Grooming the Hairball - How to Tidy up Network Visualizations?

Hans-Jörg Schulz[1], Christophe Hurter[2]

VIS Tutorial 2013

1. University of Rostock, Rostock, Germany
2. Ecole Nationale de l'Aviation Civile, Toulouse, France

PART 1:
NODE SET SIMPLIFICATION

Speaker: Hans-Jörg Schulz

Node Set Simplification

Why simplify the Node Set?

- Simplifies edge set at the same time
- Compatible with subsequent simplification of the edge set
- Simplifies even if no edges are given/drawn

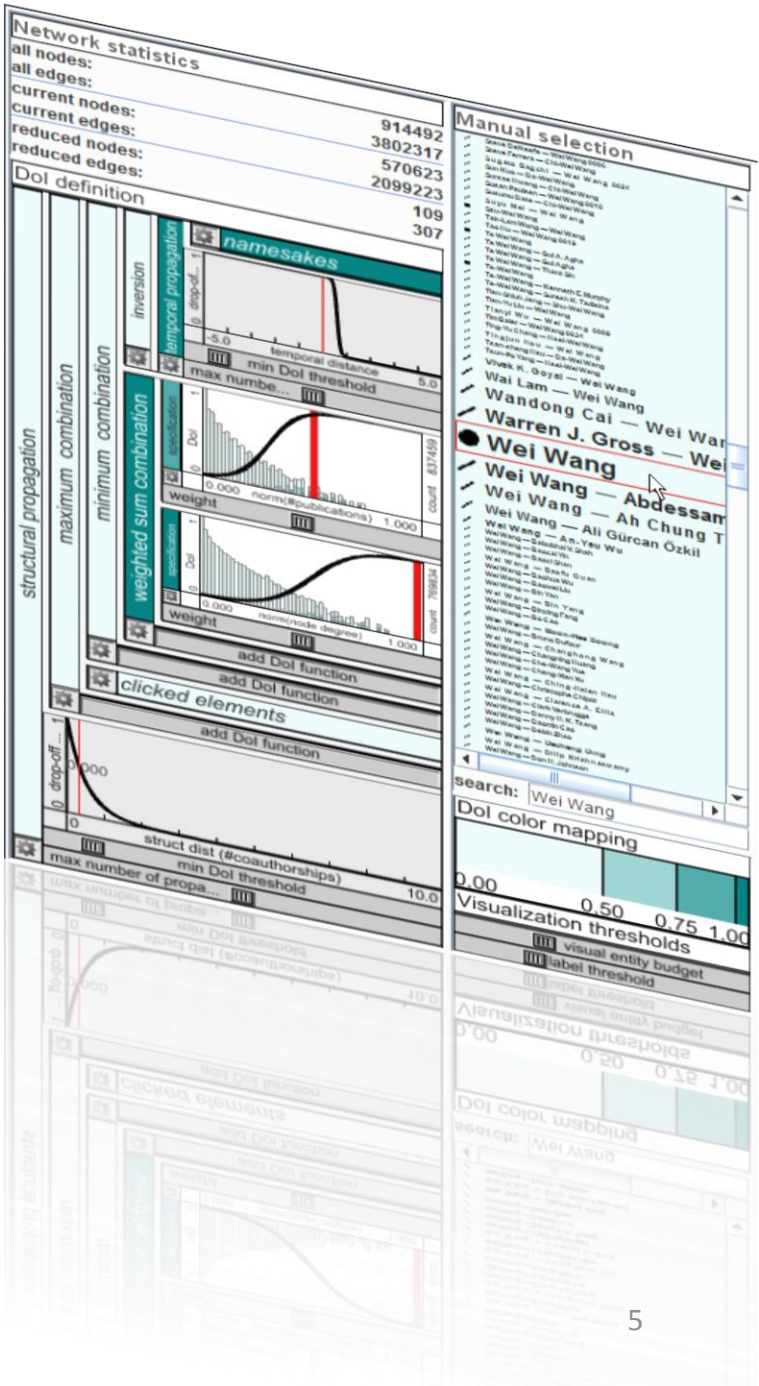
Objectives

At the end of Part I, you will be able to explain:

- the principles of **detecting** and **reducing** clutter by node set simplification
- at least one example for each principle and the **interactive means** to steer it
- Which simplification methods can be combined across **data level**, **geometry level**, and **image level**

Node Set Simplification on

DATA LEVEL (FILTERING)



Node Set Simplification on

Data Level (Filtering)

If a given network produces clutter, the only way to counter that on data level is by reducing the number of nodes. (or edges… cf. Part II)

There are two ways to perform such a reduction:

1. Indiscriminately – e.g., sampling
2. Selectively – e.g., filtering w.r.t. user interest

Focus

Indiscriminate Simplification on

Data Level (Filtering)

Most popular method: **Traversal-based Sampling**
(as it maintains connectivity)

- Breadth/Depth/Random First Sampling
- Snow-Ball Sampling
- Random Walk Sampling
- …

Further reading on sampling: [Hu+Lau 2013]
"A Survey and Taxonomy of Graph Sampling"

Selective Simplification on

Data Level (Filtering)

Challenge of the selective node set simplification: Find nodes that are expendable w.r.t. some given (or implied) criterion.

There are two kinds of expendable nodes:

a) **Uninteresting nodes**
Cardinality: individual (single) nodes
Example: singletons

b) **Redundant nodes**
Cardinality: group of (multiple) nodes
Example: similar nodes

Detecting

Uninteresting Nodes on Degree-of-Interest Measures

Data Level (Filtering)

The Traditional Monolithic DoI Specification

different DoI components:

- a-priori interestingness [Furnas 1986]
- distance to focus [Furnas 1986]
- user interest [van Ham + Perer 2009]
- navigation history [Gladisch et al. 2013]

Weighted Sum

Detecting

Uninteresting Nodes on Degree-of-Interest Measures

Data Level (Filtering)

A Modular Approach to DoI Specification

[Abello et al. 2014]

- specification components
  = *DoI generators* providing the base DoI values
- transformation components
  = *unary DoI operators* modifying a single given DoI value
- combination components
  = *n-ary DoI operators* (1 graph element, n DoI values) aggregating multiple DoI values defined for a single graph element
- propagation components
  = *n-ary DoI operators* (n graph elements, 1 DoI value) spreading a DoI value across neighboring graph elements

[slide courtesy of Steffen Hadlak]

