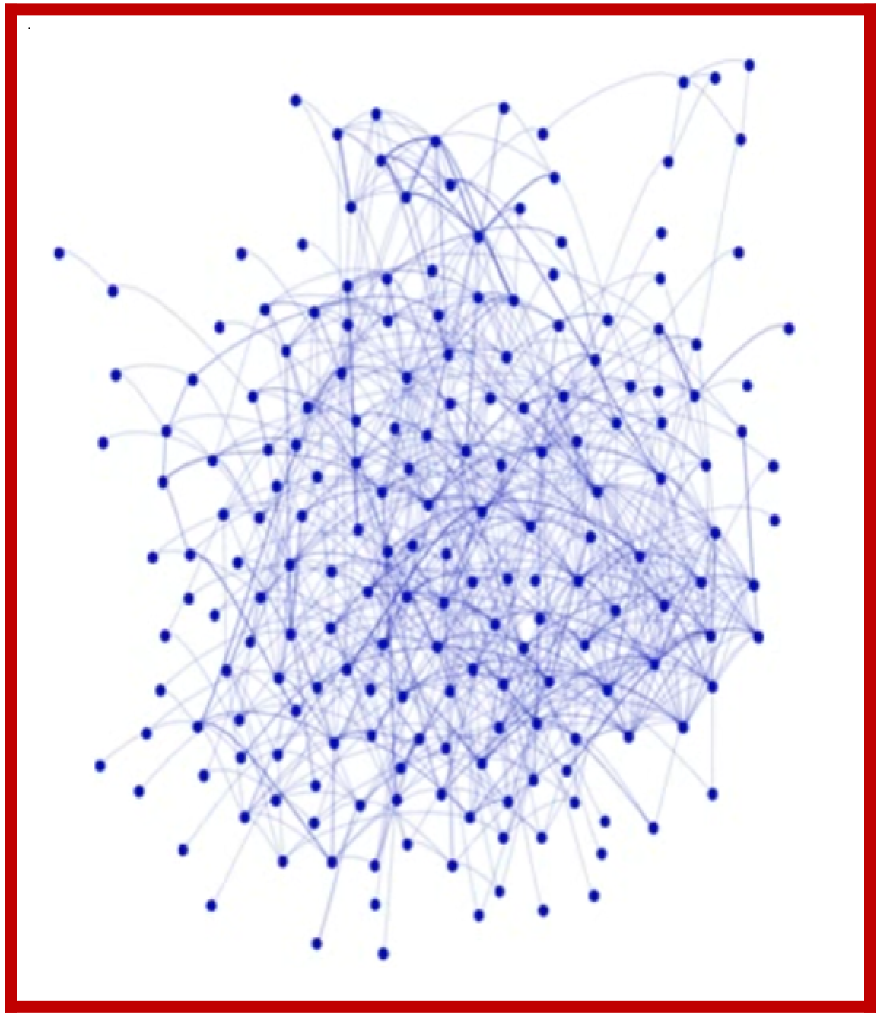
- Social networks
- Computer networks
- Call graphs of a software
- Traffic networks
- ...

If that relation is hierarchical, we also speak of a tree structure.

DATA VALUES VS. DATA RELATION







FUNDAMENTAL TERMS

Data item => **Node**

Data relation => **Edge**

# of Data items related to a given data item X => **Degree of X**

Numerical attributes of nodes and edges => **Weights**

Graph w/o cycles => **Tree**

Dedicated node of a tree => **Root**

Data item A (closer to the root) having an edge to B (further from the root)

=> A is the **Parent,** B is the **Child**

Nodes having the same parent => **Siblings**

Nodes having no children => **Leaves**

Average # of children in a tree => **Branching factor**

OVERVIEW

1. Node Link Diagrams
- Force-Directed Layout Mechanism & Extensions

2. Matrix Visualizations
- Matrices and Hierarchical Matrices
- Ordering Matrices

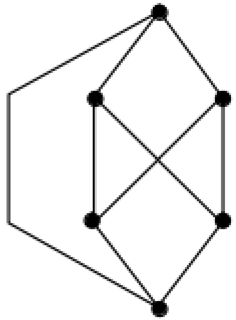
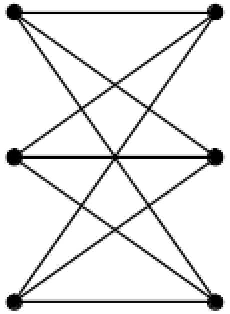
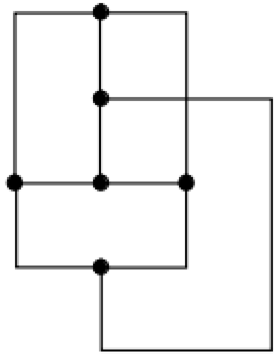
3. Implicit Tree Visualization
- TreeMap, Icicle Plot, Sunburst

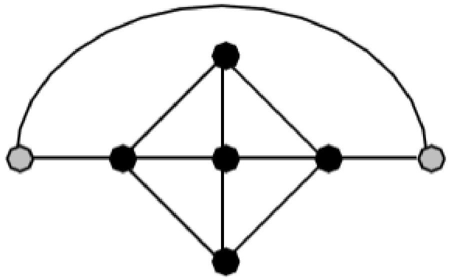
4. Visualization of PO-Sets
- The Sugiyama Layout Framework

NODE-LINK DIAGRAMS

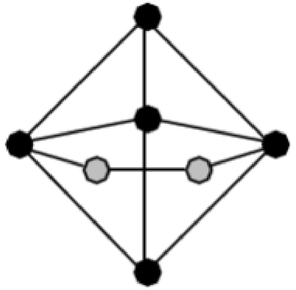
WHAT MAKES A GOOD NODE-LINK-DIAG.?

- Draw edges straight, polyline, orthogonally,...
- Minimize Edge Crossings
- Minimize Edge Bends
- Minimize Aspect Ratio
- Minimize Drawing Area
- Maximize Symmetries
- Maintain Uniform Edge Length
- ...

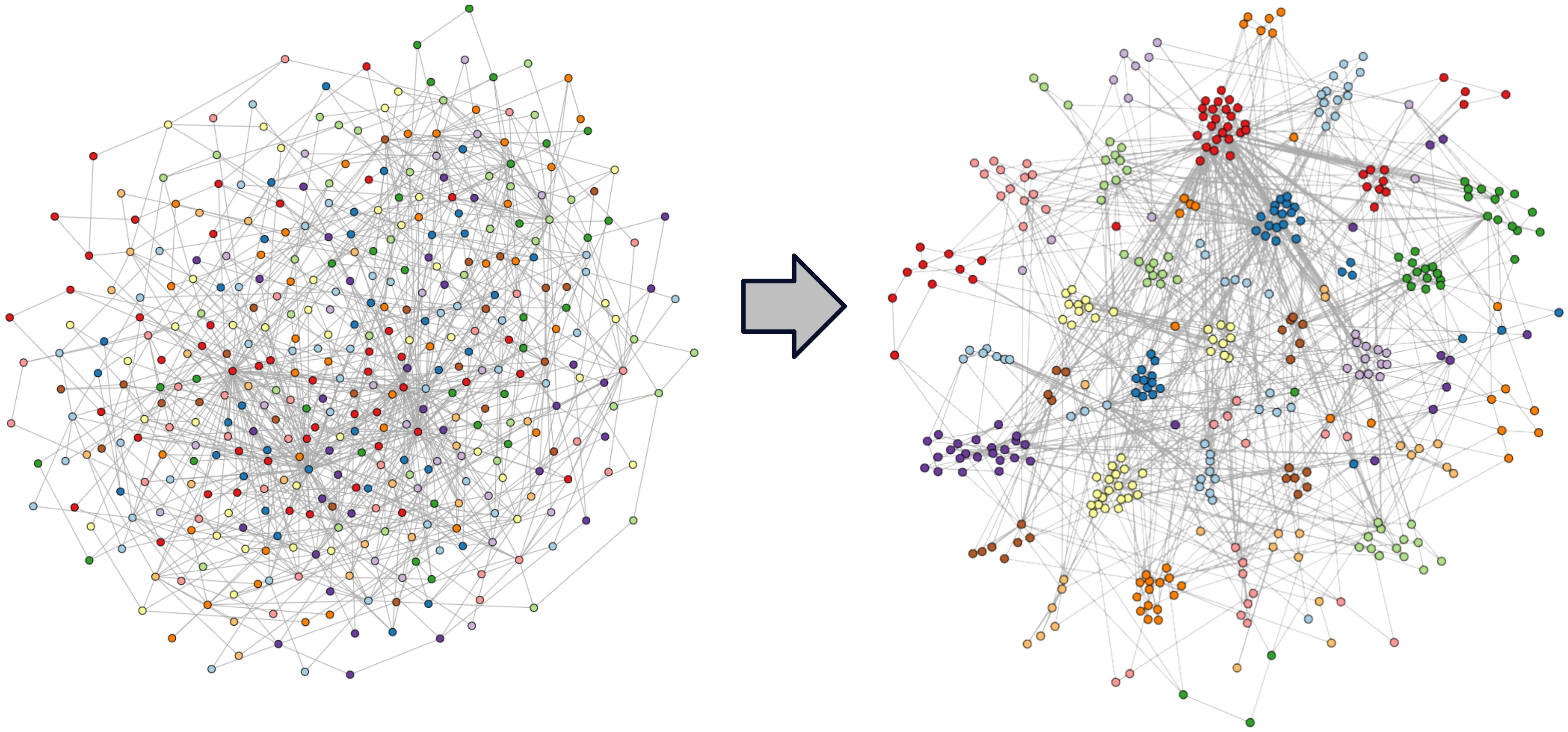


Minimum number of edge crossings

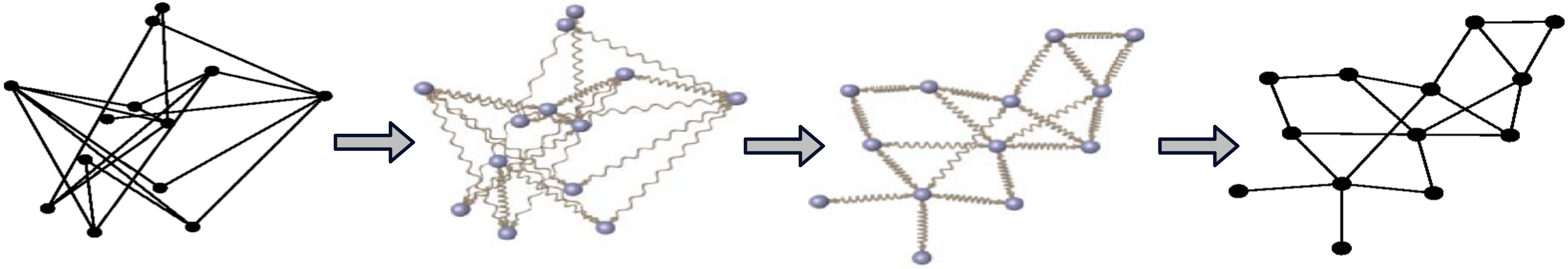


Uniform edge length

NODE LINK LAYOUTS



FORCE-DIRECTED GRAPH LAYOUT



picture source: [Brandes 2001]

For each edge between nodes U and V:
attractive force (spring) → Hooke's Law

For each pair of nodes U and V:
repulsive force (expander) → Coulomb's Law

A FORCE-DIRECTED LAYOUT ALGORITHM

1. Initialize Node Positions (e.g., randomly or via pre-layout)
2. For a number of iterations do
    3. For all node pairs (u,v) do

repulsion constant

“charges” of u and v

distance between u and v

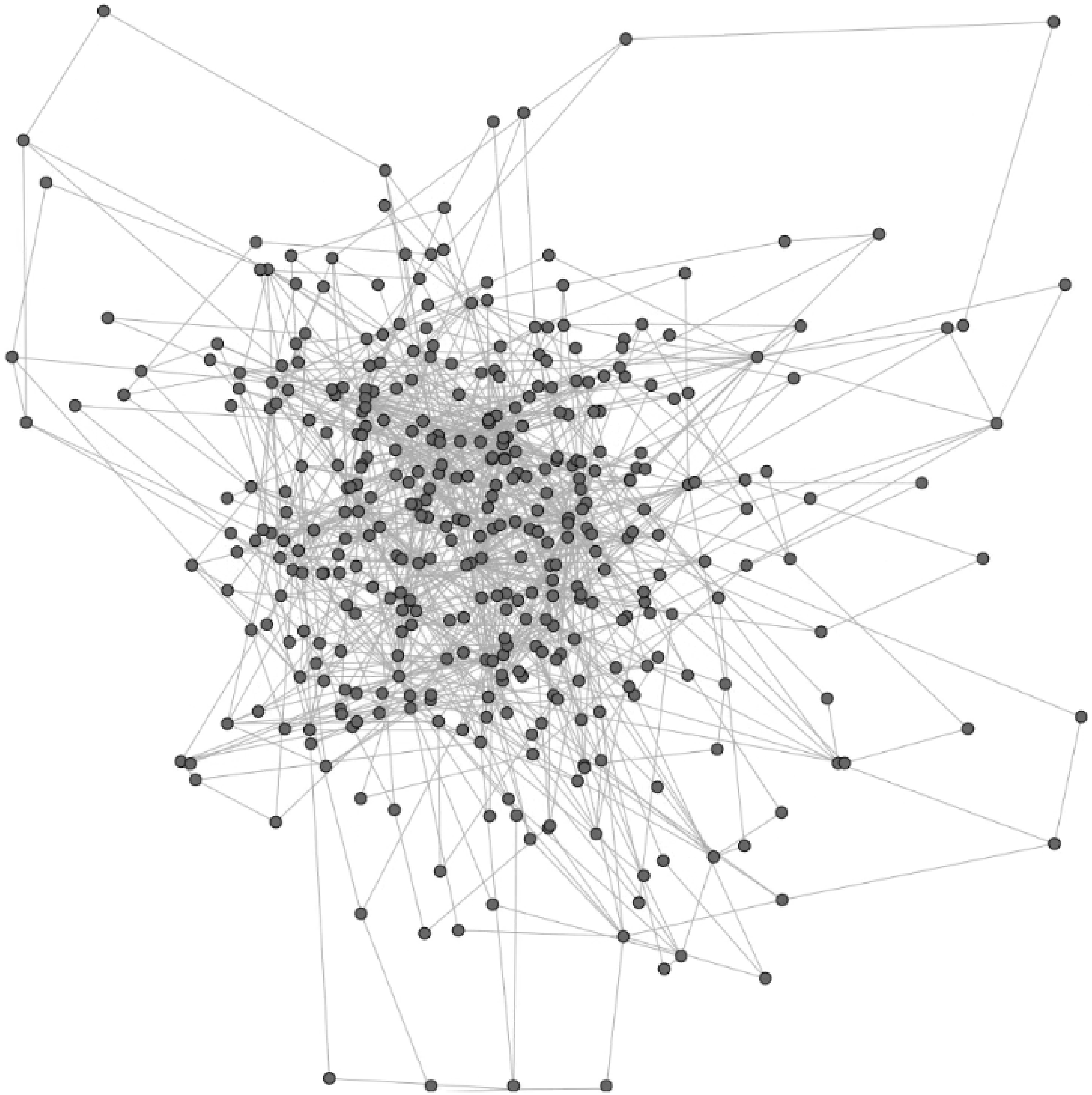
        4. Computer Repulsive Force according to Coulomb’s Law: $F_C=k_r(Q_1*Q_2/r^2)$
        5. Move u and v apart from each other by a distance F_C

    6. For all edges (u,v) do

spring constant

difference between the distance (u,v) and the length of the spring

        7. Compute Attracting Force according to Hooke’s Law: $F_H=-k_a * x$
        8. Move u and v closer to each other by F_H



…ADD SIMULATED ANNEALING TO THE MIX

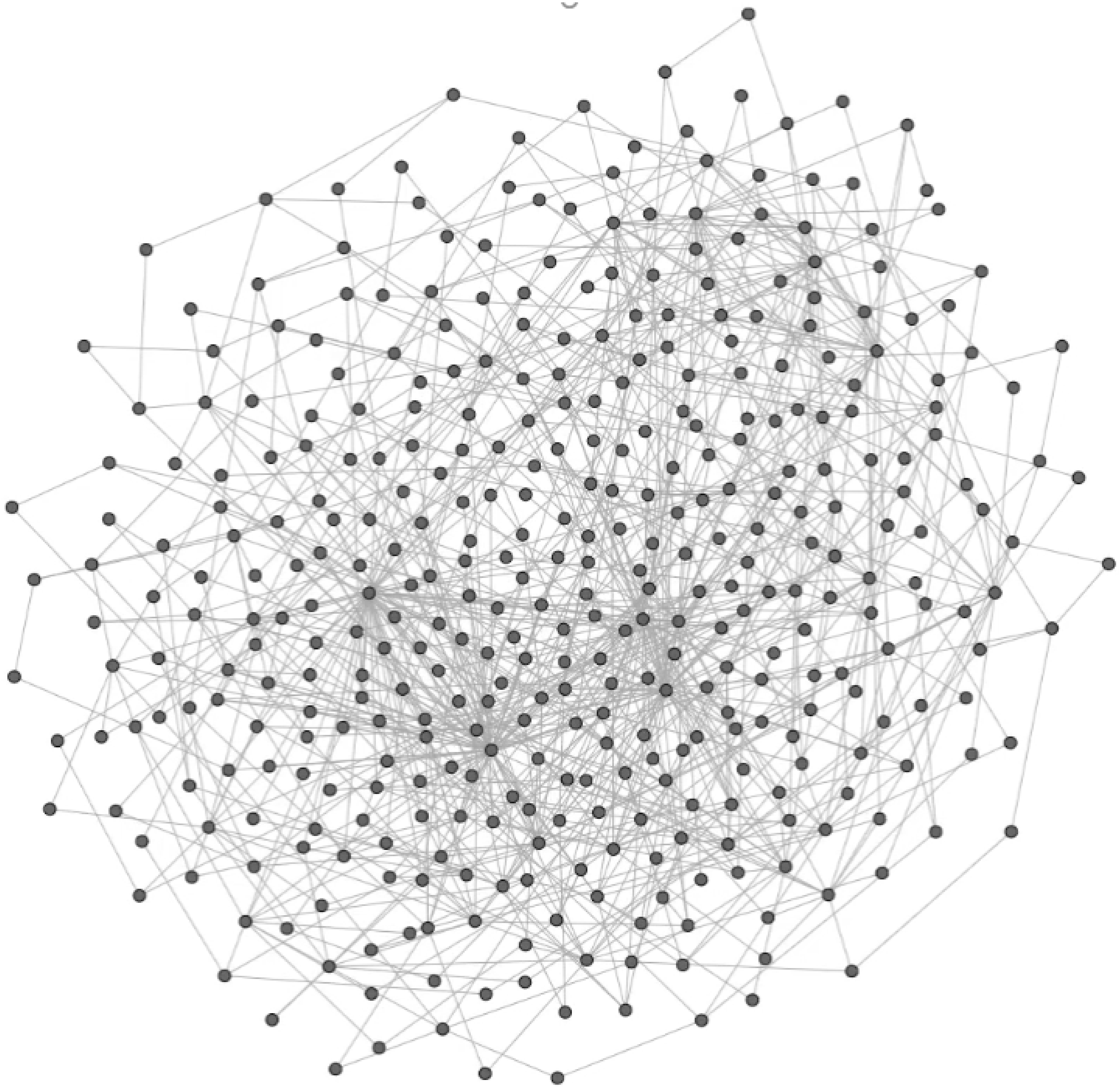
1. Initialize Node Positions (e.g., randomly or via pre-layout)
   Initialize velocity = 1
2. **While velocity > 0.0001 do**