Detecting

Uninteresting Nodes on Degree-of-Interest Measures

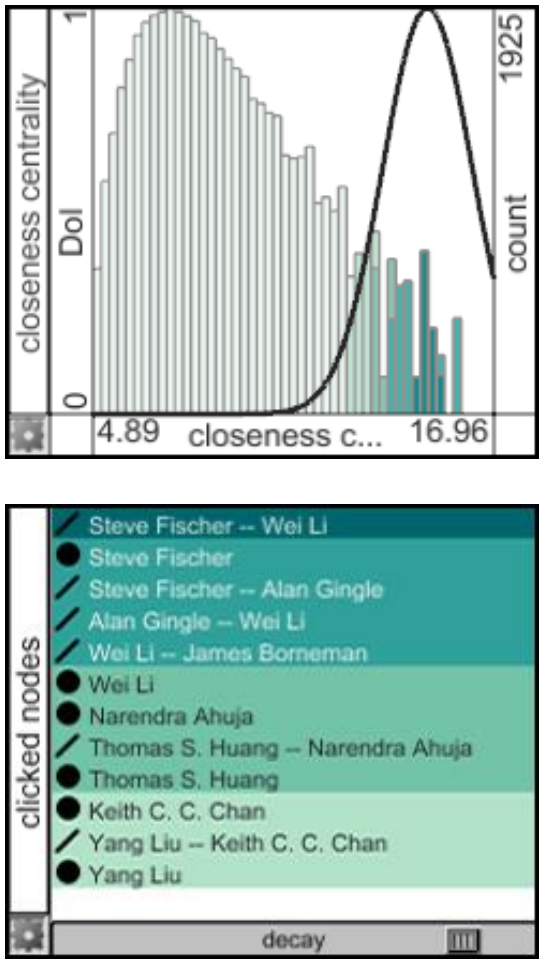
Data Level (Filtering)

Specification Components

$$spec(x_i) = inter(comp(x_i))$$

$$comp(x_i) : attr(x_i) \mapsto \mathbb{R}$$

$$inter : \mathbb{R} \mapsto [0 \ldots 1]$$



[slide courtesy of Steffen Hadlak]

Detecting

Uninteresting Nodes on Degree-of-Interest Measures

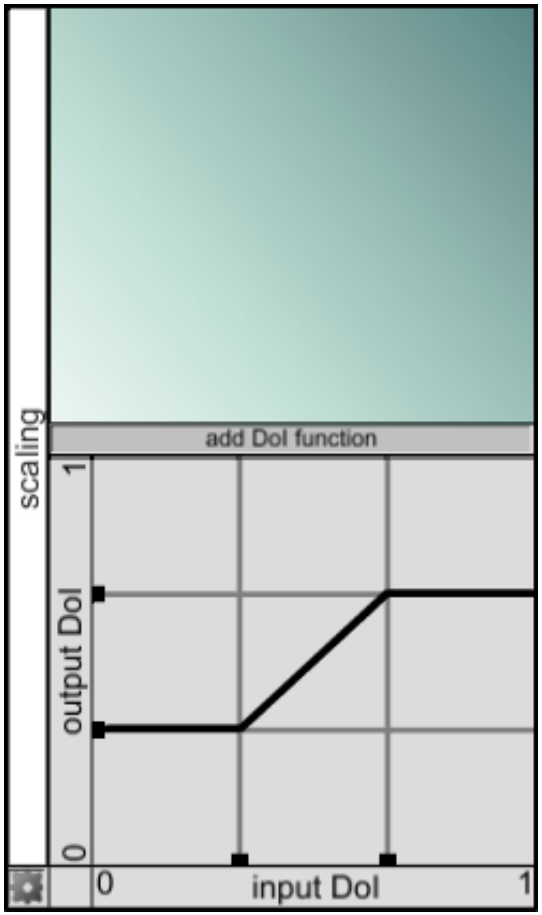
Data Level (Filtering)

Transformation Components

$$trans : [0 \ldots 1] \mapsto [0 \ldots 1]$$

$$inv(doi(x_i)) = 1 - doi(x_i)$$

$$scale_{const,exp}(doi(x_i)) = const * doi(x_i)^{exp}$$



[slide courtesy of Steffen Hadlak]

Detecting

Uninteresting Nodes on Degree-of-Interest Measures

Data Level (Filtering)

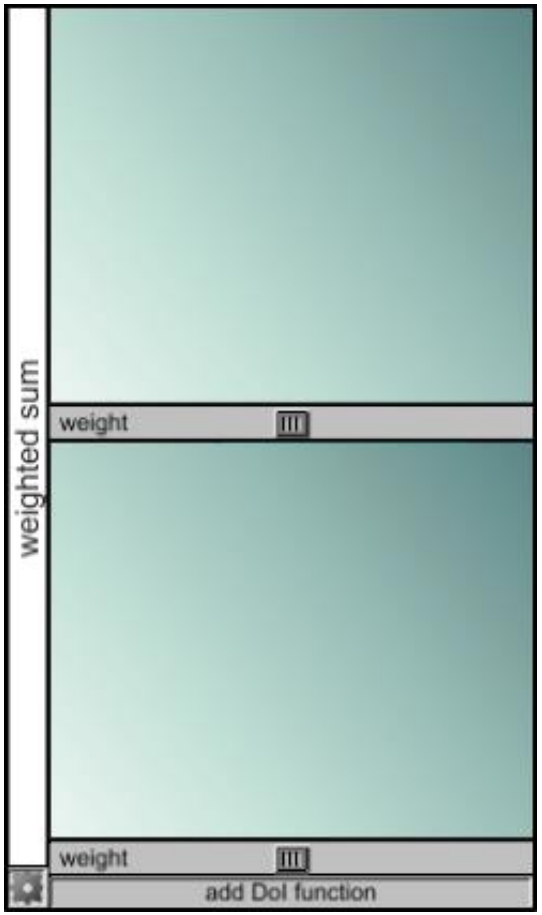
Combination Components

$$comb : [0 \ldots 1]^n \mapsto [0 \ldots 1]$$

$$max(DOI, x_i) = max_{doi_n \in DOI}(doi_n(x_i))$$

$$min(DOI, x_i) = min_{doi_n \in DOI}(doi_n(x_i))$$

$$sum(DOI, x_i) = \sum_{doi_n \in DOI}(w_n \cdot (doi_n(x_i))$$



[slide courtesy of Steffen Hadlak]