   3. For all node pairs (u,v) do
      4. Computer Repulsive Force according to Coulomb's Law: $F_C=k_r(Q_1 * Q_2/r^2)$
      5. Move u and v apart from each other by a distance F_C *** velocity**

   6. For all edges (u,v) do
      7. Compute Attracting Force according to Hooke's Law: $F_H=-k_a * x$
      8. Move u and v closer to each other by F_H *** velocity**

   9. Velocity = velocity * 0.95

cooling rate



...ADD MORE FORCES INTO THE MIX

Given a pre-clustered graph =>

Pull nodes of the same cluster closer together:

For **all clusters (c)** in the graph do

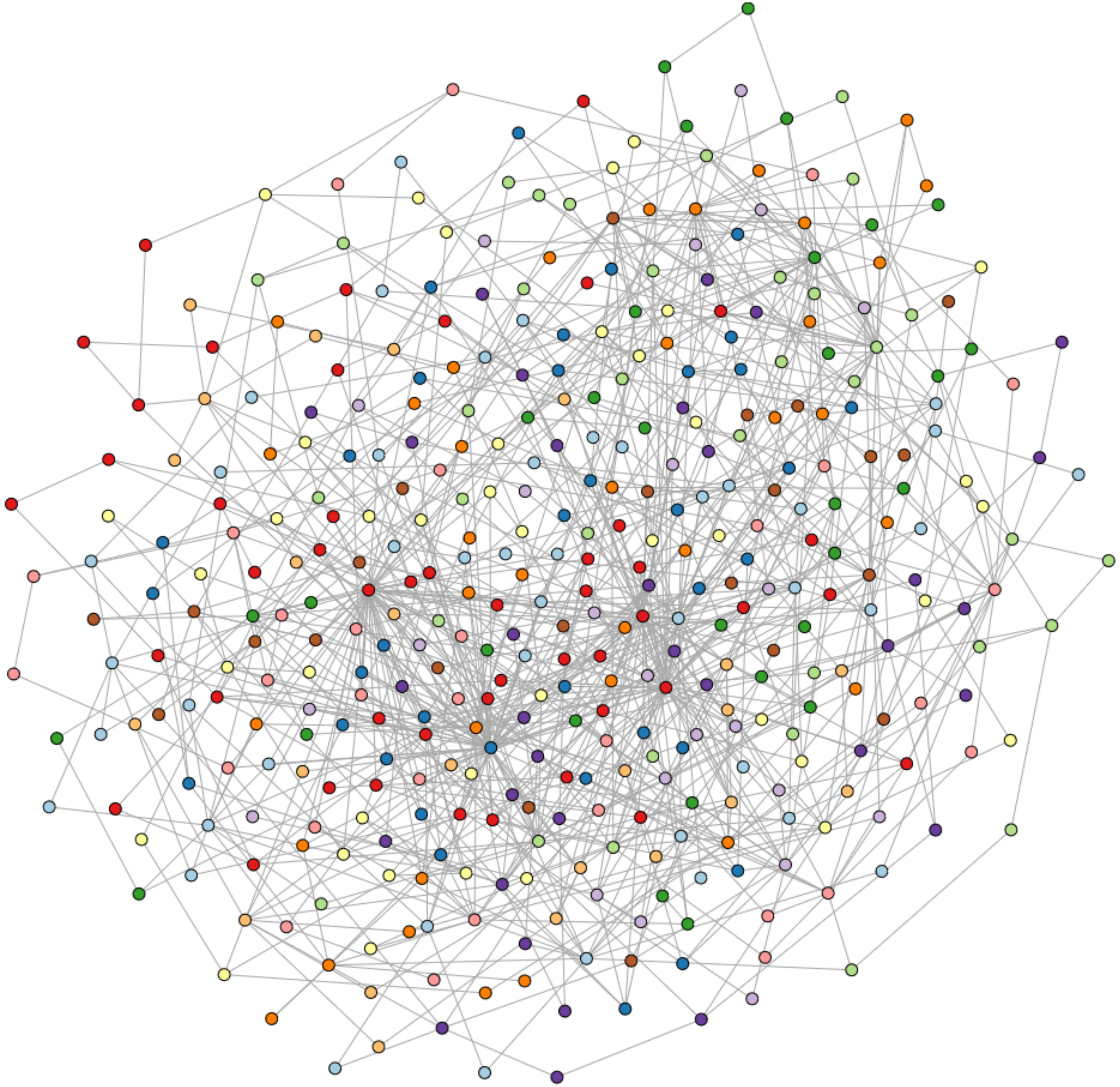
For **all nodes u** in c do

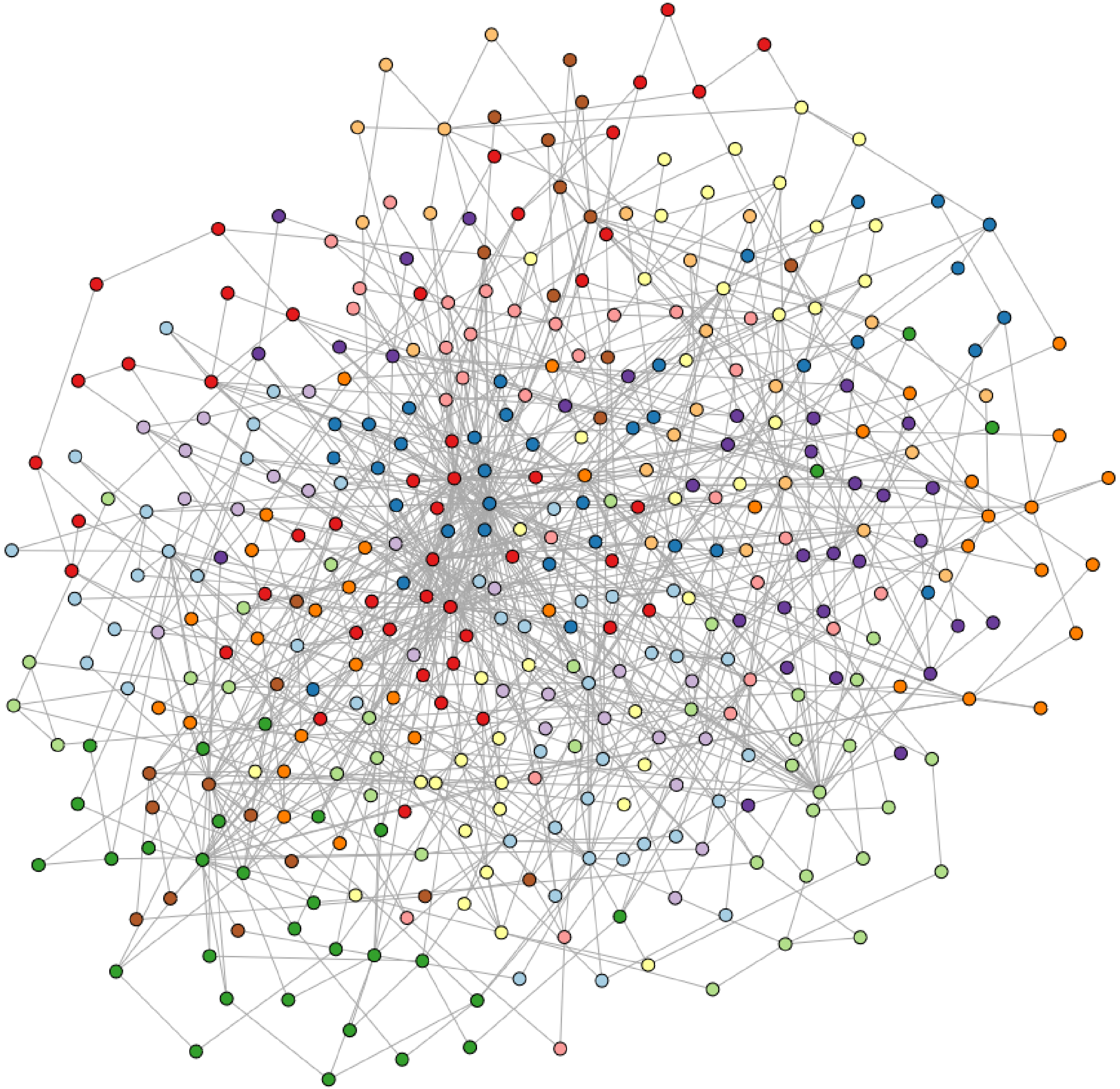
For **all nodes v** in c do

If u!=v do

Compute additional attracting force F_C=-k_c * x

Move u and v even closer to each other by F_C * velocity





...AND ADJUST OTHER FORCES

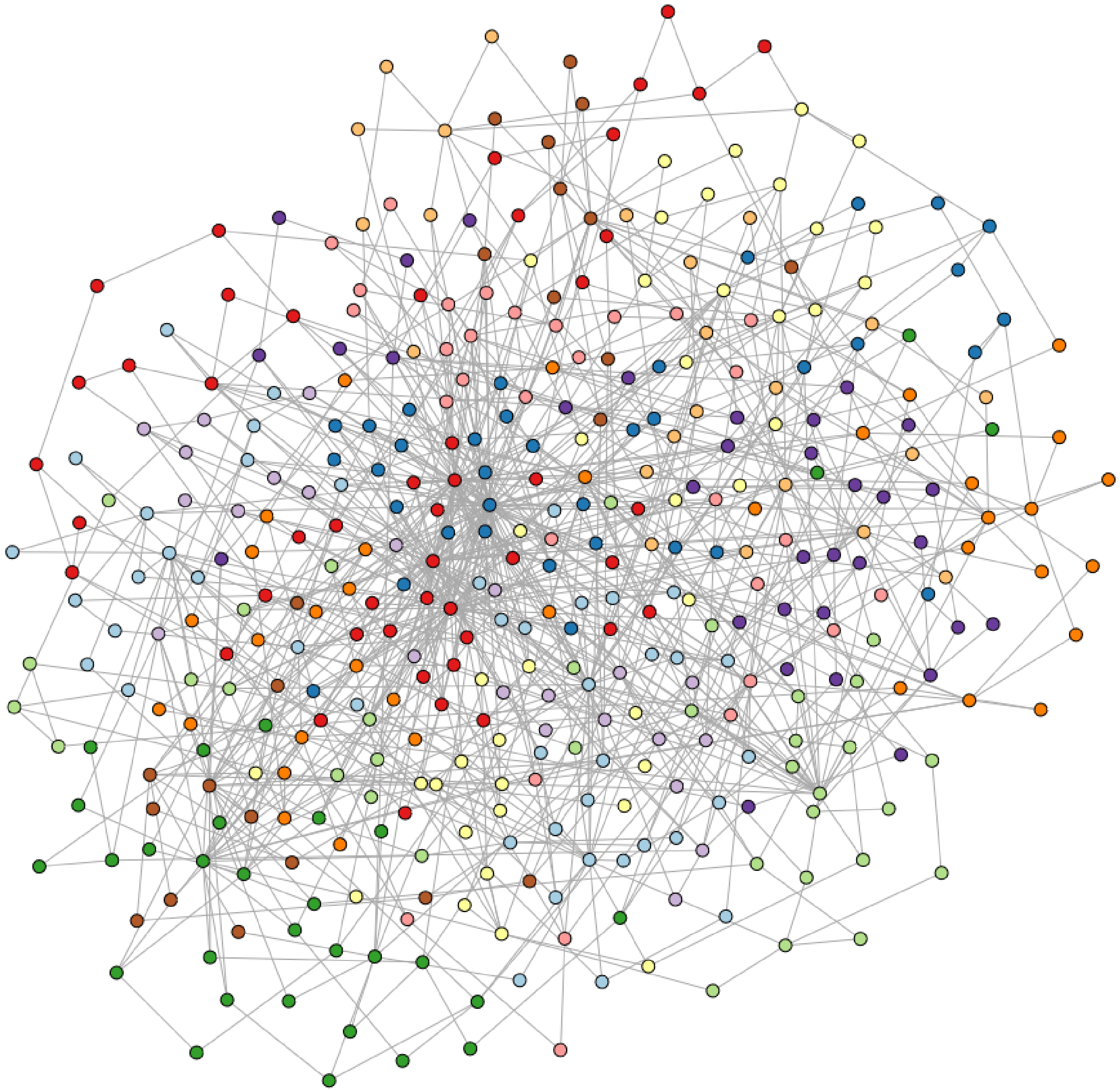
Given a pre-clustered graph =>

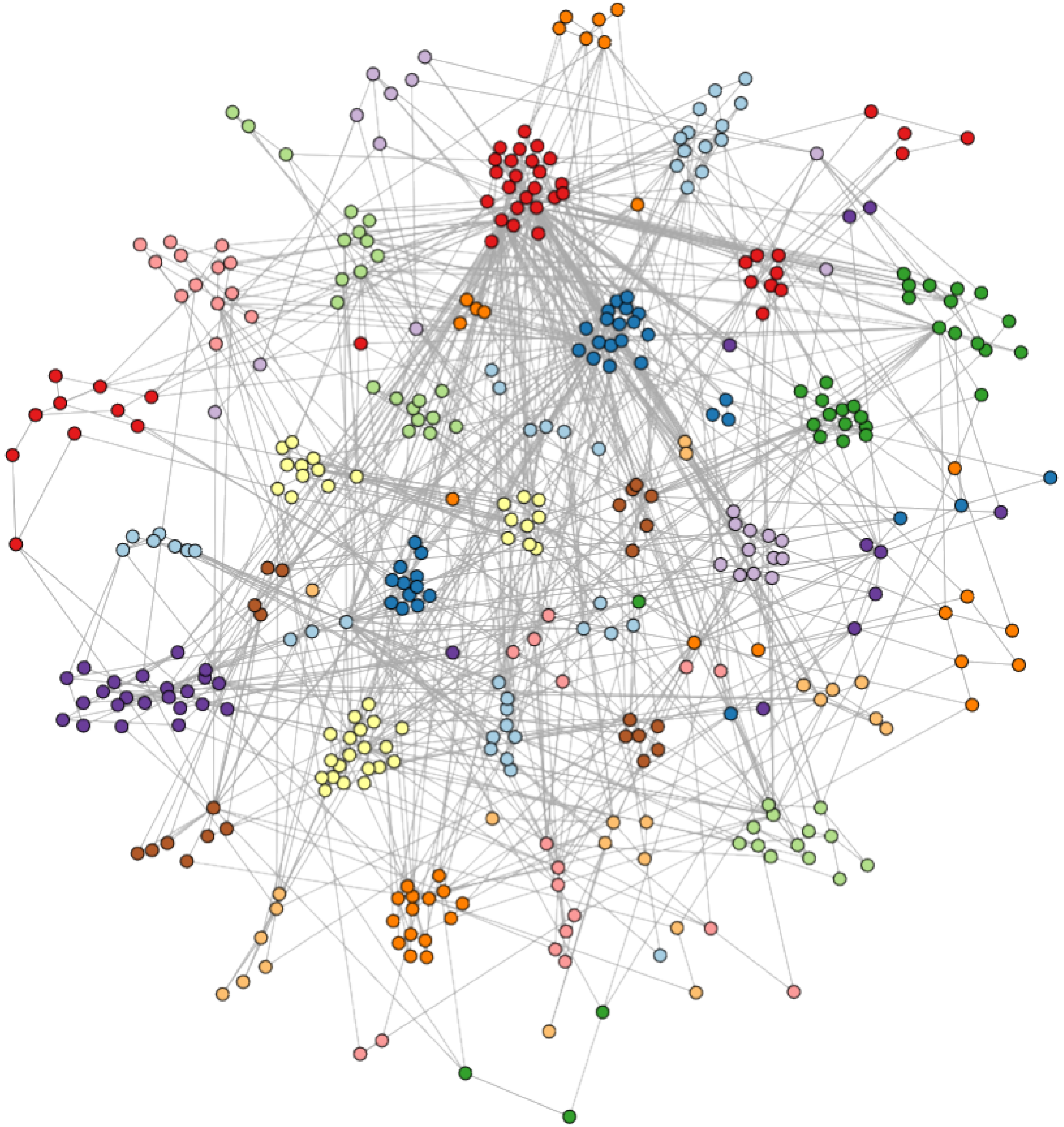
Push nodes of different cluster further apart:

3. For all node pairs (u,v) do
    4. Computer Repulsive Force according to Coulomb's Law: $F_C = k_r(Q_1 * Q_2 / r^2)$
    5. **If cluster(u) == cluster(v) do**
        6. Move u and v apart from each other by a distance F_C * velocity

    Else

        7. Move u and v apart from each other by a distance F_C *** 10** * velocity

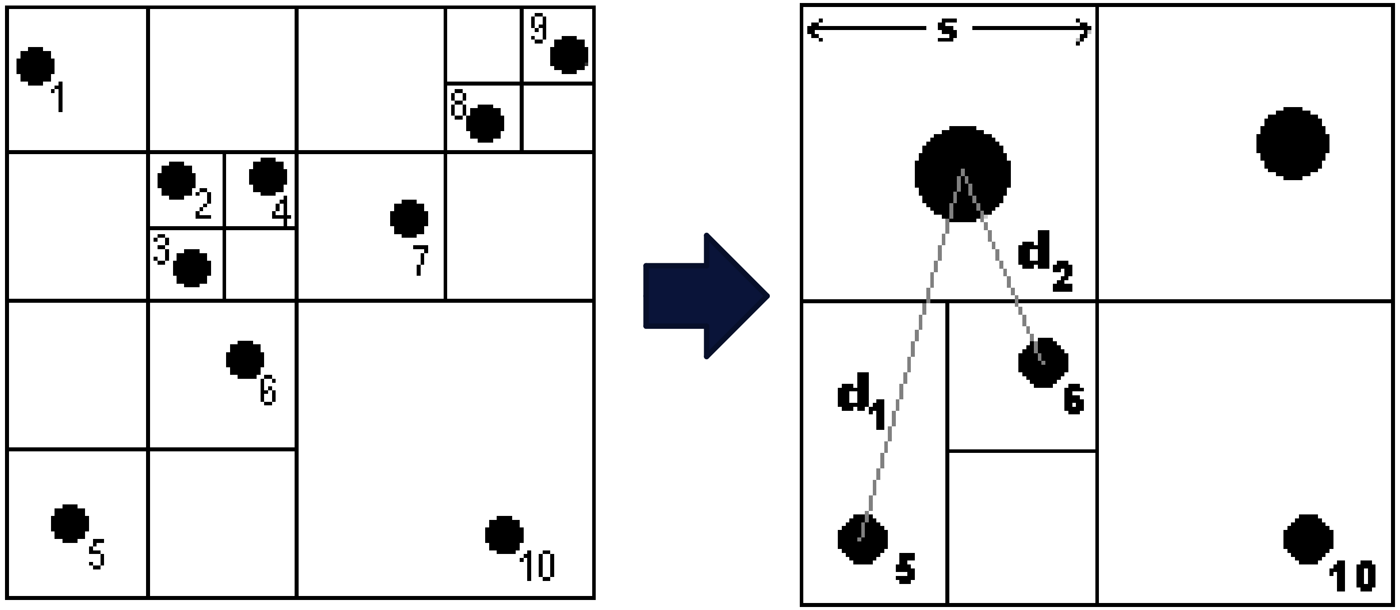




ADDON: BARNES-HUT OPTIMIZATION

Speed-up by reducing the number of comparisons

-> use a quadtree data structure to approximate forces from further-away nodes in bulk [Barnes & Hut 1986, Quigley & Eades 2000]



picture source: [Quigley et al. 2000]

ADDON: STIFFNESS FOR STREAMING GRAPHS

Different approaches ensure layout stiffness to the displacement of nodes:

Inter-Timeslice Edges (TGRIP) [Erten et al. 2004]

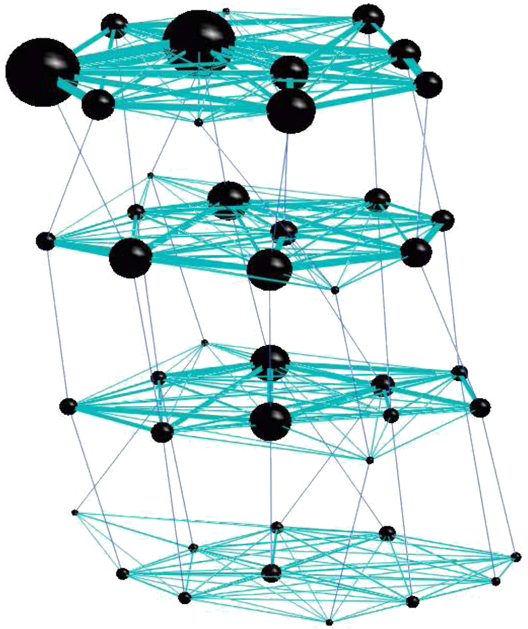
-> nodes that existed before have an added attracting force to previous position

Pinning [Tal, Frishman 2007]

-> node’s velocity !~ shortest path to last change

Aging [Gorochowski et al. 2012]

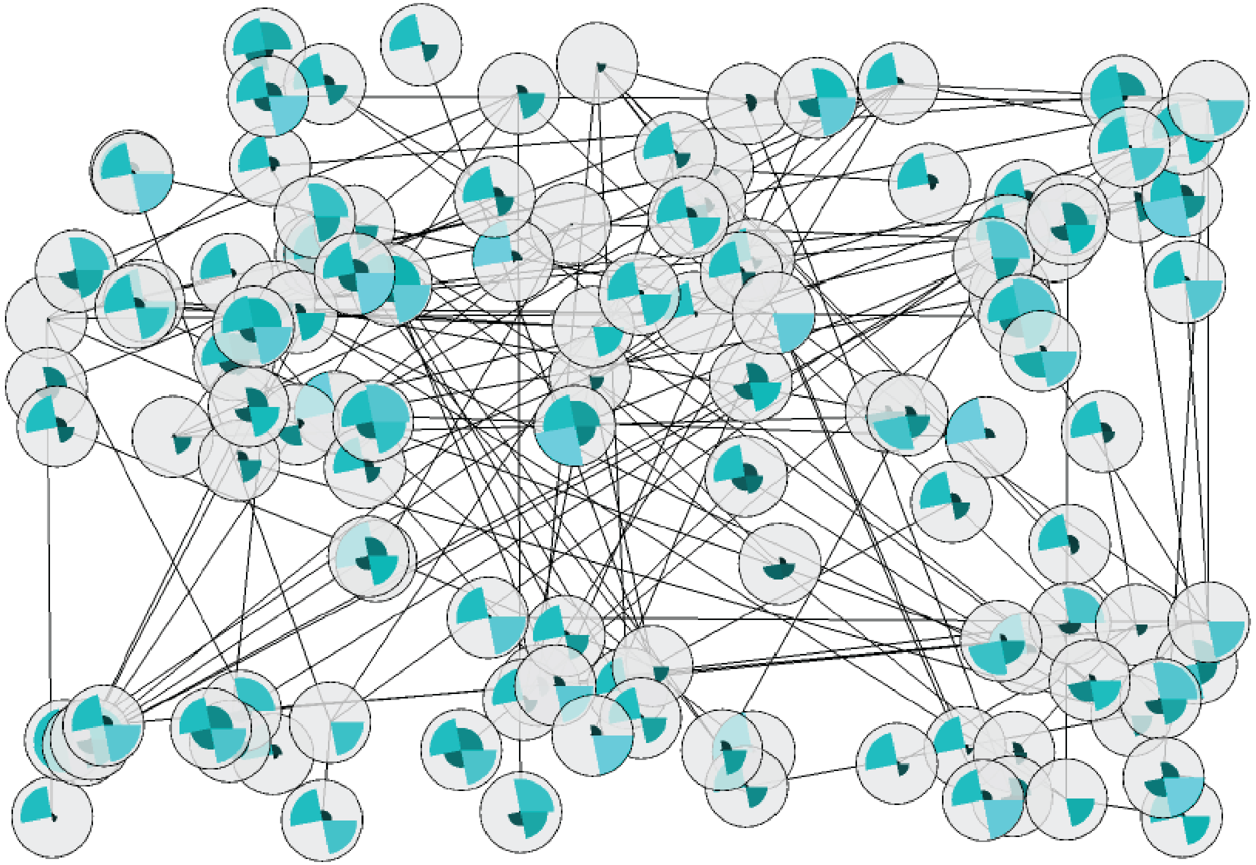
-> node’s velocity !~ time steps since it was inserted



[Erten et al 2004]

VISUALIZING NODE ATTRIBUTES

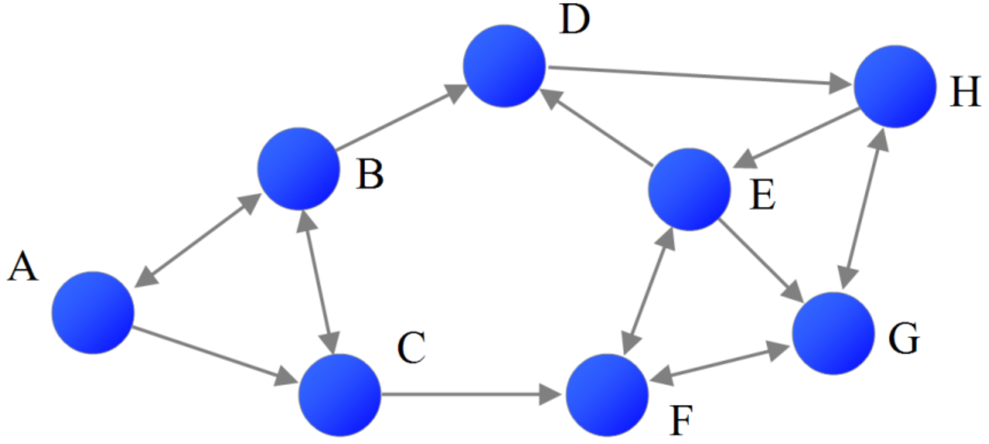
Using Glyphs for more complex Node Data:

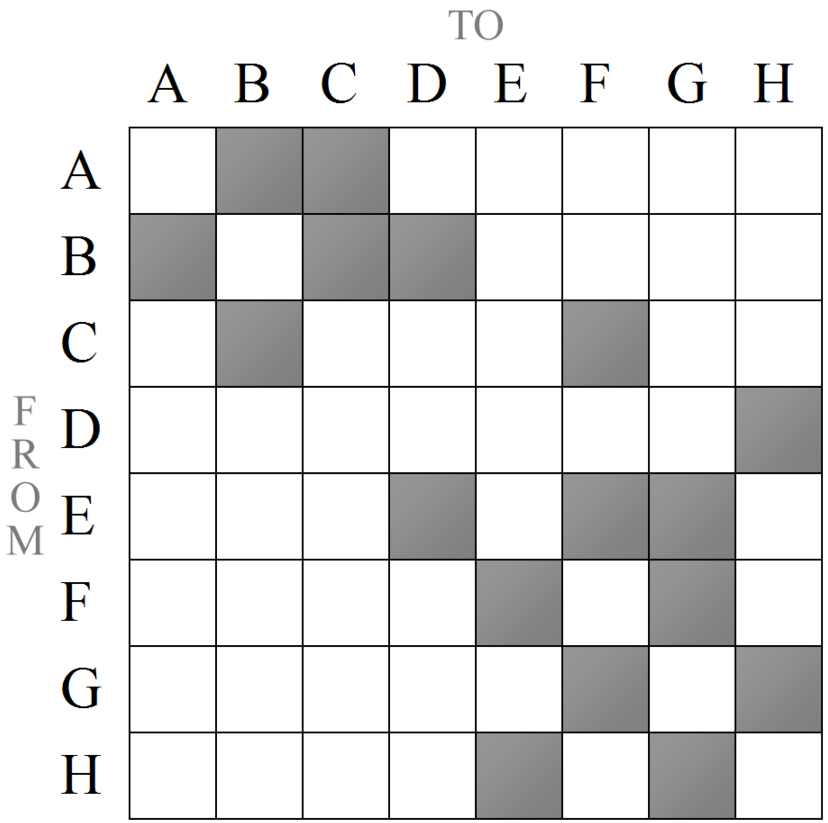


picture source: [McDonnel & Elmqvist 2009]

MATRIX REPRESENTATIONS

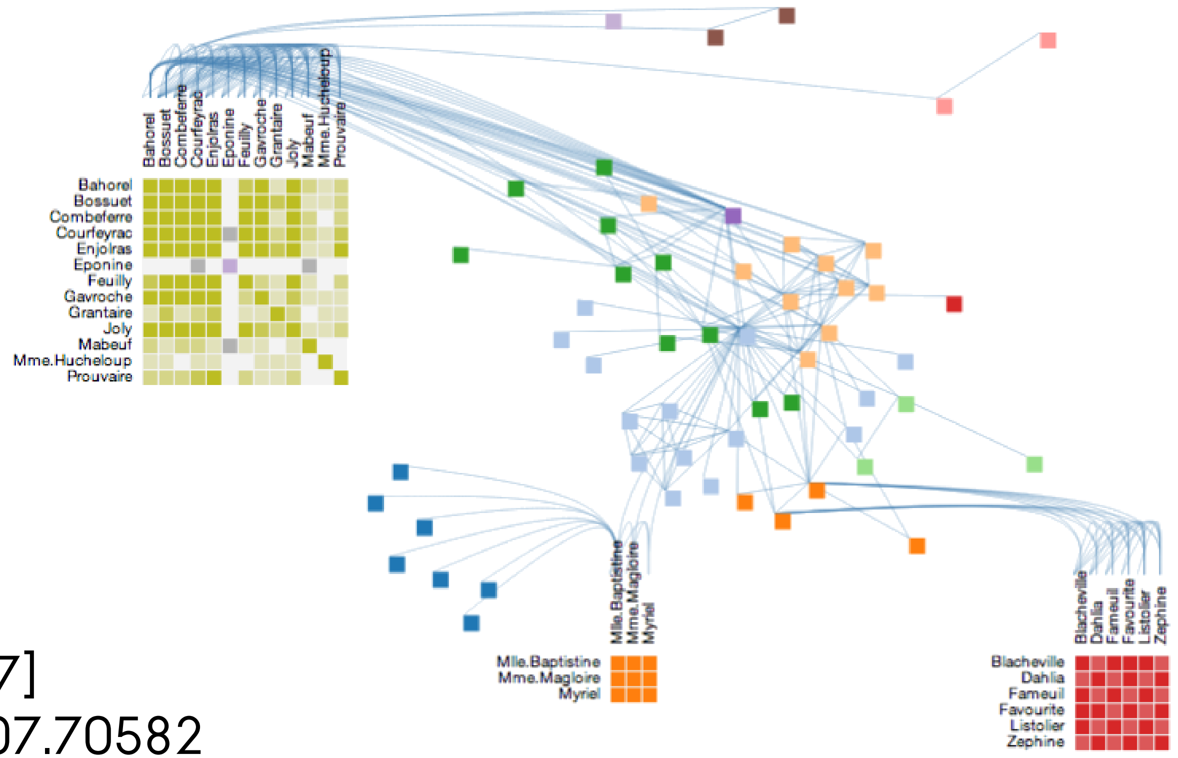
MATRIX VISUALIZATION





WHY USE MATRIX VISUALIZATIONS?