Detecting

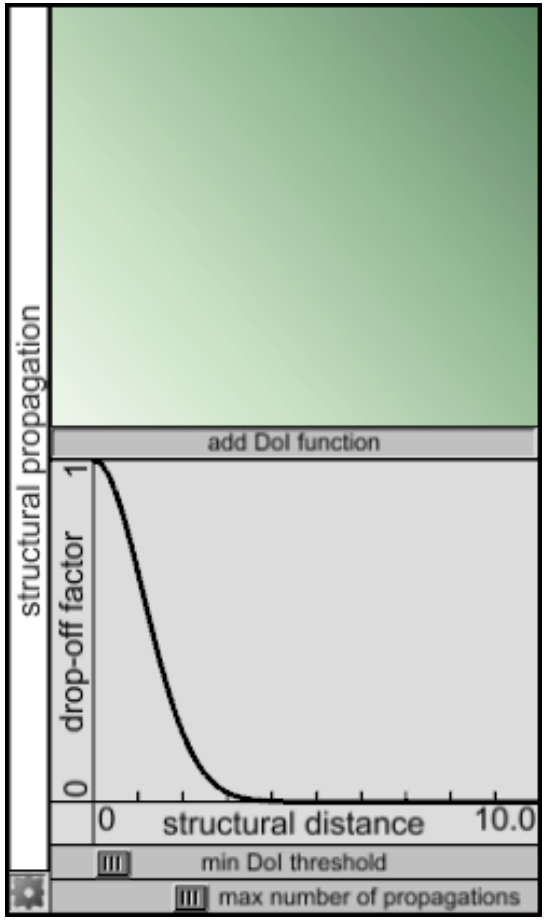
Uninteresting Nodes on Degree-of-Interest Measures

Data Level (Filtering)

Propagation Components

$$prop_s(doi(y_i), x_i) = \max_{y_i \in V_i \cup E_i} drop(doi(y_i), dist_{attr}(x_i, y_i))$$

$$prop_t(doi(x_j), x_i) = \max_{j=1}^{|T|} drop(doi(x_j), t_j - t_i)$$



[slide courtesy of Steffen Hadlak]

Detecting

Uninteresting Nodes on Degree-of-Interest Measures

Data Level (Filtering)

A Modular Furnas-like DoI

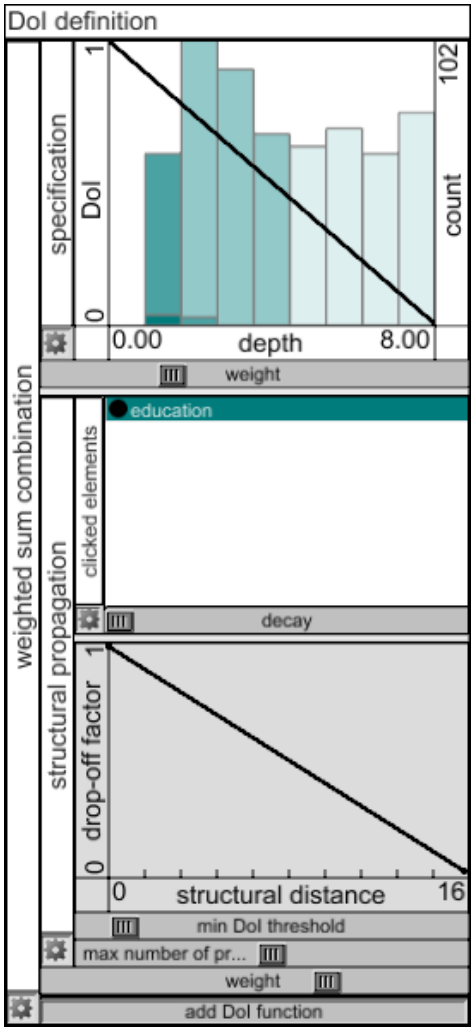
$$DOI(x|y) = -dist(x, root) - dist(x, y)$$

$$doi(x) = 1/3 * inter(height(x)) + 2/3 * prop_s(select_0(y), x)$$

$$inter(height(x)) = 1 - height(x)/max\_height$$

$$dist_{attr}(x, y) = gd(x, y) \text{ with } attr(x) = 1$$

$$drop(select_0(y), dist_1(x, y)) = 1 - dist_1(x, y)/(2 * max\_height)$$



[image courtesy of Steffen Hadlak]

Detecting

Redundant Nodes on Similarity + Clustering

Data Level (Filtering)

Similarity can be defined in a number of spaces:

- **Topological space**
  e.g., graph-theoretic distance of nodes
- **Attribute space**
  e.g., Euclidean distance of attribute vectors
- **Temporal space**
  e.g., length of time span separating nodes

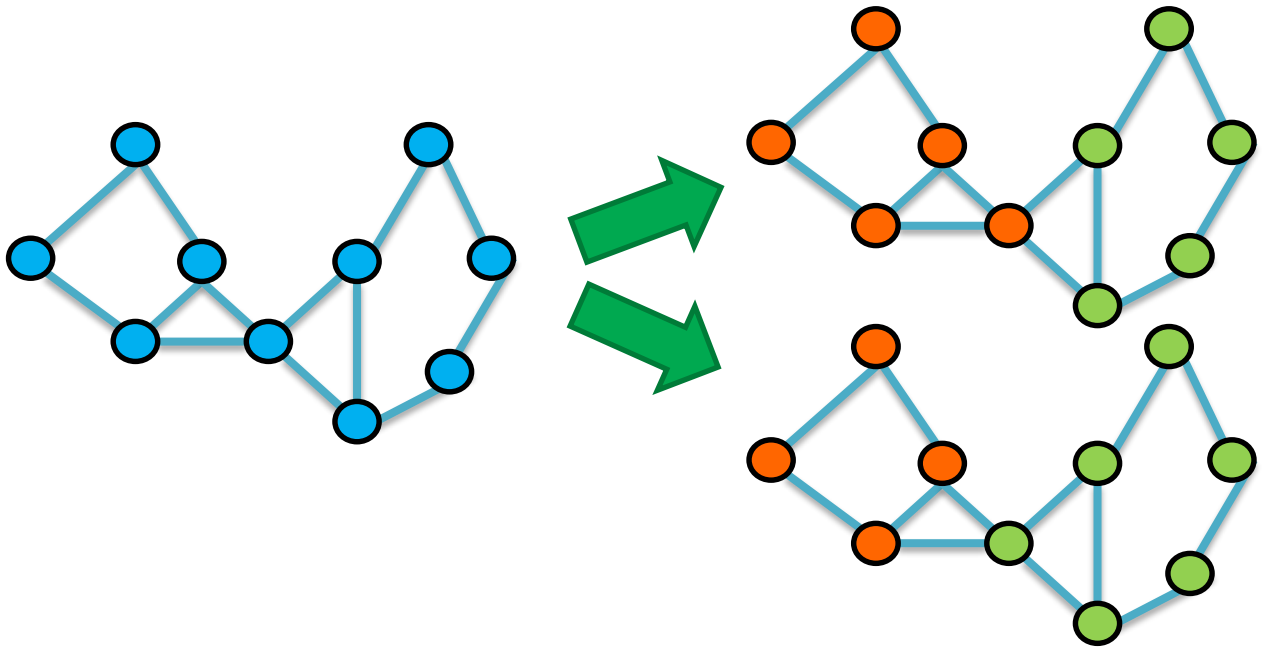
Detecting

Redundant Nodes on Similarity + Clustering

Data Level (Filtering)

The principles of clustering are assumed as known.