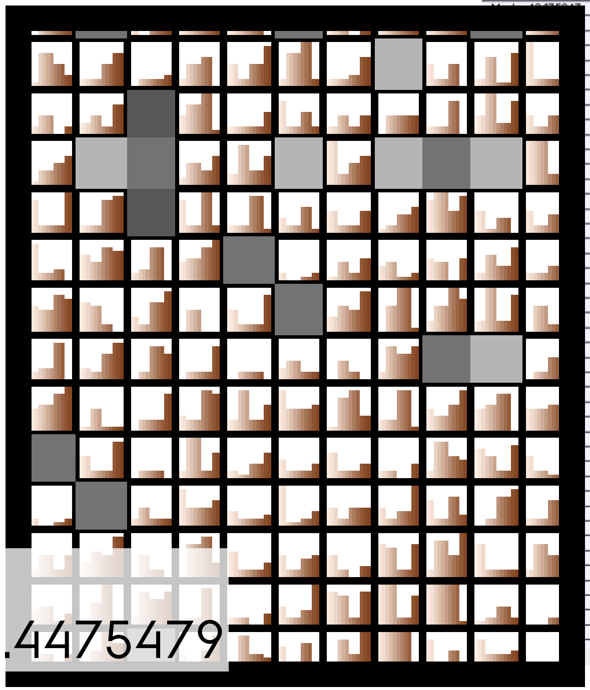
1) Compact representation of dense subgraphs

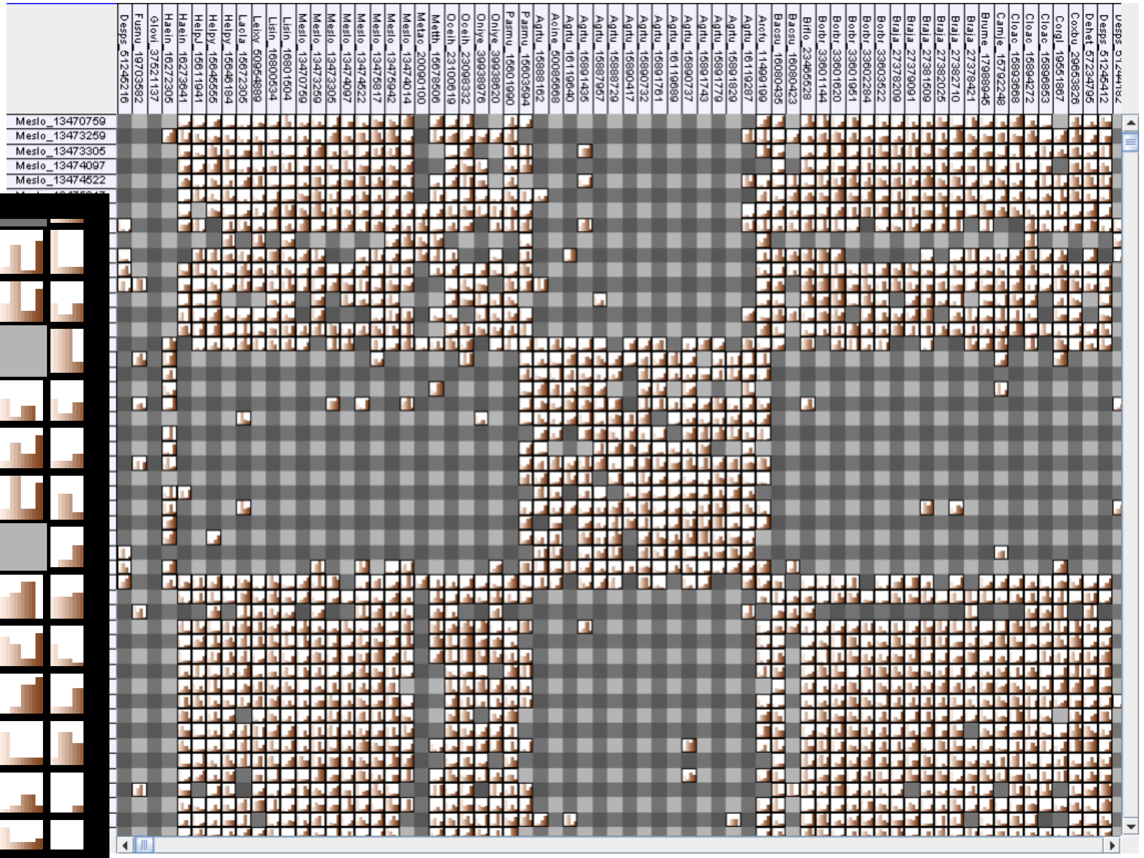


[Henry et al. 2007]
DOI: 10.1109/TVCG.2007.70582

WHY USE MATRIX VISUALIZATIONS?

2) For edge-centric displays

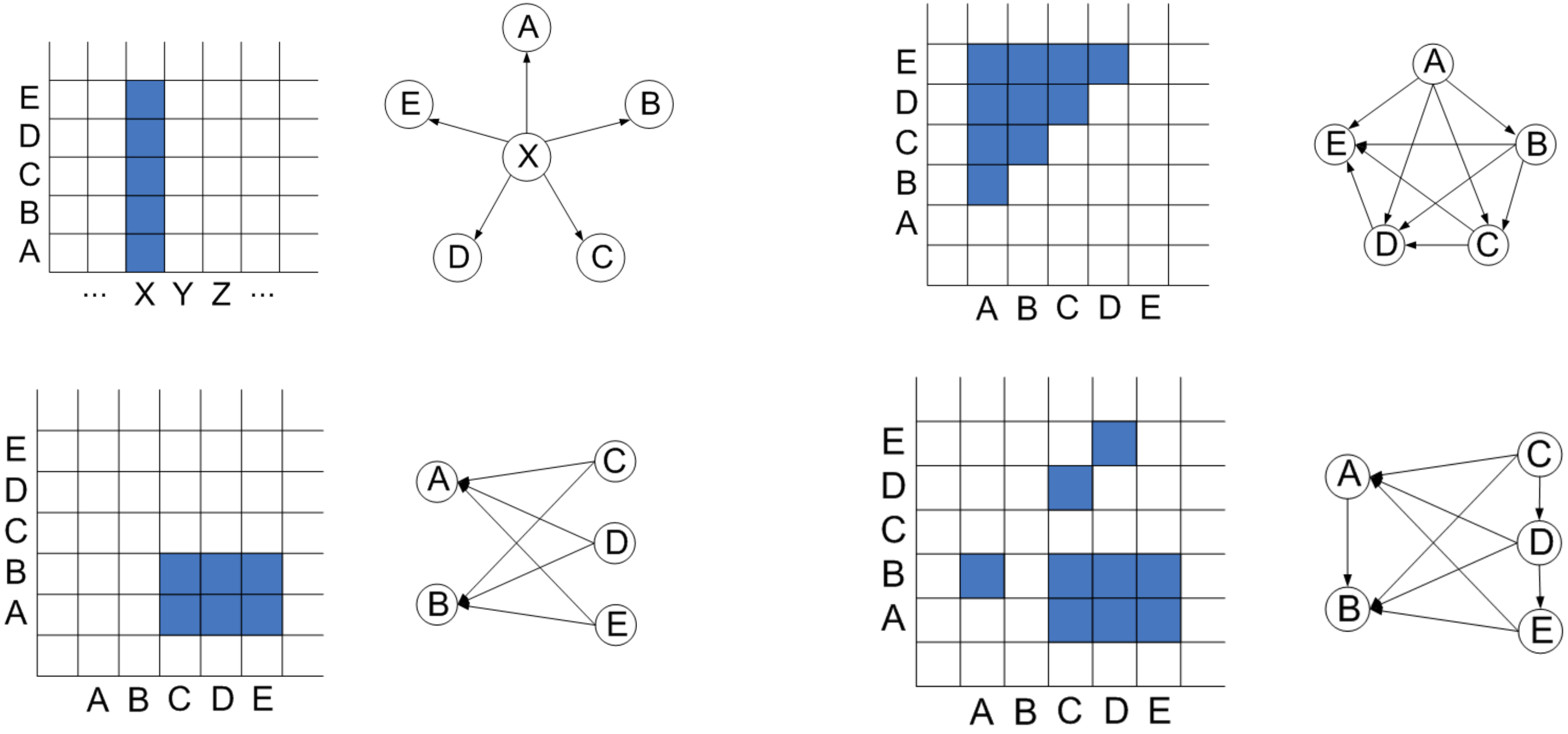




[Elmqvist et al. 2008]
DOI: 10.1109/PACIFICVIS.2008.4475479

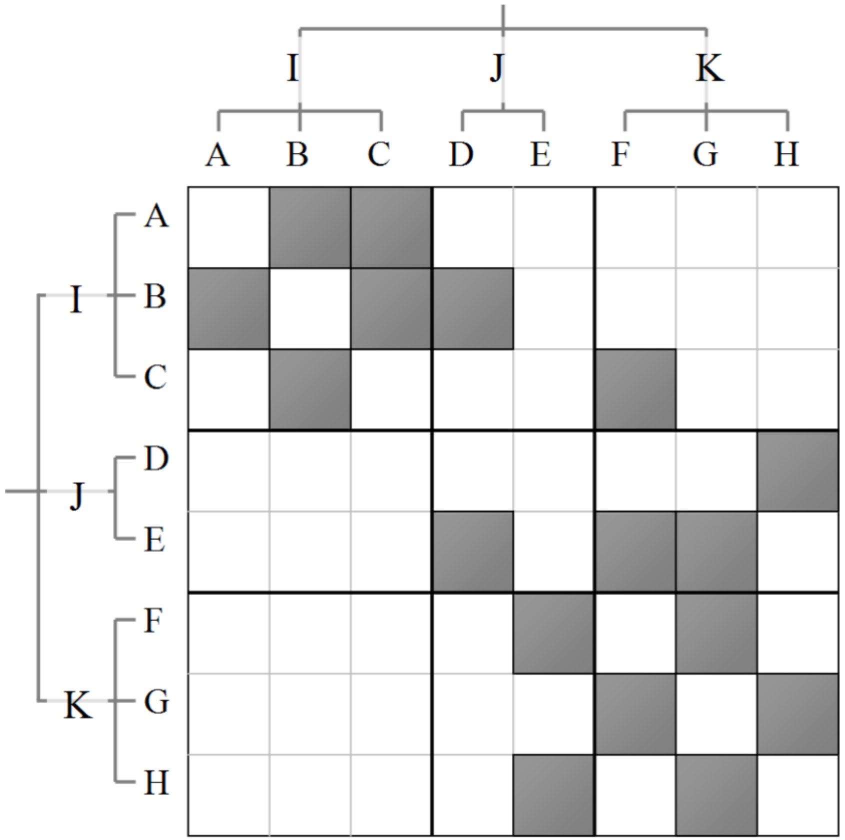
MATRIX VISUALIZATION

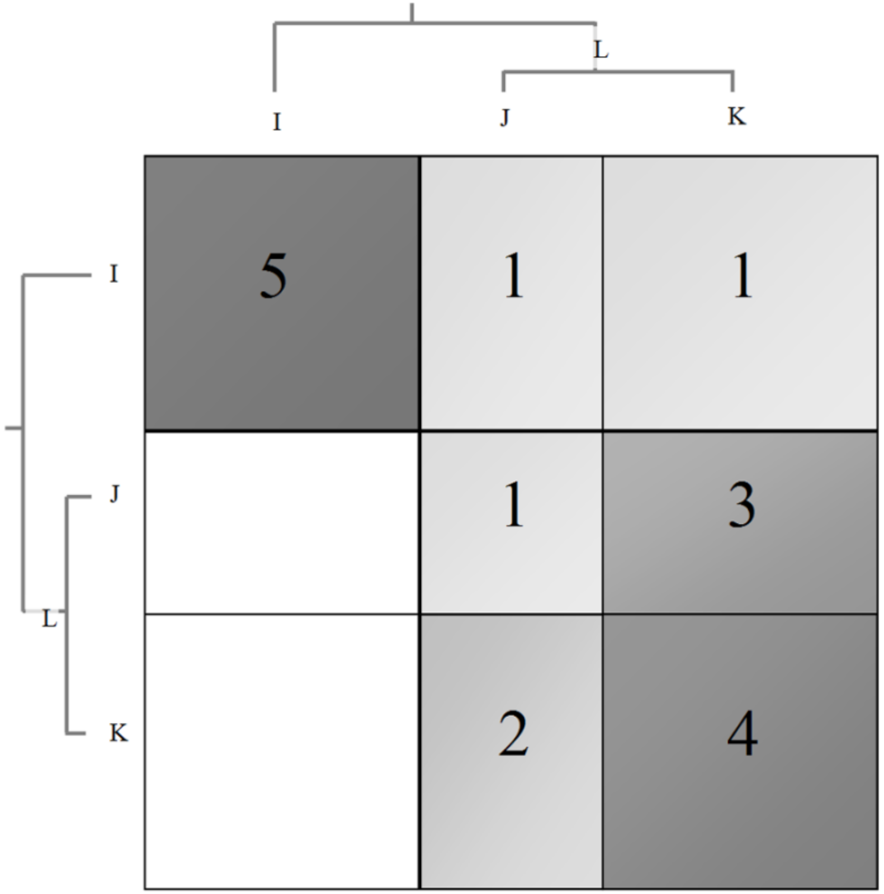
Examples [Shen & Ma 2007]:



HIERARCHICAL MATRIX VISUALIZATIONS

[v.Ham et al 2009]





EXPECTATION OF CONNECTING EDGES

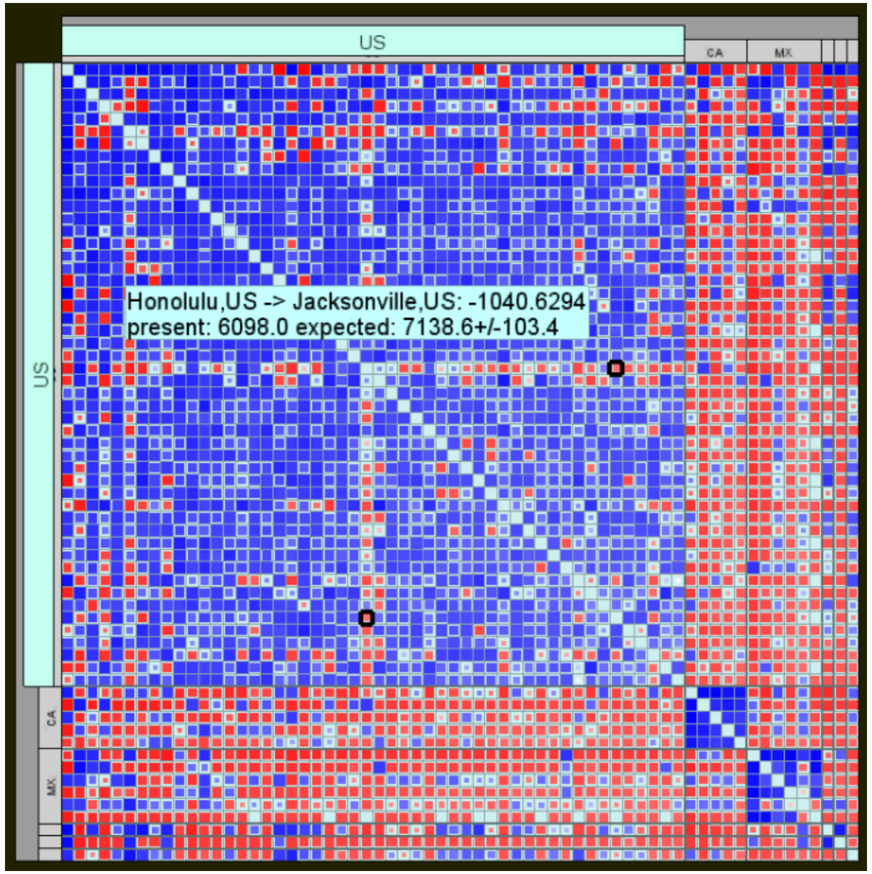
[v.Ham et al 2009]

$$P(X,Y) = \frac{X_{out}}{E} * \frac{Y_{in}}{E}$$

Probability of edges starting in X and ending in Y

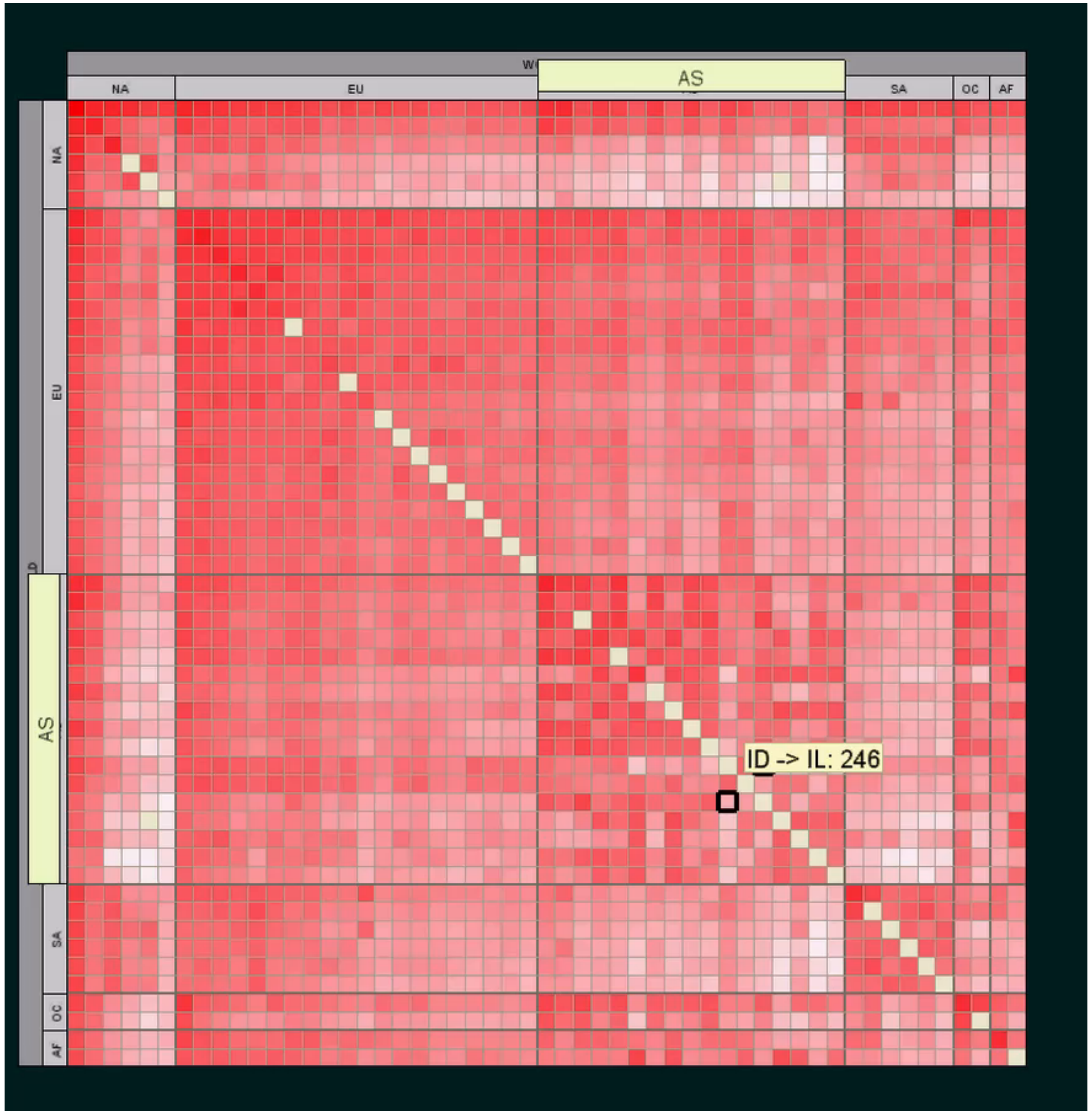
$$EXP(X,Y) = E * P(X,Y) = \frac{X_{out} \cdot Y_{in}}{E}$$