Clustering challenges that arise in networks relate to its runtime complexity and semantics.

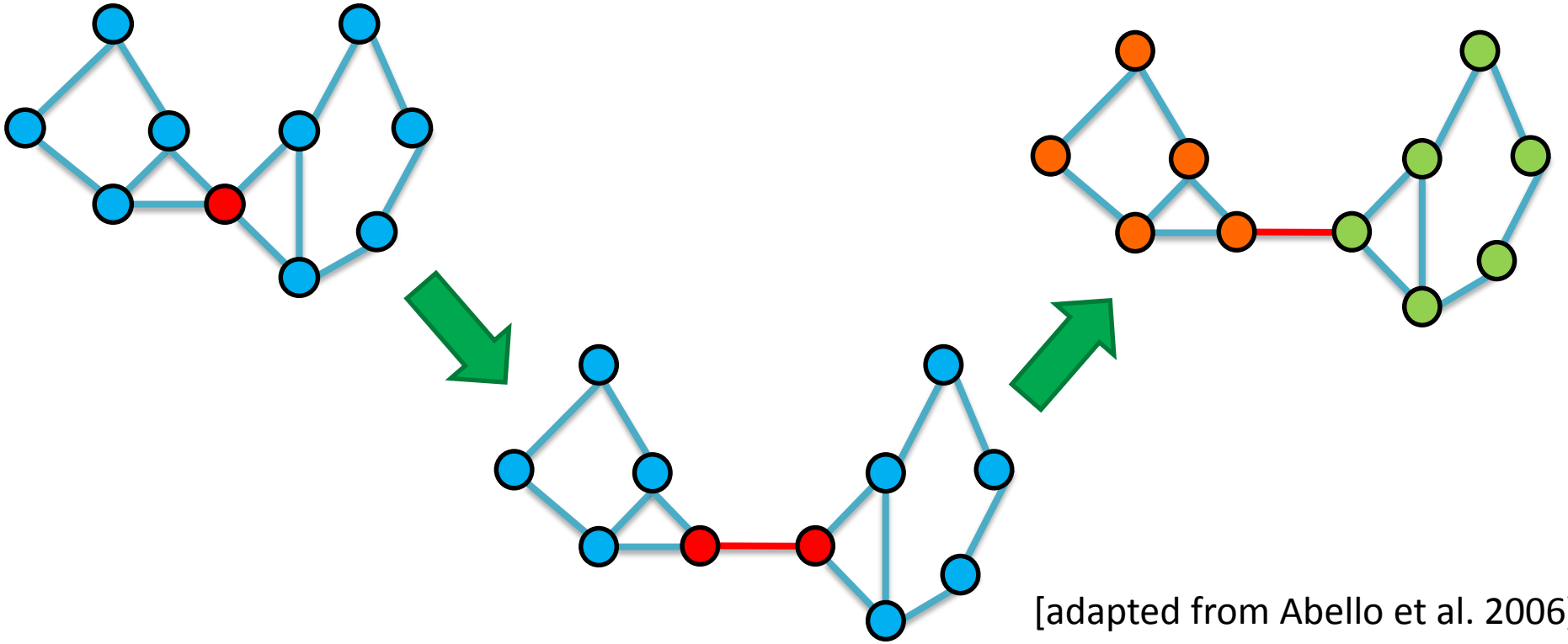


Detecting

Redundant Nodes on Similarity + Clustering

Data Level (Filtering)

Some simple preprocessing can resolve such clustering problems: e.g., **cloning nodes** to clarify semantics



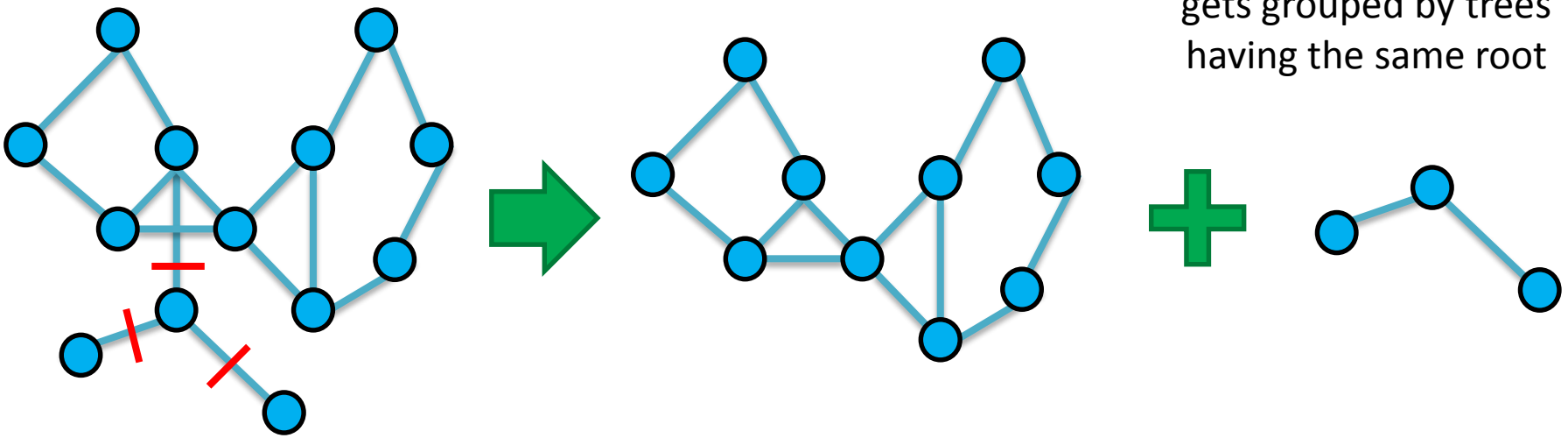
[adapted from Abello et al. 2006]

Detecting

Redundant Nodes on Similarity + Clustering

Data Level (Filtering)

Some simple preprocessing can resolve such clustering problems: e.g., **peeling** to reduce runtime



Peripheral Forest
gets grouped by trees having the same root

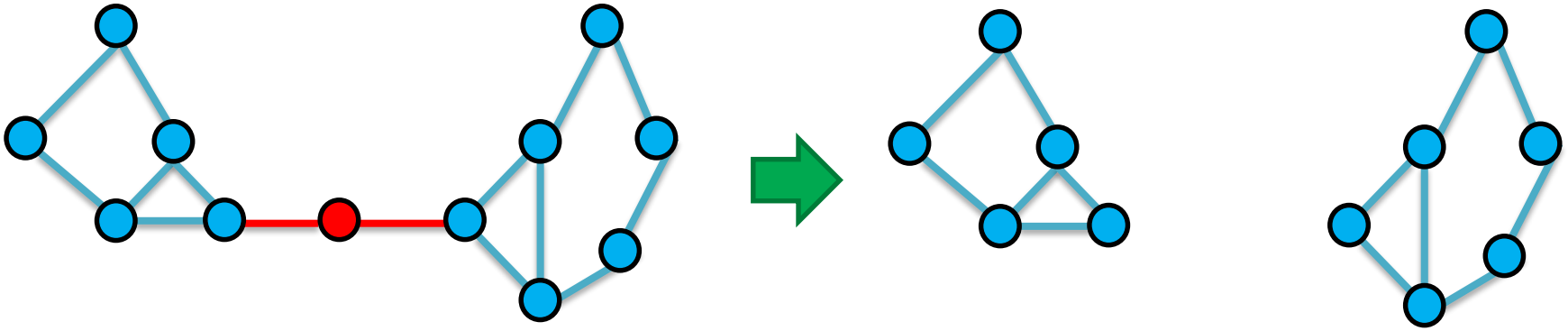
[adapted from Abello et al. 2006]

Reducing

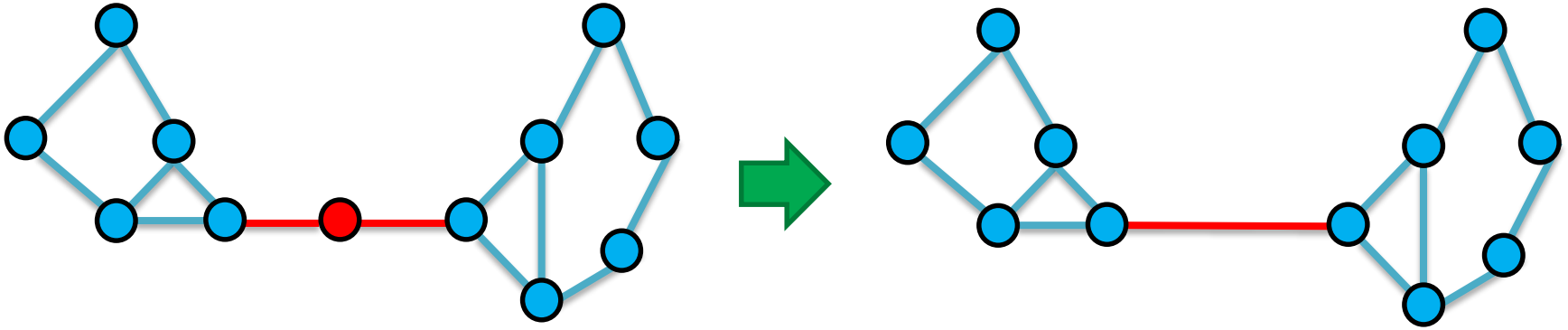
Uninteresting Nodes on Node Contraction

Data Level (Filtering)

Node removal can disrupt network topology:



Node contraction maintains network topology:

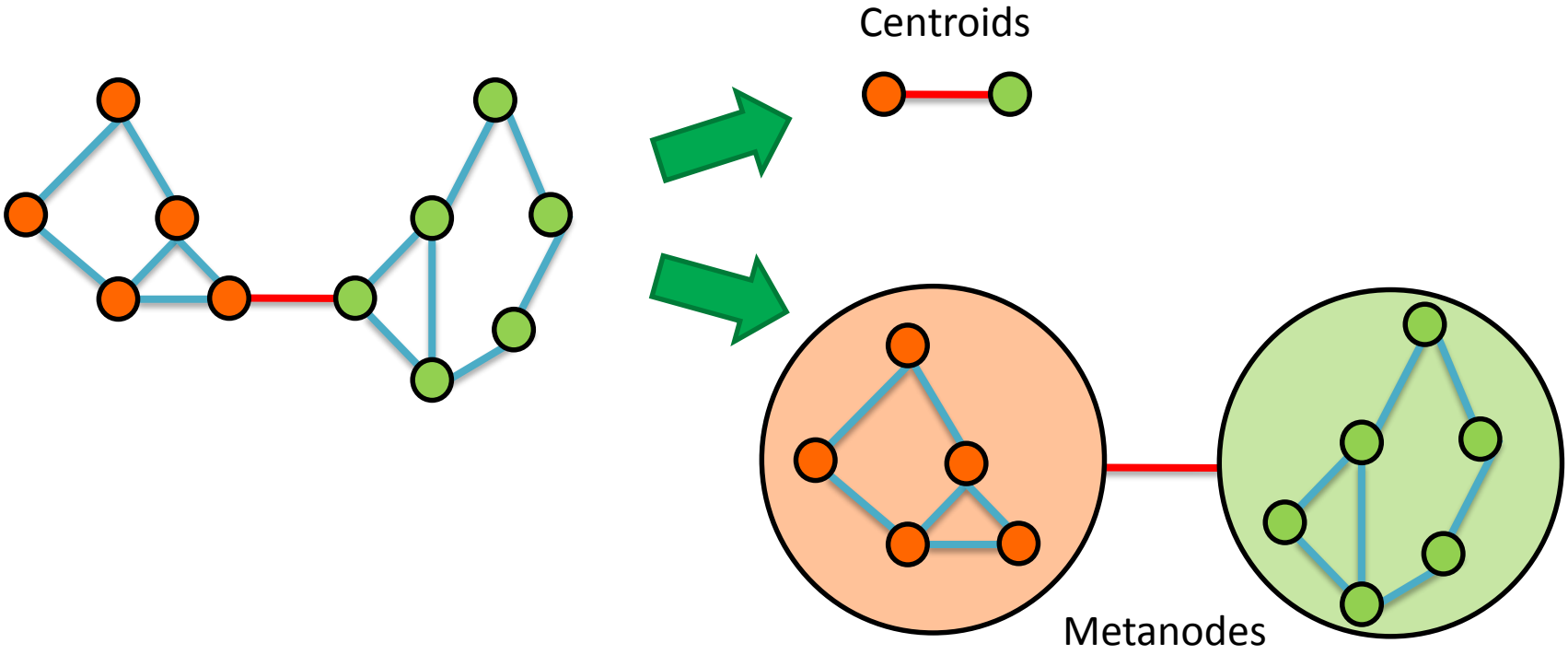


Reducing

Redundant Nodes on Node Set Substitution

Data Level (Filtering)

Substitute each Cluster with a Cluster Representative



Interacting

with Reduced Node Sets on Thresholding

Data Level (Filtering)

Adjust the removal of nodes by adjusting the **Degree-of-Interest Threshold**:

- **directly:** set a threshold below which nodes are removed
- **indirectly:** set a number *N* of nodes that shall remain, pick the *N* nodes with highest DoI values, remove the rest

Interacting

with Reduced Node Sets on Thresholding

Data Level (Filtering)