Expectation of the number of edges between X and Y out of all edges E



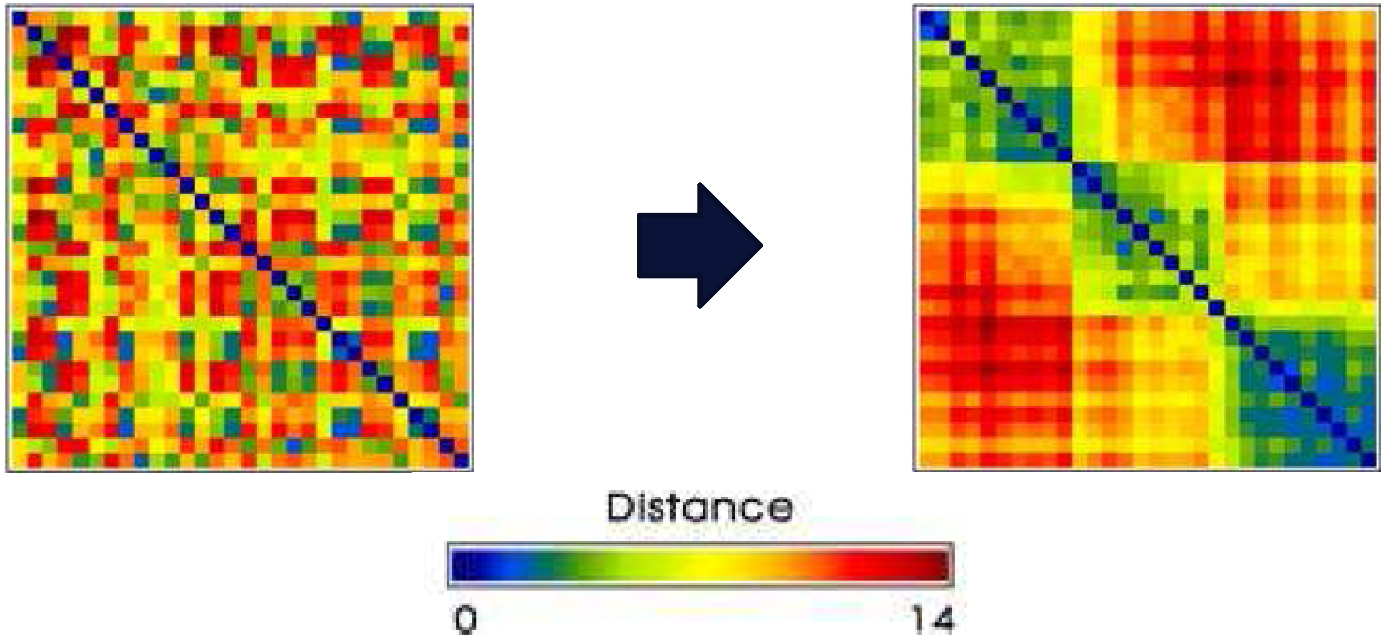
NAVIGATING HIERARCHICAL MATRICES

[v.Ham et al 2009]



ORDERING A MATRIX REPRESENTATION (1)

Aim: a linear order of rows and columns that makes the topological patterns clear(er)

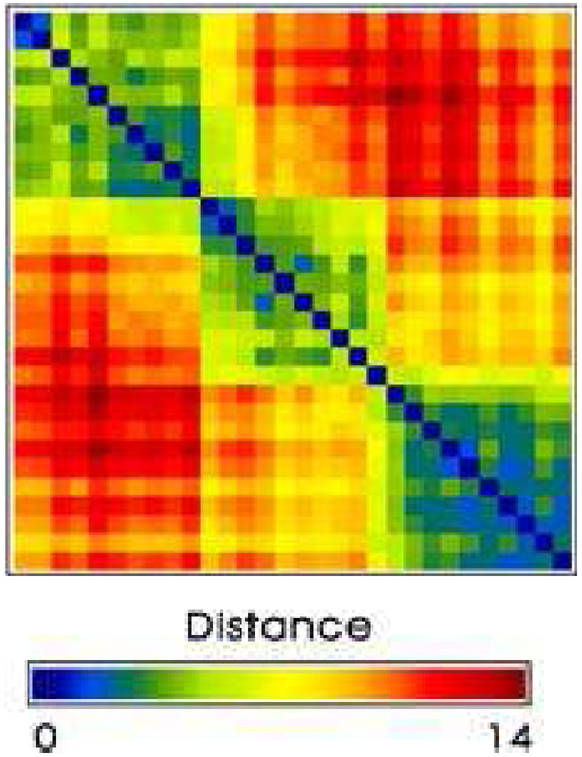


picture source: [Wu et al. 2008]

ORDERING A MATRIX REPRESENTATION (2)

Transform the matrix into a *Robinson Matrix*

picture source: [Wu et al. 2008]



Desired form: (near-)Robinson Matrix

“A MATRIX IS CALLED ROBINSON IF THE ELEMENTS IN ITS ROWS AND COLUMNS DO NOT INCREASE WHEN MOVING AWAY HORIZONTALLY OR VERTICALLY FROM ITS MAIN DIAGONAL.” (from N.Henry 2008)

There are multiple ways of computing this transformation.

ORDERING A MATRIX REPRESENTATION (3)

Solution: Make it into an optimization problem

Possible cost functions [Wu et al. 2008]:

$$AR(i) = \sum_{i=1}^{p}\left[\sum_{j<k<i} I(d_{ij} < d_{ik}) + \sum_{i<j<k} I(d_{ij} > d_{ik})\right]$$

# of all Robinson violations

$$AR(s) = \sum_{i=1}^{p}\left[\sum_{j<k<i} I(d_{ij} < d_{ik}) \cdot |d_{ij} - d_{ik}| + \sum_{i<j<k} I(d_{ij} > d_{ik}) \cdot |d_{ij} - d_{ik}|\right]$$

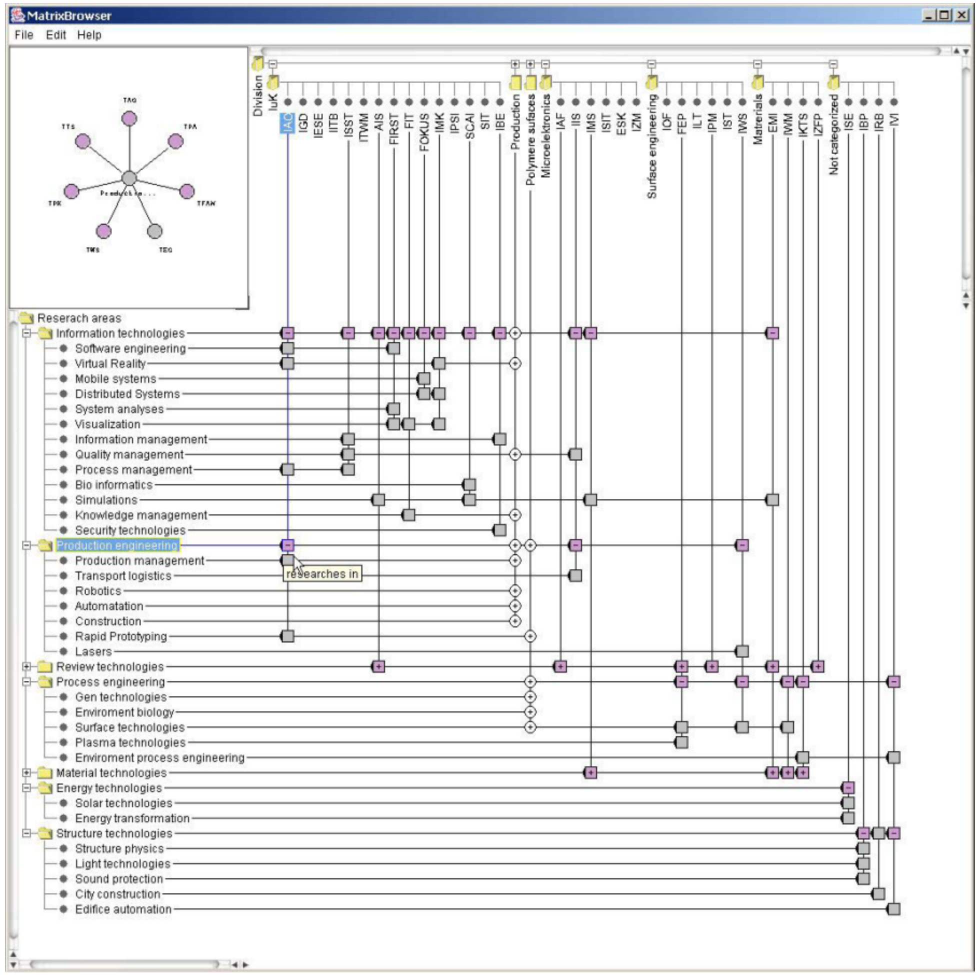
Σ of the deviations of all Robinson violations

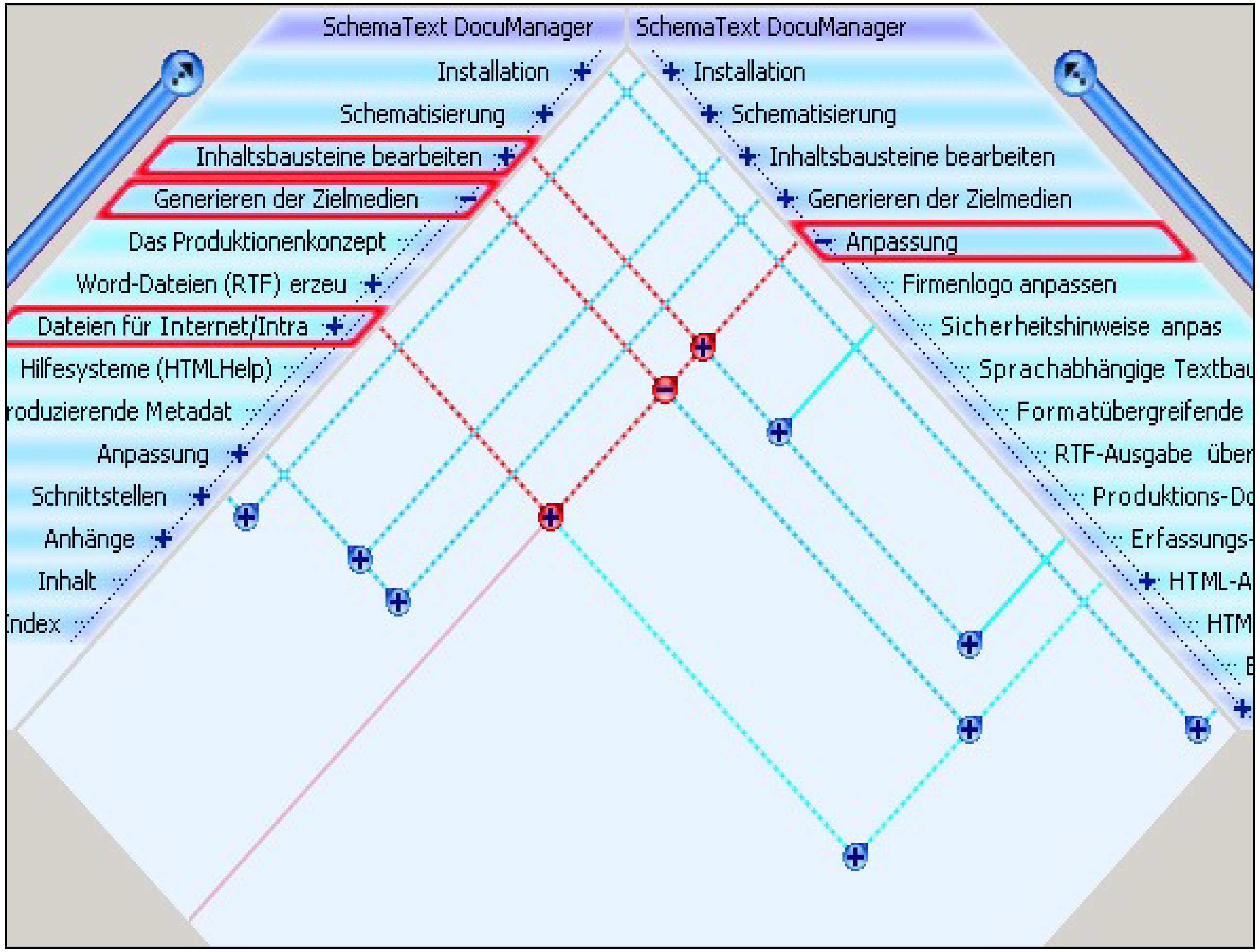
$$AR(w) = \sum_{i=1}^{p}\left[\sum_{j<k<i} I(d_{ij} < d_{ik})|j-k||d_{ij} - d_{ik}| + \sum_{i<j<k} I(d_{ij} > d_{ik})|j-k||d_{ij} - d_{ik}|\right]$$

Σ of the weighted deviations of all Robinson violations (weight = index difference)

NODE-LINK MATRICES

picture source: [Beinhauer & Ziegler 2003]

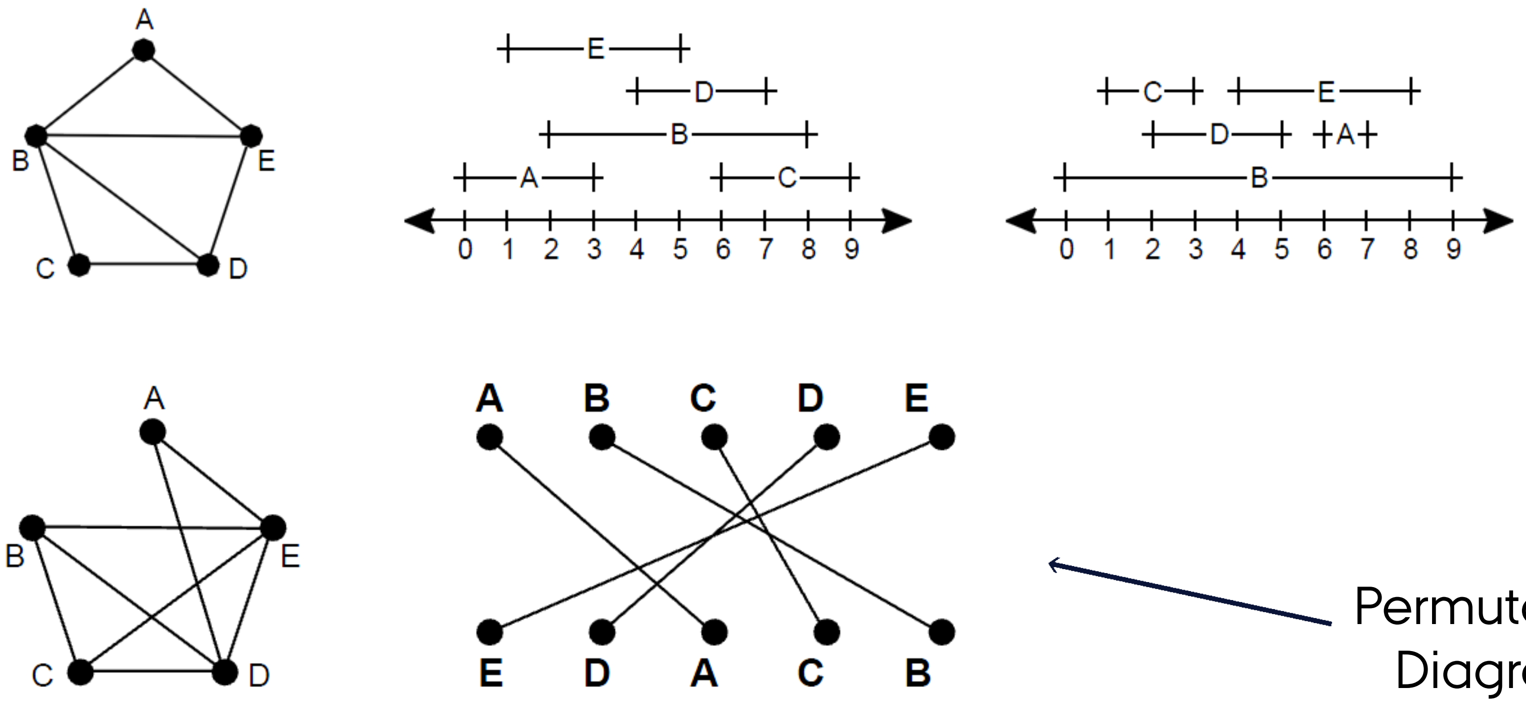




IMPLICIT TREE VISUALIZATIONS

MAIN IDEA BEHIND IMPLICIT LAYOUTS

Representation of the edges through meaningful positioning of the nodes

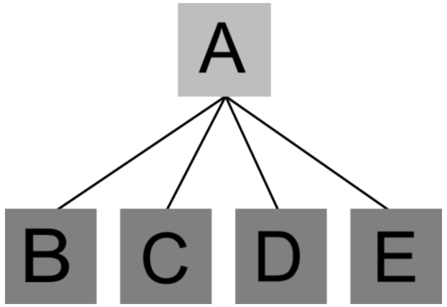


Interval Representation

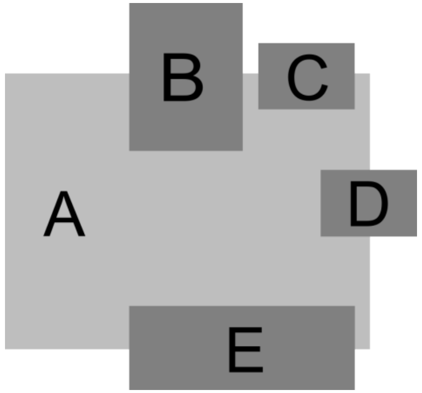
Permutation Diagram

...SPECIFICALLY FOR TREES

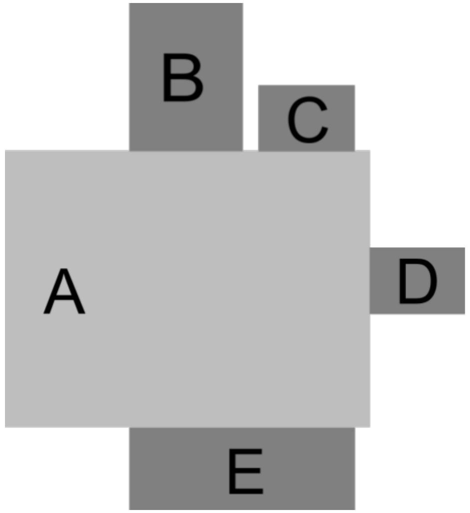
Explicit = Node-Link-Diagram



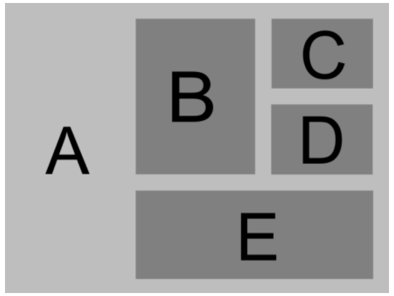
Implicit = Positional Hierarchy



Overlap

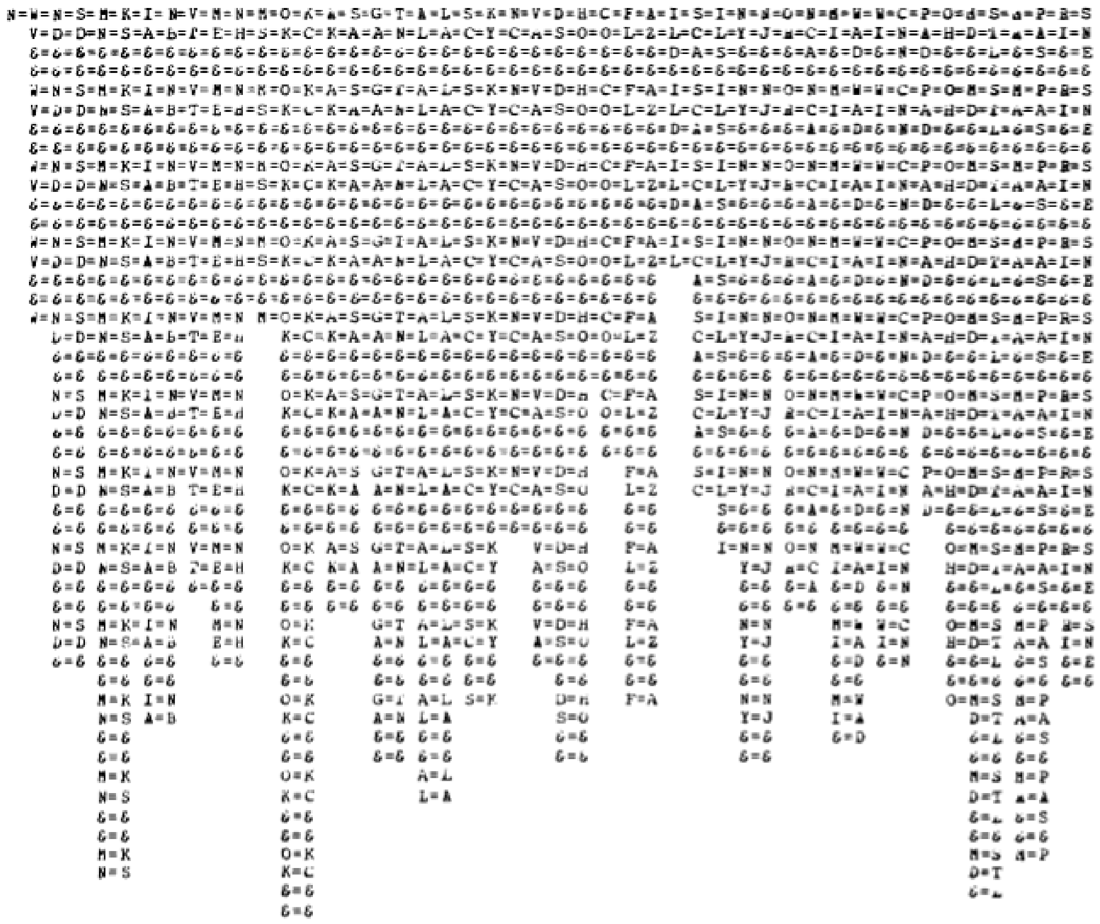


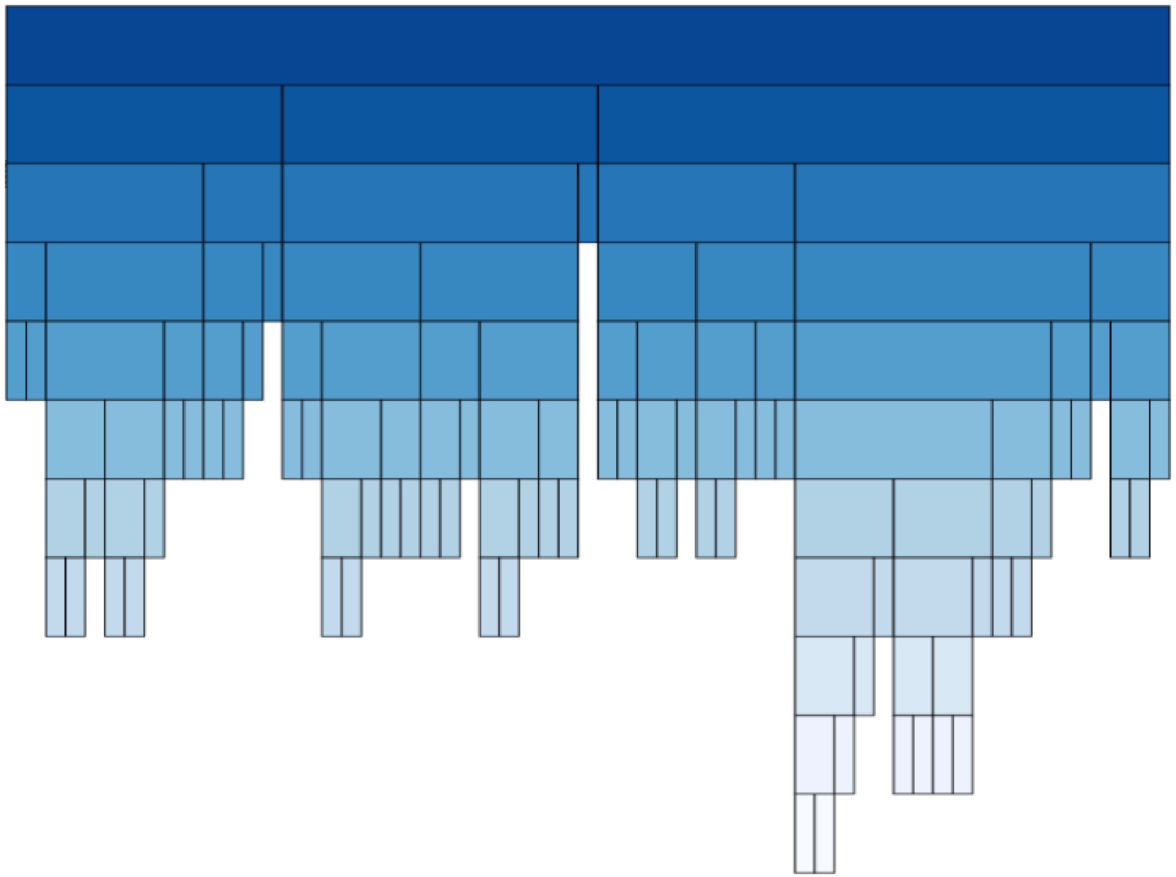
Adjacency



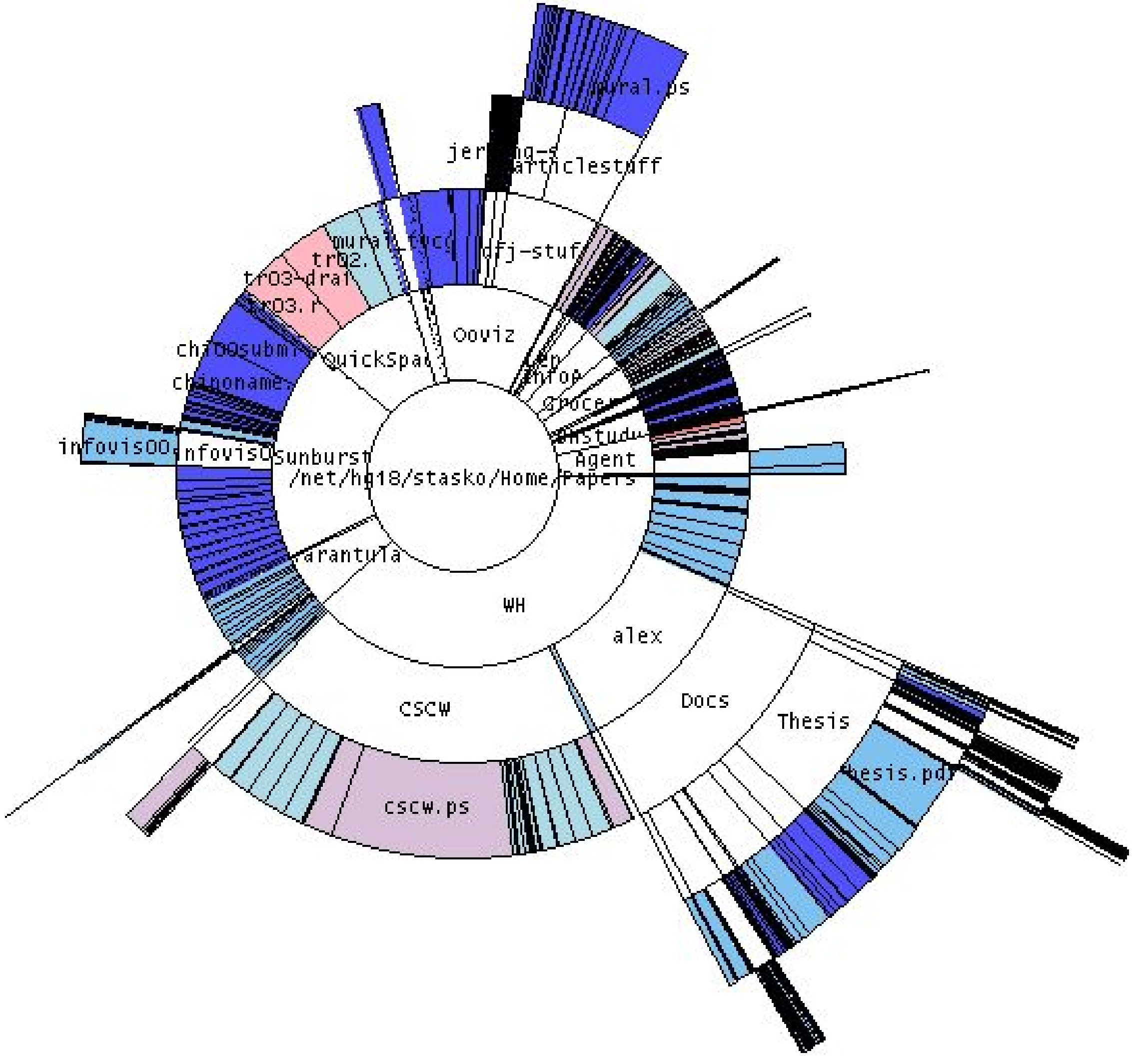
Inclusion

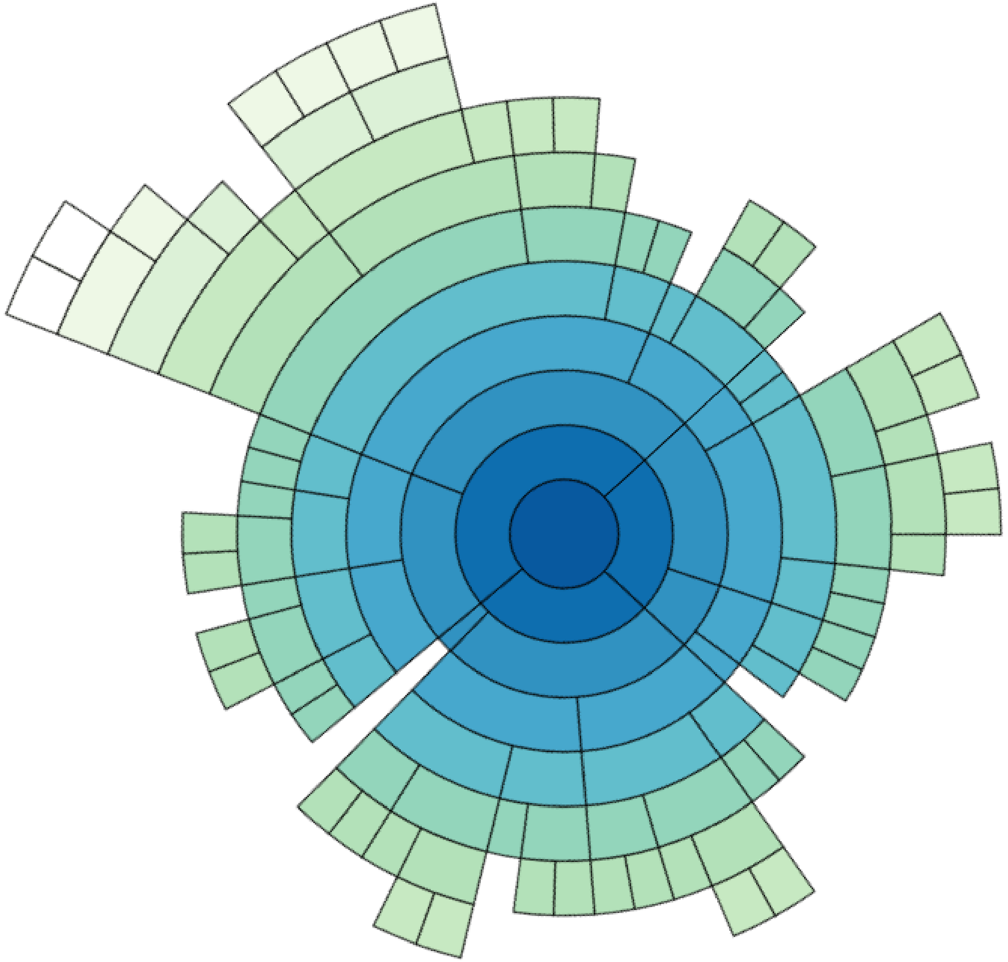
ADJACENCY: ICICLE PLOT (1983)



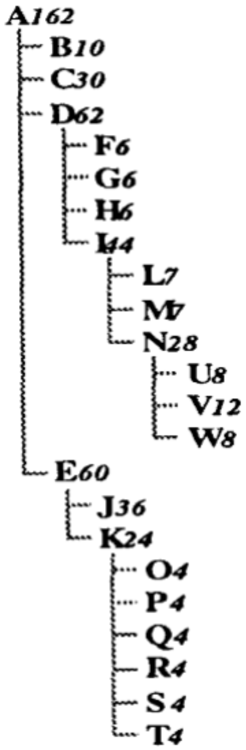


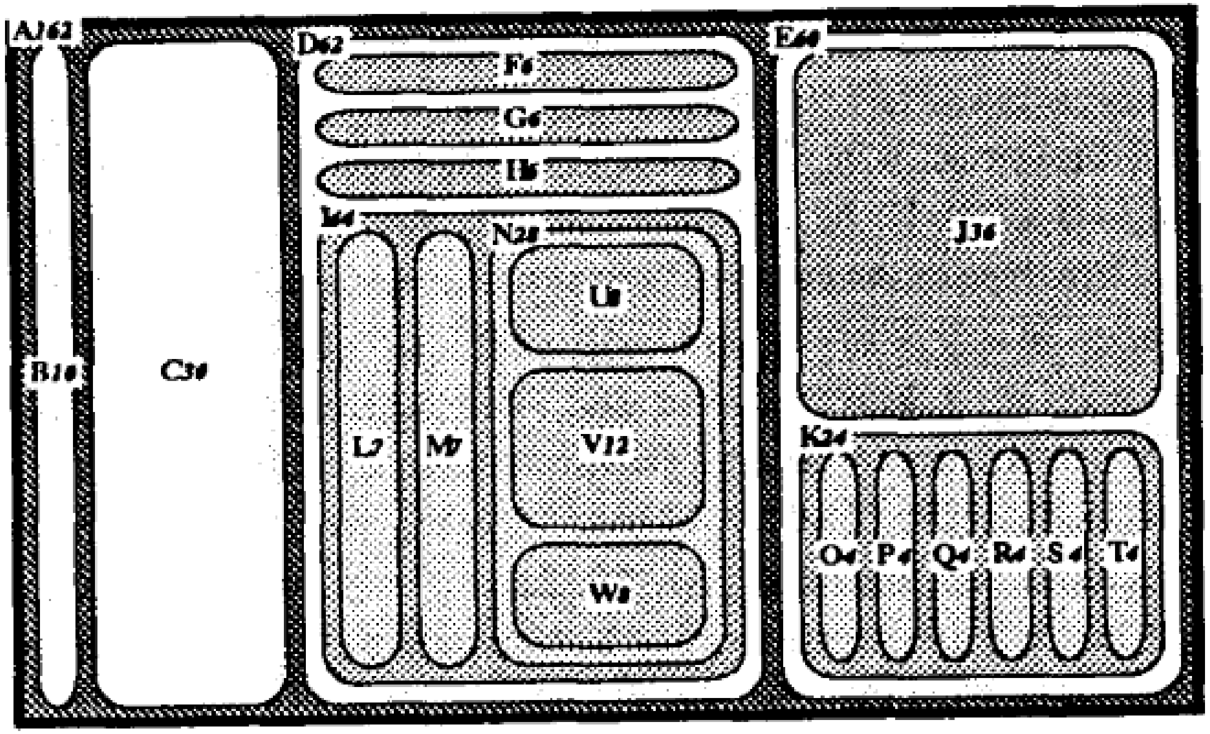
ADJACENCY: SUNBURST (2000)



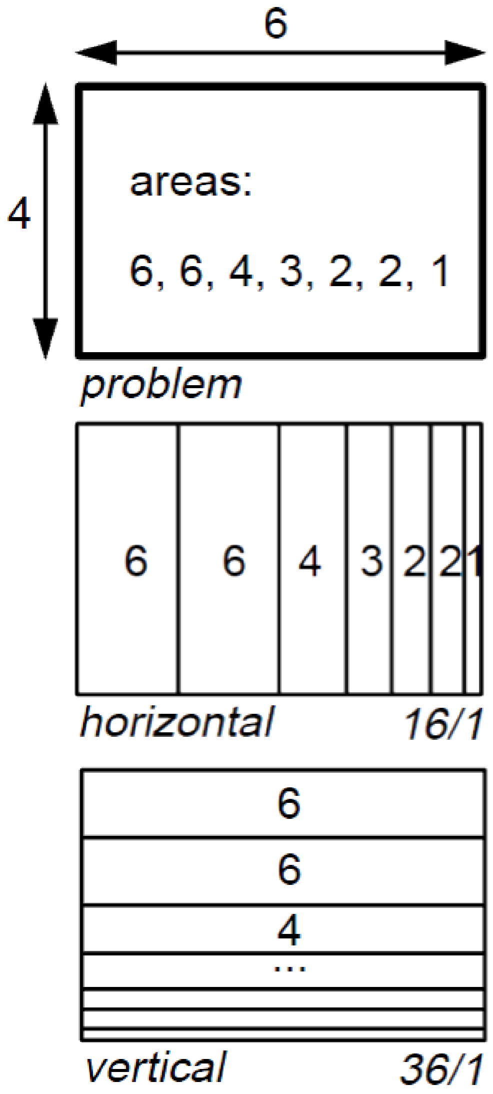


INCLUSION: TREEMAP (SLICE & DICE)





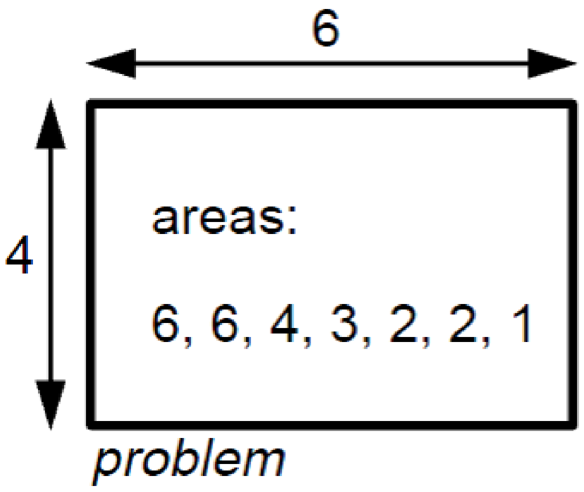
picture source: [Johnson & Shneiderman 1991]

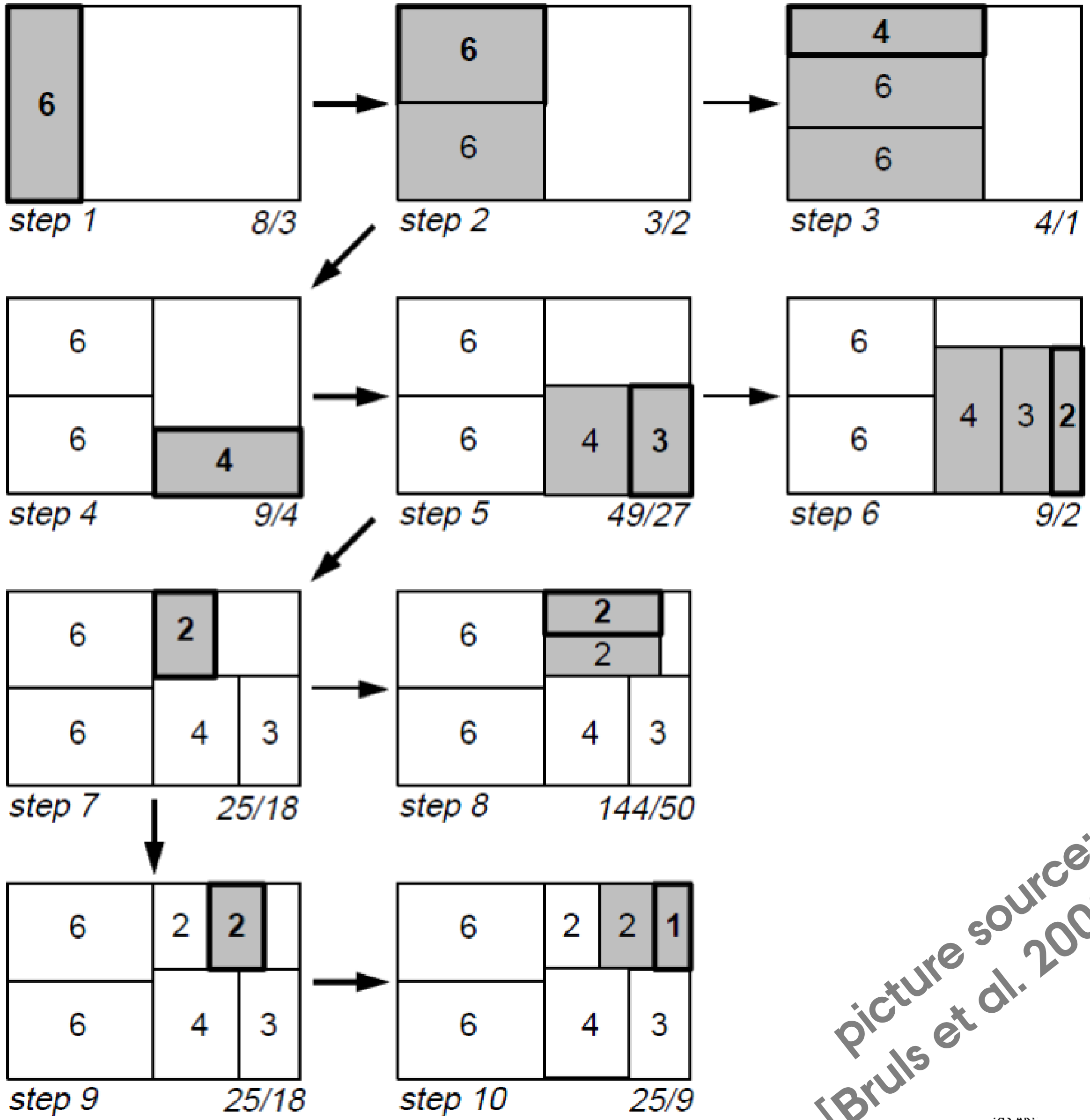


picture source: [Bruls et al. 2000]

INCLUSION: TREEMAP (SQUARIFIED)

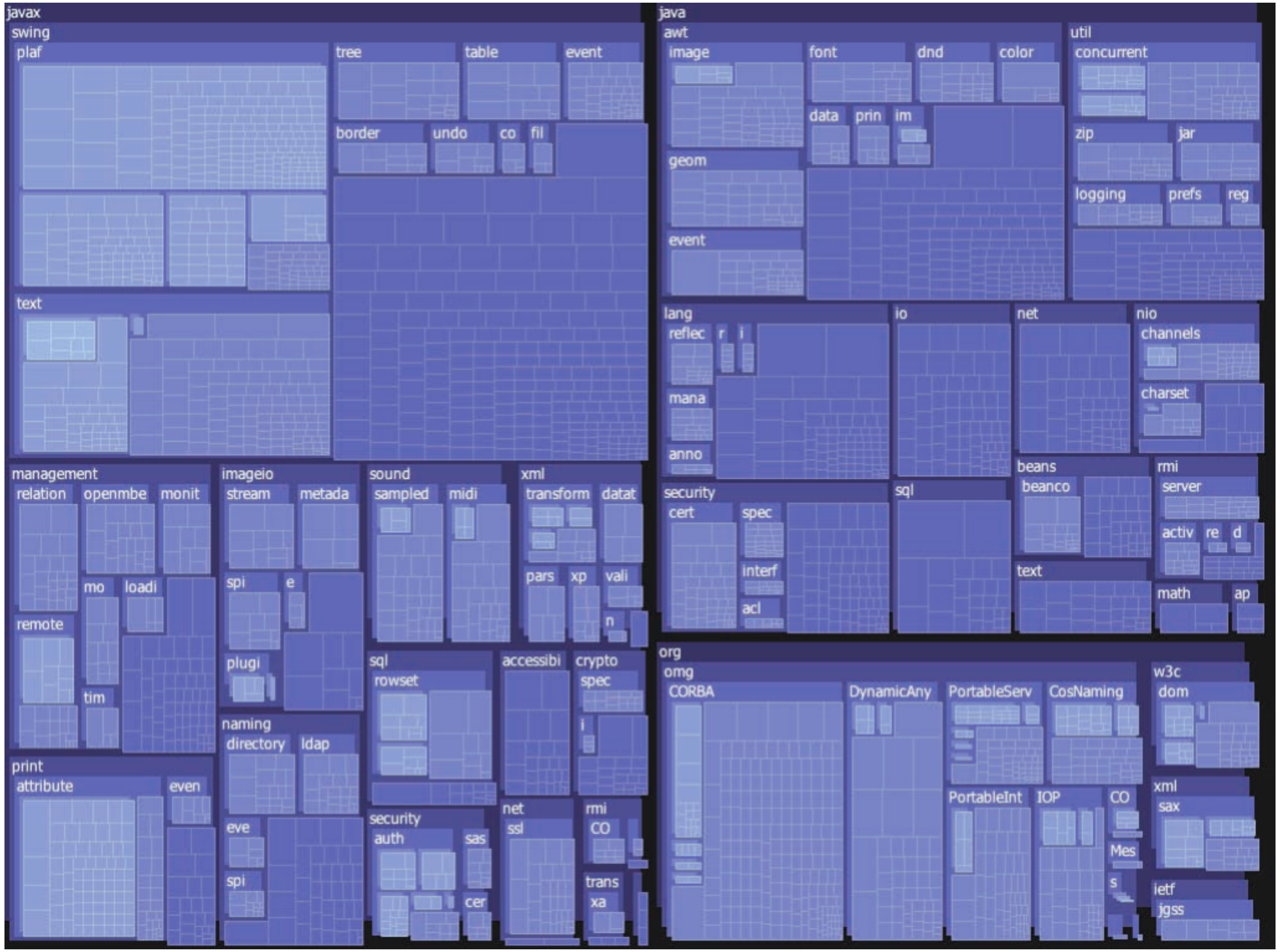
Idea: choose greedily which axis to add a region to -> always the one that yields the better aspect ratio
(start with the largest regions)

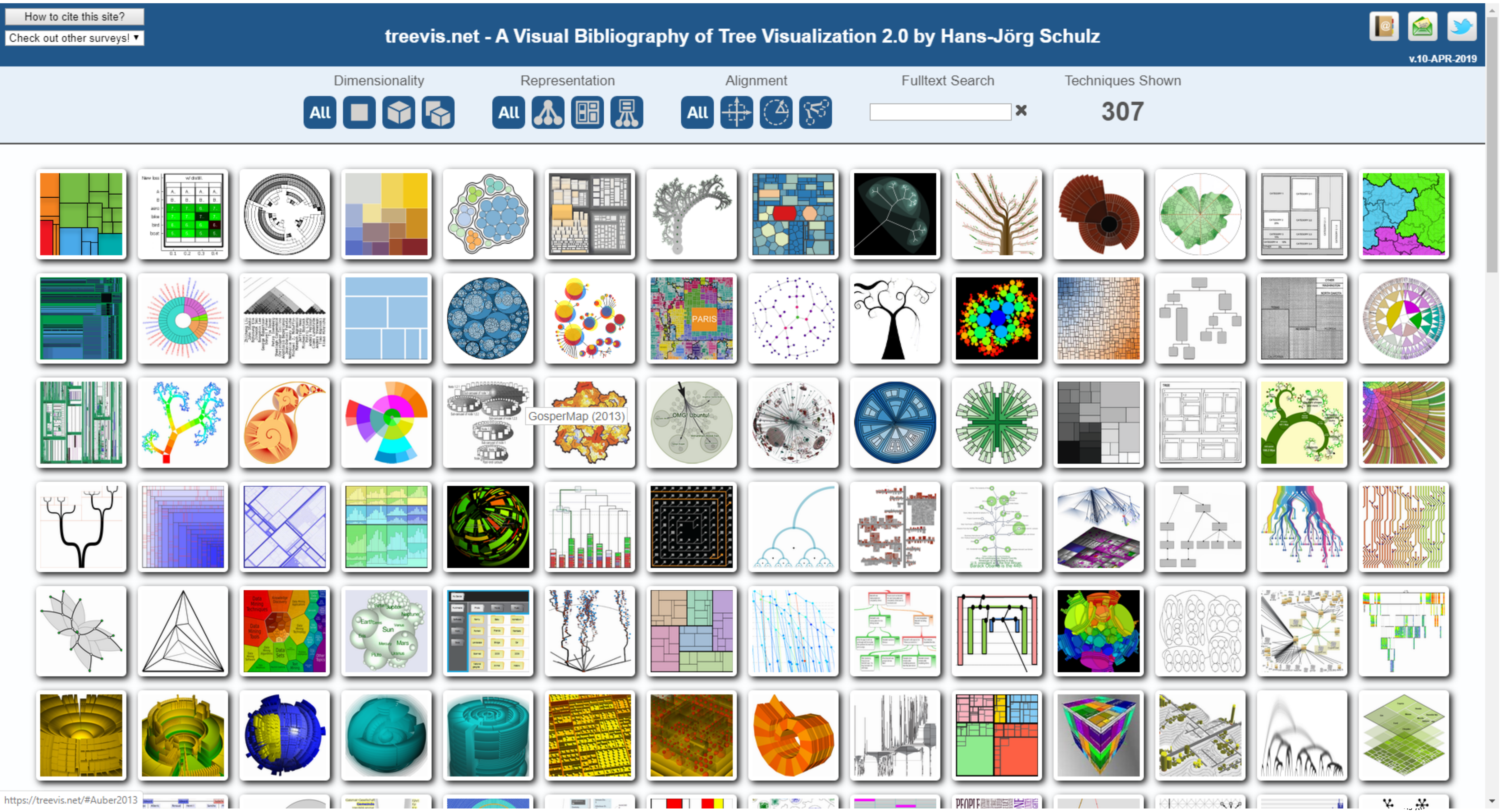




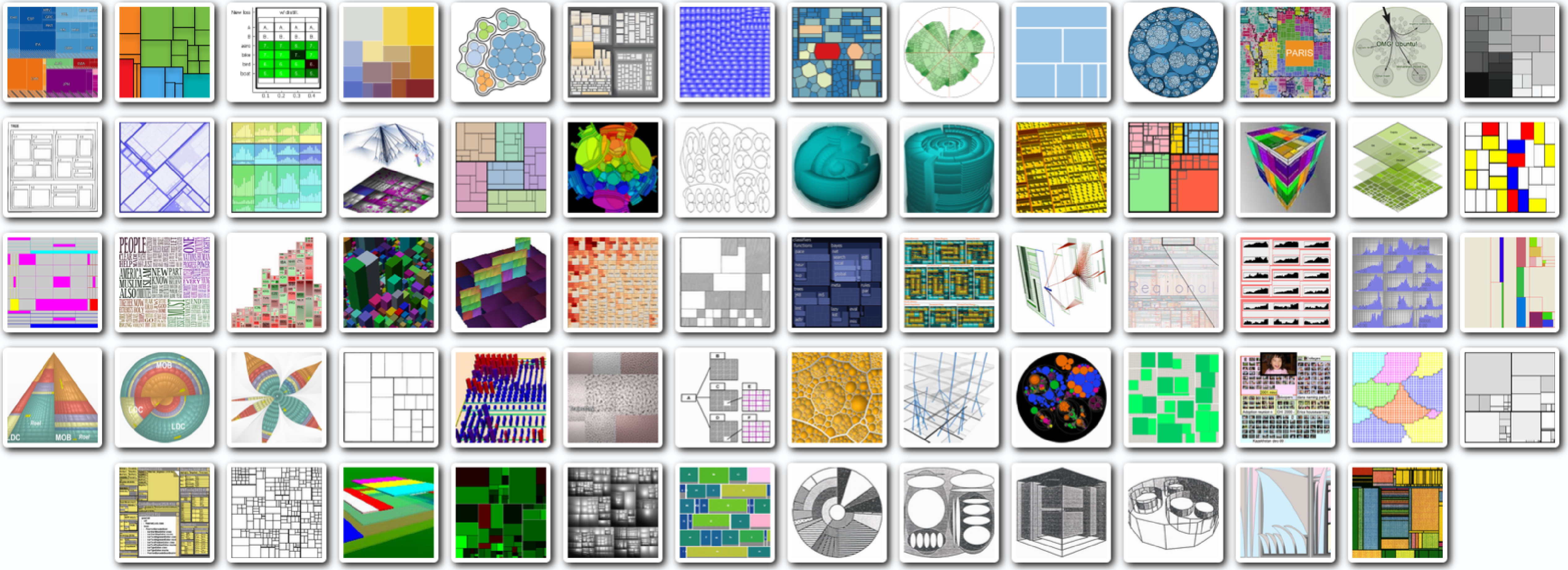
picture source: [Bruls et al. 2000]

OVERLAP: CASCADED TREEMAPS (2008)

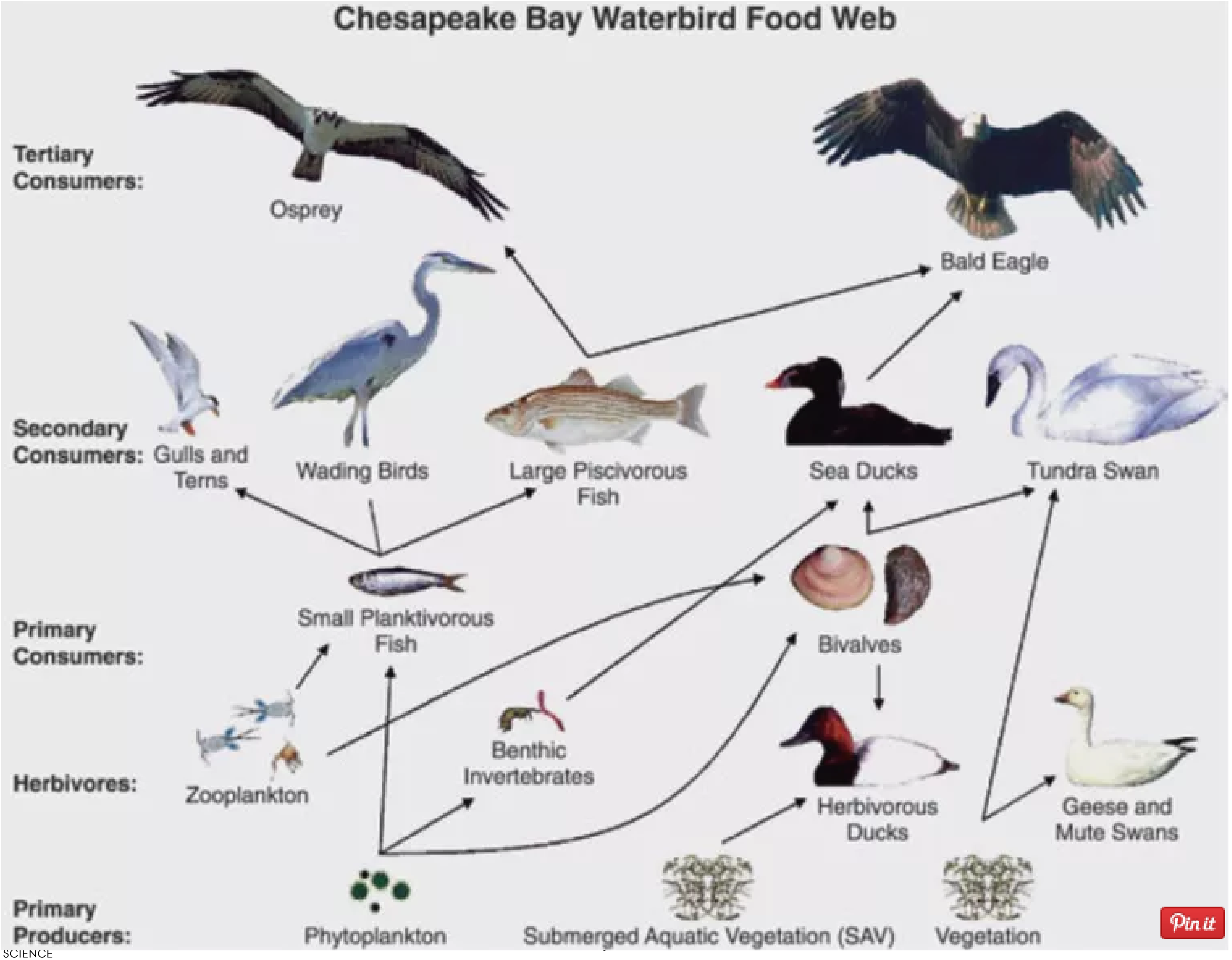



How to cite this site?
Check out other surveys!
treevis.net - A Visual Bibliography of Tree Visualization 2.0 by Hans-Jörg Schulz
v.10-APR-2019
Dimensionality
All
Representation
All
Alignment
All
Fulltext Search
Techniques Shown
307
GosperMap (2013)
PARIS
https://treevis.net/#Auber2013

TREEMAPS ON TREEVIS.NET



VISUALIZATION OF PO-SETS


Chesapeake Bay Waterbird Food Web
Tertiary Consumers:
Osprey
Bald Eagle
Secondary Consumers:
Gulls and Terns
Wading Birds
Large Piscivorous Fish
Sea Ducks
Tundra Swan
Primary Consumers:
Small Planktivorous Fish
Bivalves
Herbivores:
Zooplankton
Benthic Invertebrates
Herbivorous Ducks
Geese and Mute Swans
Primary Producers:
Phytoplankton
Submerged Aquatic Vegetation (SAV)
Vegetation
Pin it

LAYERED DRAWING (SUGIYAMA ET AL. 1981)

3 Steps:

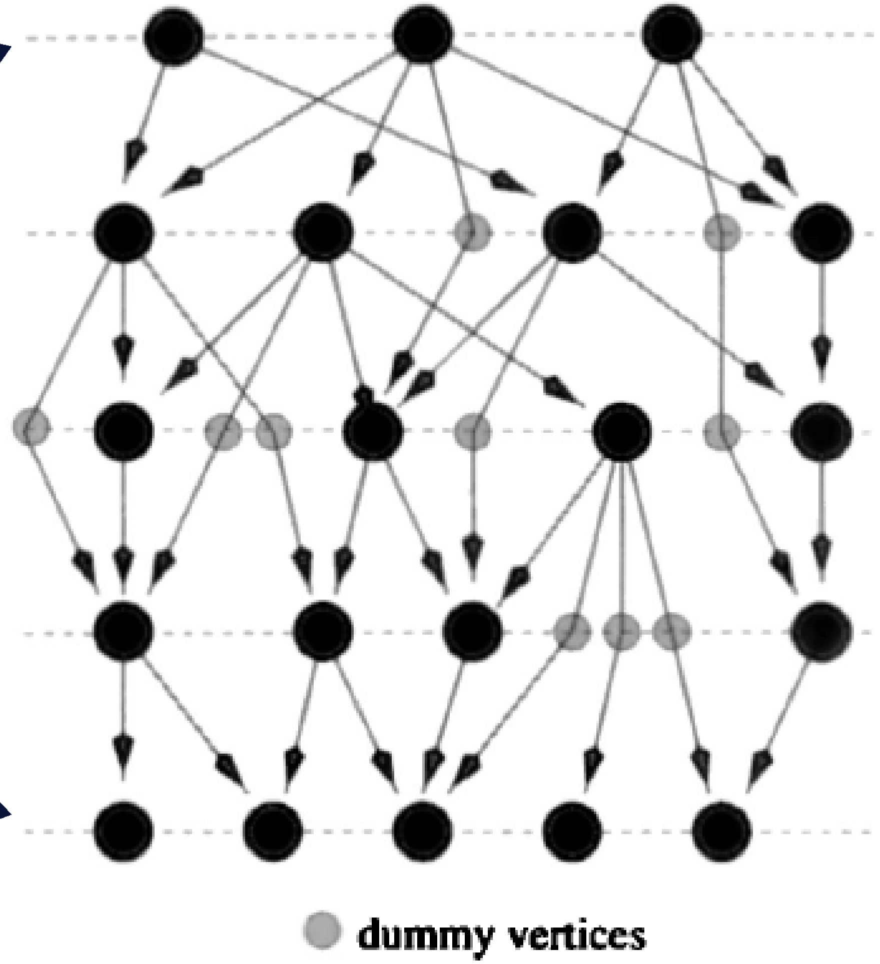
1. **Layer Assignment:** produces a hanging of the graph from its source node(s) -> assigns y-coordinates to each node
2. **Crossing Reduction:** determines an order of the nodes on each layer that minimizes edge crossings
3. **Horizontal Coordinate Assignment:** produces an x-coordinate for each node from the ordering

1. LAYER ASSIGNMENT

1st layer: source nodes (only outgoing edges)

dummy nodes introduced for edges spanning multiple levels

last layer: sink nodes (only incoming edges)



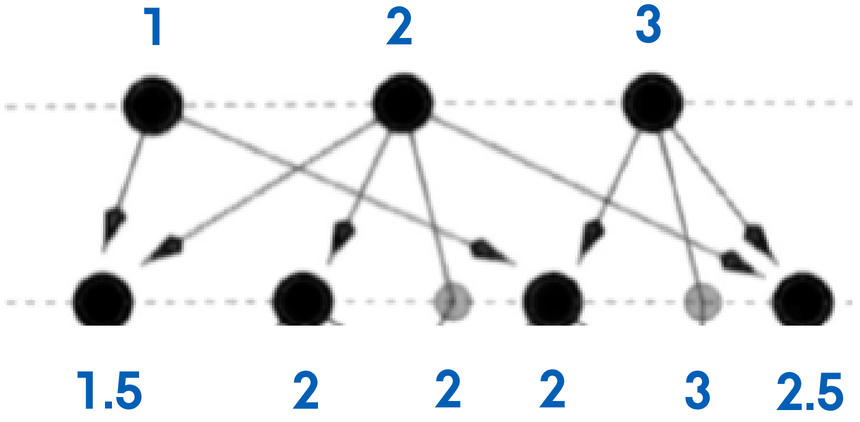
picture source: [di Battista et al. 1999]

2. CROSSING MINIMIZATION

Local Level-by-Level Optimization

L_i => i^{th} level, n => # of levels
for (i=1…n-1) **optimize** L_i w.r.t. L_{i-1}

e.g., using the **Barycenter Method** where a nodes position on level L_i is the mean of the positions of all connected nodes on level L_{i-1}

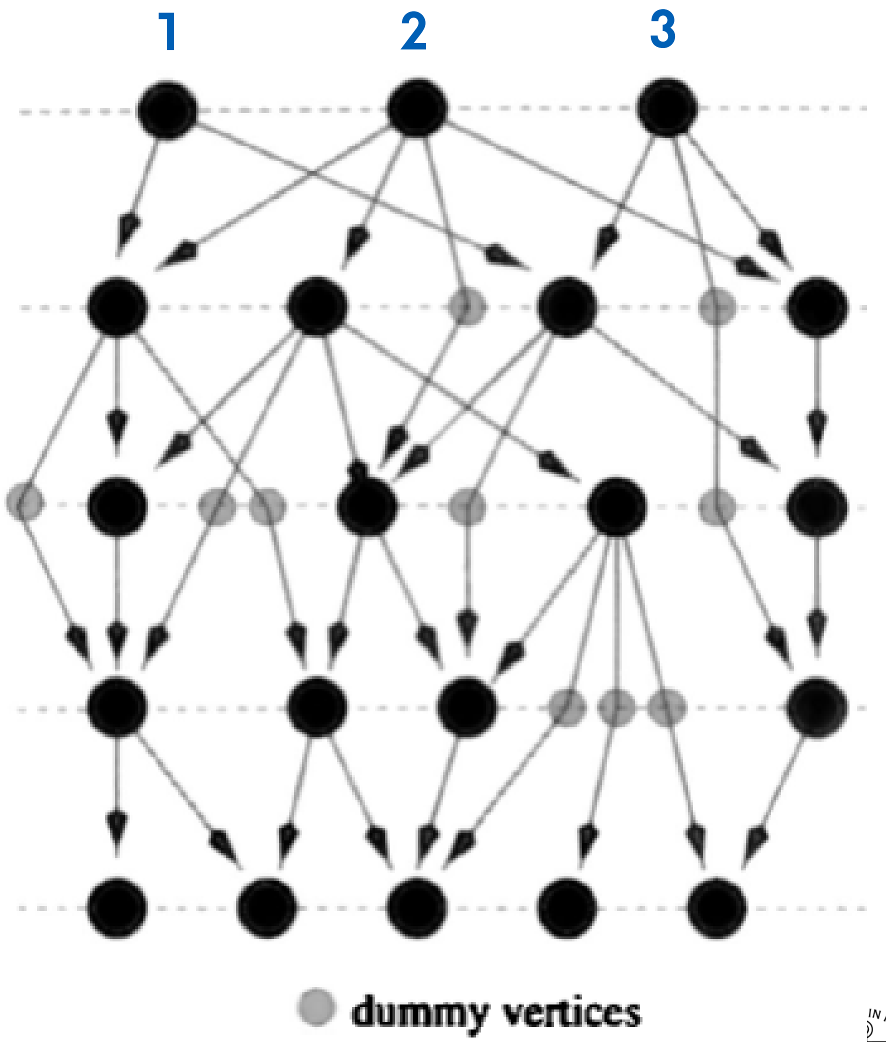


2. CROSSING MINIMIZATION

Local Level-by-Level Optimization

L_i => i^{th} level, n => # of levels
for (i=1…n-1) **optimize** L_i w.r.t. L_{i-1}

e.g., using the **Barycenter Method** where a nodes position on level L_i is the mean of the positions of all connected nodes on level L_{i-1}



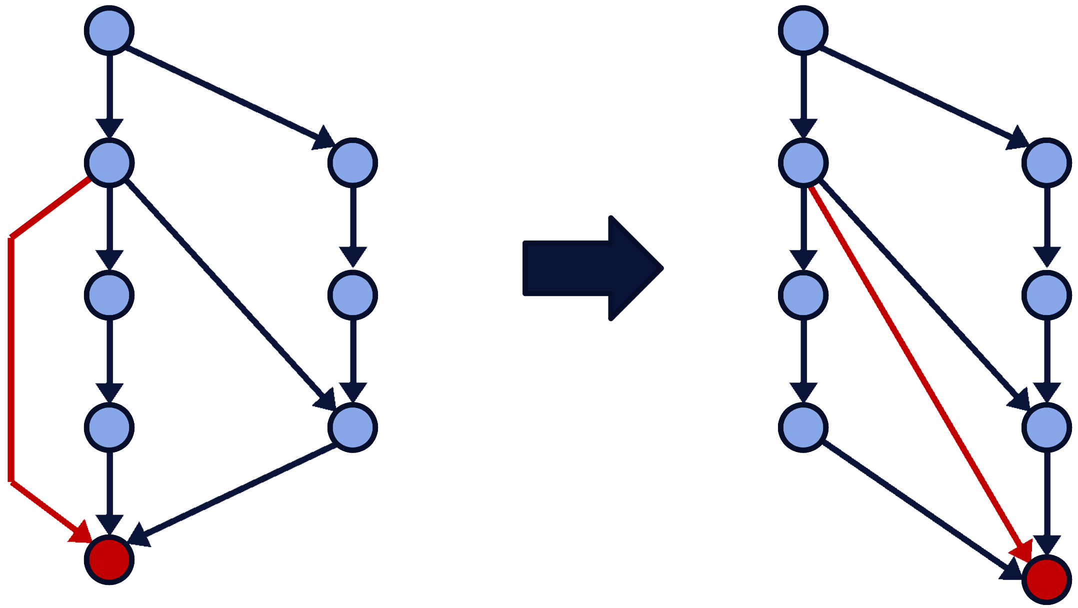
3. HORIZONTAL COORDINATE ASSIGNMENT

Bend Minimization

-> vertical lines are preferable to straight but diagonal lines

-> diagonal lines are preferable to bent lines

-> bent lines with fewer bends are preferable to those with more bends



LIST OF LITERATURE SOURCES (PART 1)

- Brandes 2001: https://doi.org/10.1007/3-540-44969-8_4
- Barnes & Hut 1986: https://doi.org/10.1038/324446a0
- Quigley & Eades 2000: https://doi.org/10.1007/3-540-44541-2_19
- Erten et al. 2004: https://doi.org/10.1117/12.539245
- Frishman & Tal 2007: https://doi.org/10.2312/VisSym/EuroVis07/075-082
- Gorochowski et al. 2012: https://doi.org/10.1109/TVCG.2011.142
- McDonnel & Elmqvist 2009: https://doi.org/10.1109/TVCG.2009.191
- Henry et al. 2007: https://doi.org/10.1109/TVCG.2007.70582
- Elmqvist et al. 2008: https://doi.org/10.1109/PACIFICVIS.2008.4475479
- Shen & Ma 2007: https://doi.org/10.2312/VisSym/EuroVis07/083-090
- v.Ham et al. 2009: https://doi.org/10.1007/978-3-642-03658-3_47
- Wu et al. 2008: https://doi.org/10.1007/978-3-540-33037-0_26
- Beinhauer & Ziegler 2003: https://www.researchgate.net/publication/267236592_Intuitive_Interaction_in_Complex_Information_Spaces_Results_and_Exploitation_of_INVITE
- Johnson & Shneiderman 1991: https://doi.org/10.1109/VISUAL.1991.175815

LIST OF LITERATURE SOURCES (PART 2)

- Bruls et al. 2000: https://doi.org/10.2312/VisSym/VisSym00/033-042
- Bederson & Shneiderman 2002: https://doi.org/10.1145/571647.571649
- Icicle Plot (1983): https://doi.org/10.2307/2685881
- Sunburst (2000): https://doi.org/10.1109/INFVIS.2000.885091
- Cascaded Treemaps (2008): http://portal.acm.org/citation.cfm?id=1375758
- Treevis.net: https://doi.org/10.1109/MCG.2011.103
- Sugiyama et al. 1981: https://doi.org/10.1109/TSMC.1981.4308636

SOLIDUM PETIT IN PROFUNDIS · UNIVERSITAS ARHUSIENSIS ·