Adjust the clustering of nodes by adjusting the **Similarity Threshold**:

- **directly:** set a threshold above which nodes are clustered
- **indirectly:** set a number *N* of nodes/clusters that shall remain and cluster nodes with decreasing similarity values until *N* are left

Interacting

with Reduced Node Sets on Thresholding

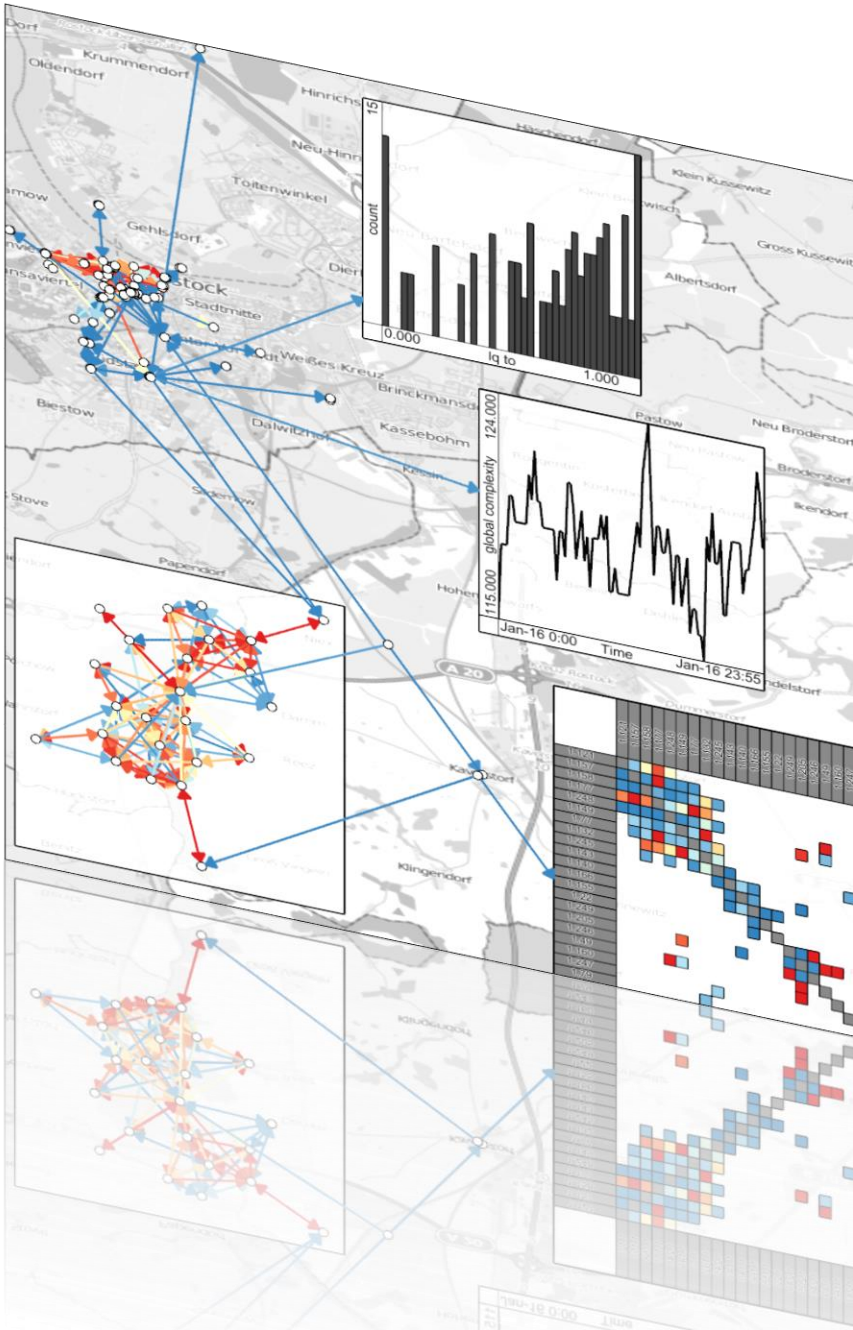
Data Level (Filtering)

Adjust the clustering of nodes by adjusting the **Similarity Threshold…**

vanHam2004 - Small World Graphs.avi

Node Set Simplification on

GEOMETRY LEVEL (MAPPING)



Node Set Simplification on

Geometry Level (Mapping)

Two fundamental approaches:

1. **Layout Generation:** React to a large node set during the layout generation
2. **Layout Readjustment:** Detect clutter/dense areas in a given layout and simplify it accordingly

Generating Layouts on

Geometry Level (Mapping)

Make nodes very small and place them very close…

Example: Point-based Tree Layout

[Schulz et al. 2011]

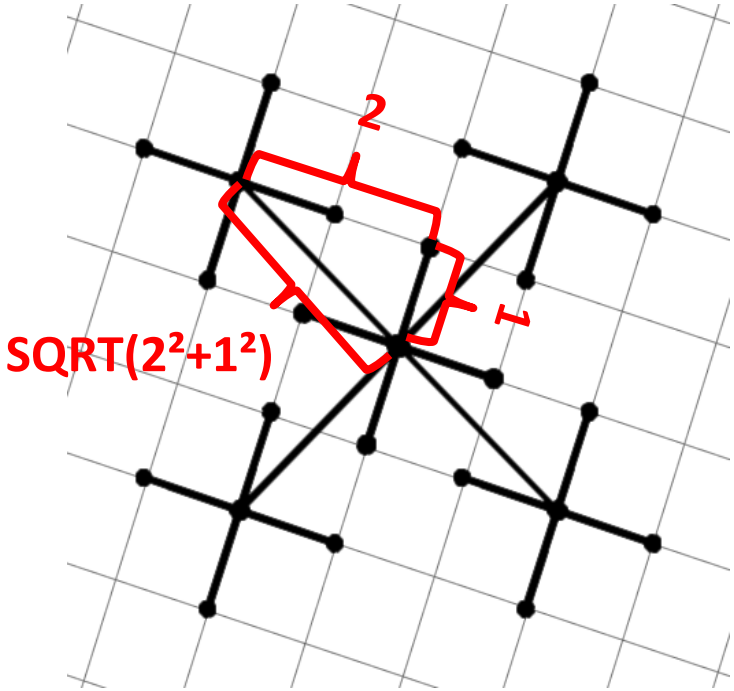
Schulz2009 - Pointbased (PacificVis).avi

Generating Layouts on

Geometry Level (Mapping)

Why rotation by 27°? – Finally the answer…

The Setup:

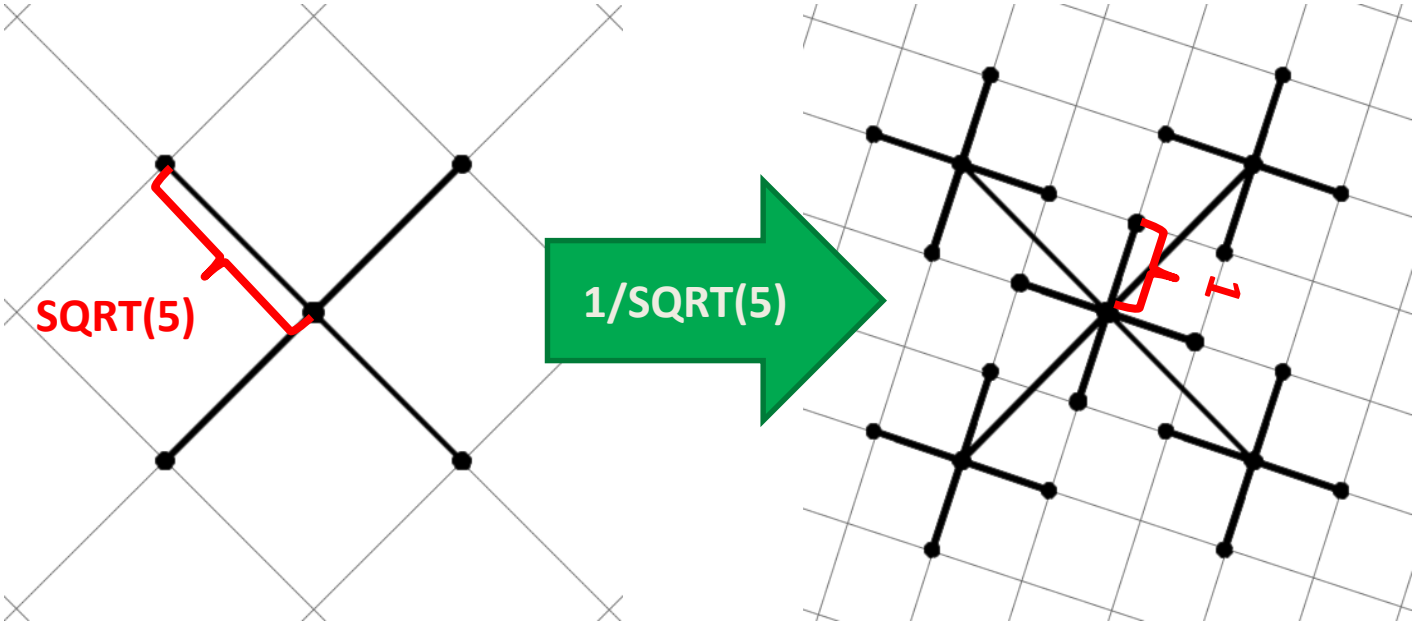


Generating Layouts on

Geometry Level (Mapping)

Why rotation by 27°? – Finally the answer…

The Scaling:



Generating Layouts on

Geometry Level (Mapping)

Why rotation by 27°? – Finally the answer…

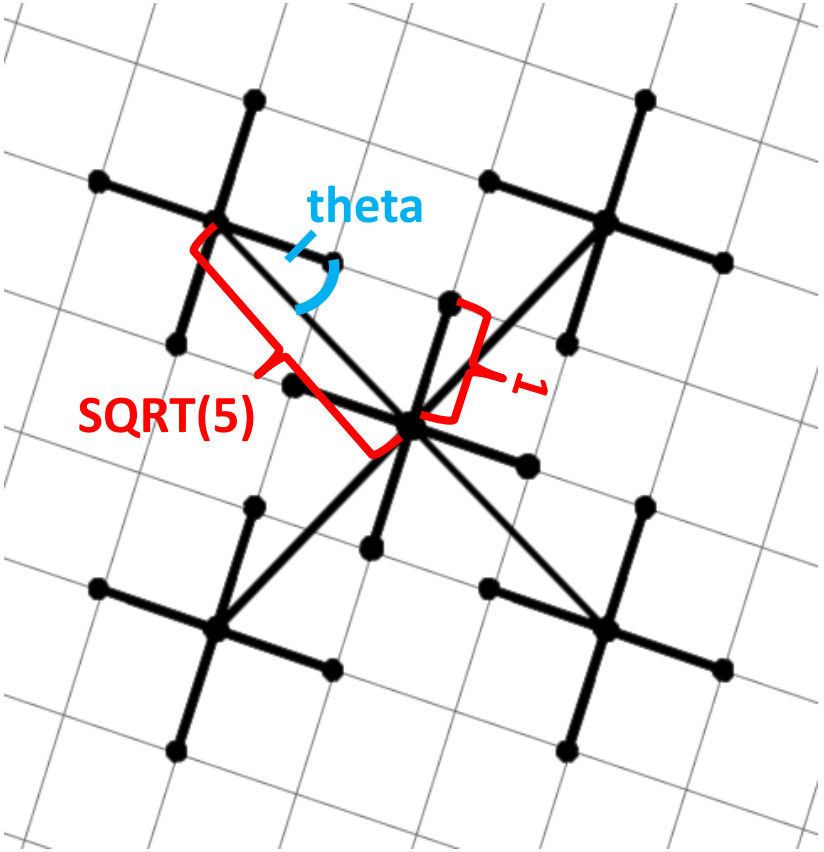
The Rotation:

$$\sin(\theta) = \frac{\text{opposite side}}{\text{hypotenuse}}$$

$\sin(\theta) = 1 / \text{SQRT}(5)$

$\theta = \arcsin(1 / \text{SQRT}(5))$

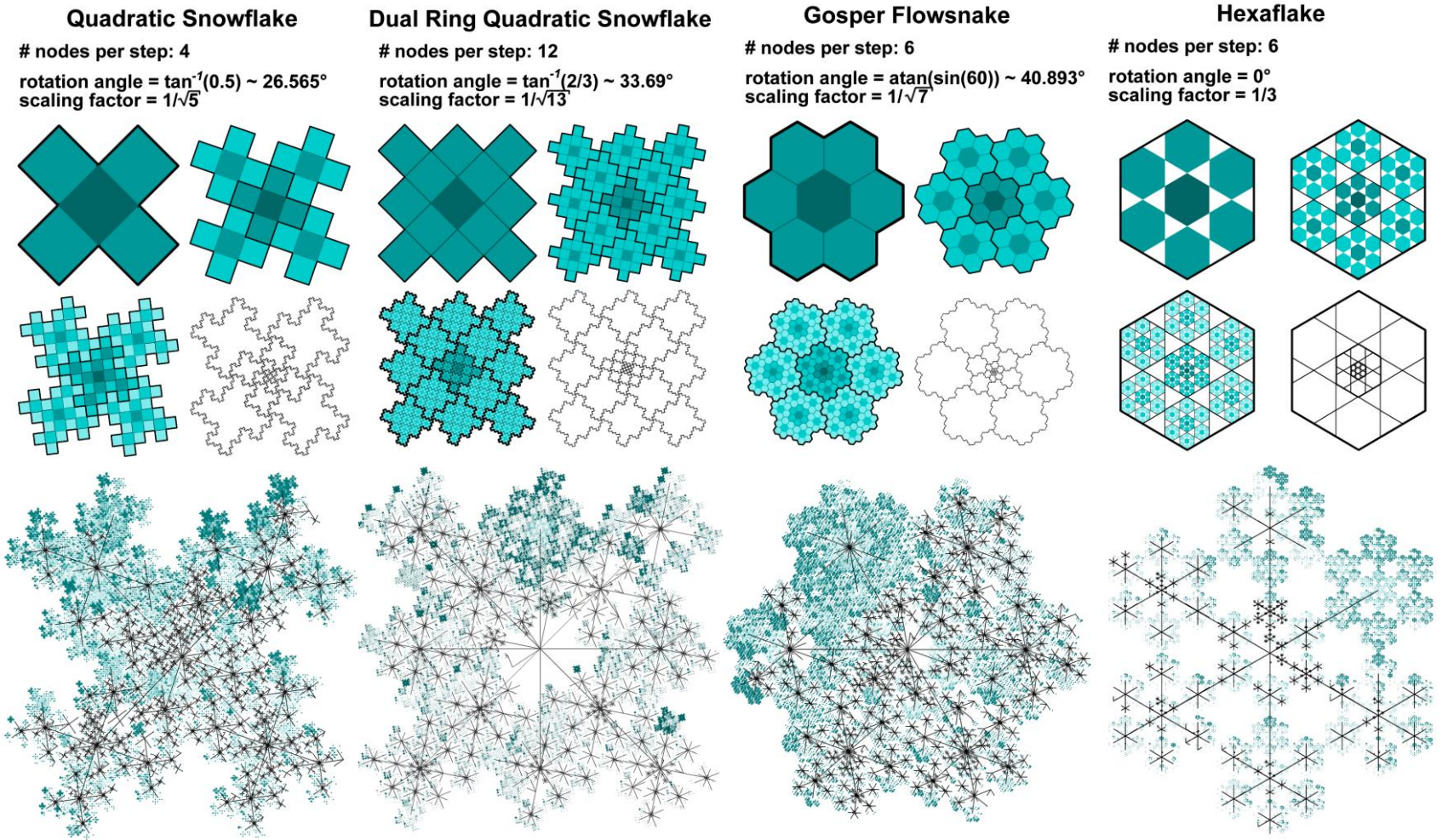
$\theta \approx 27°$



Generating Layouts on

Geometry Level (Mapping)

This also works for other scaling factors/angles:

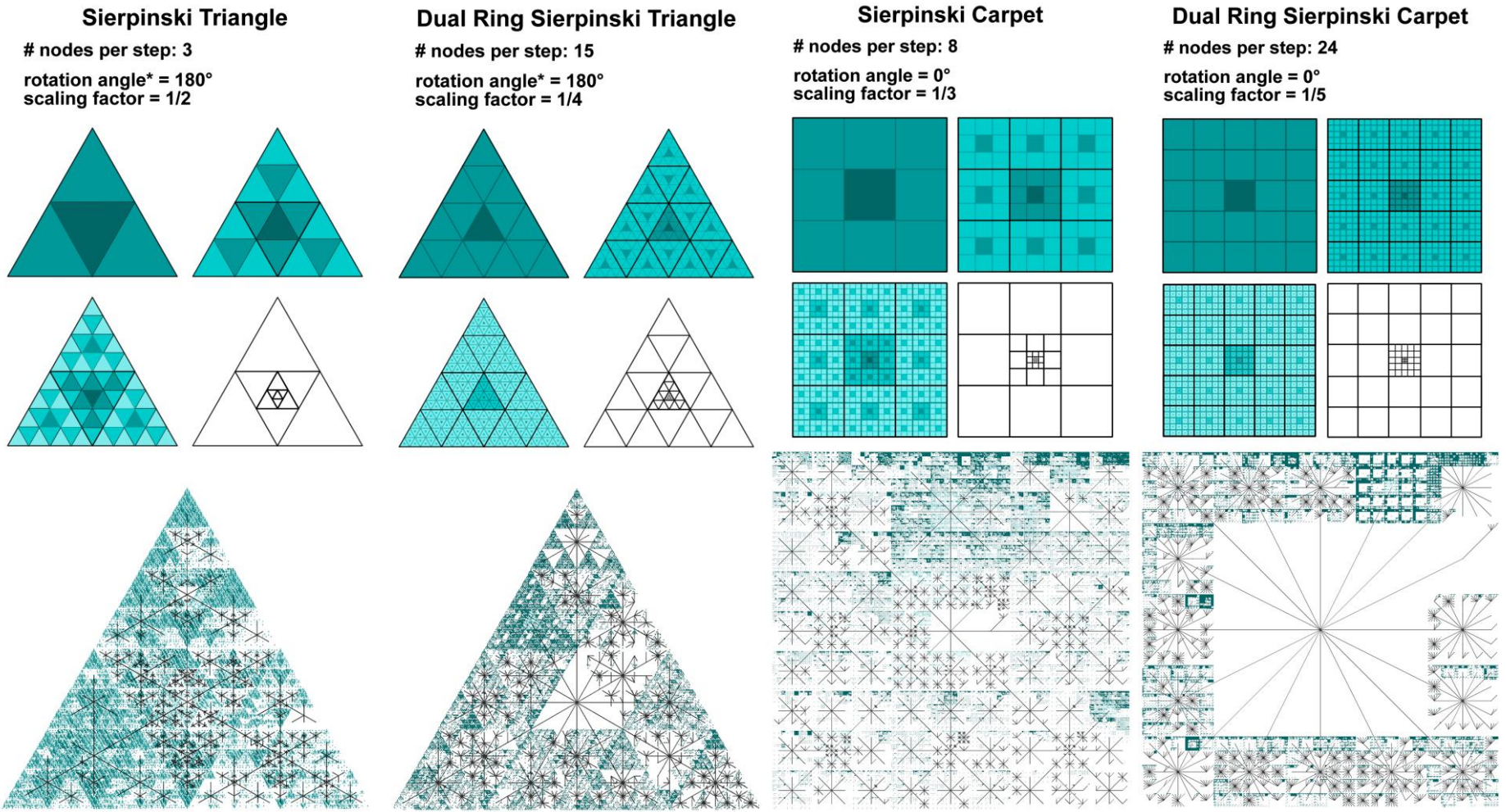


[images courtesy of Steffen Hadlak]

Generating Layouts on

Geometry Level (Mapping)

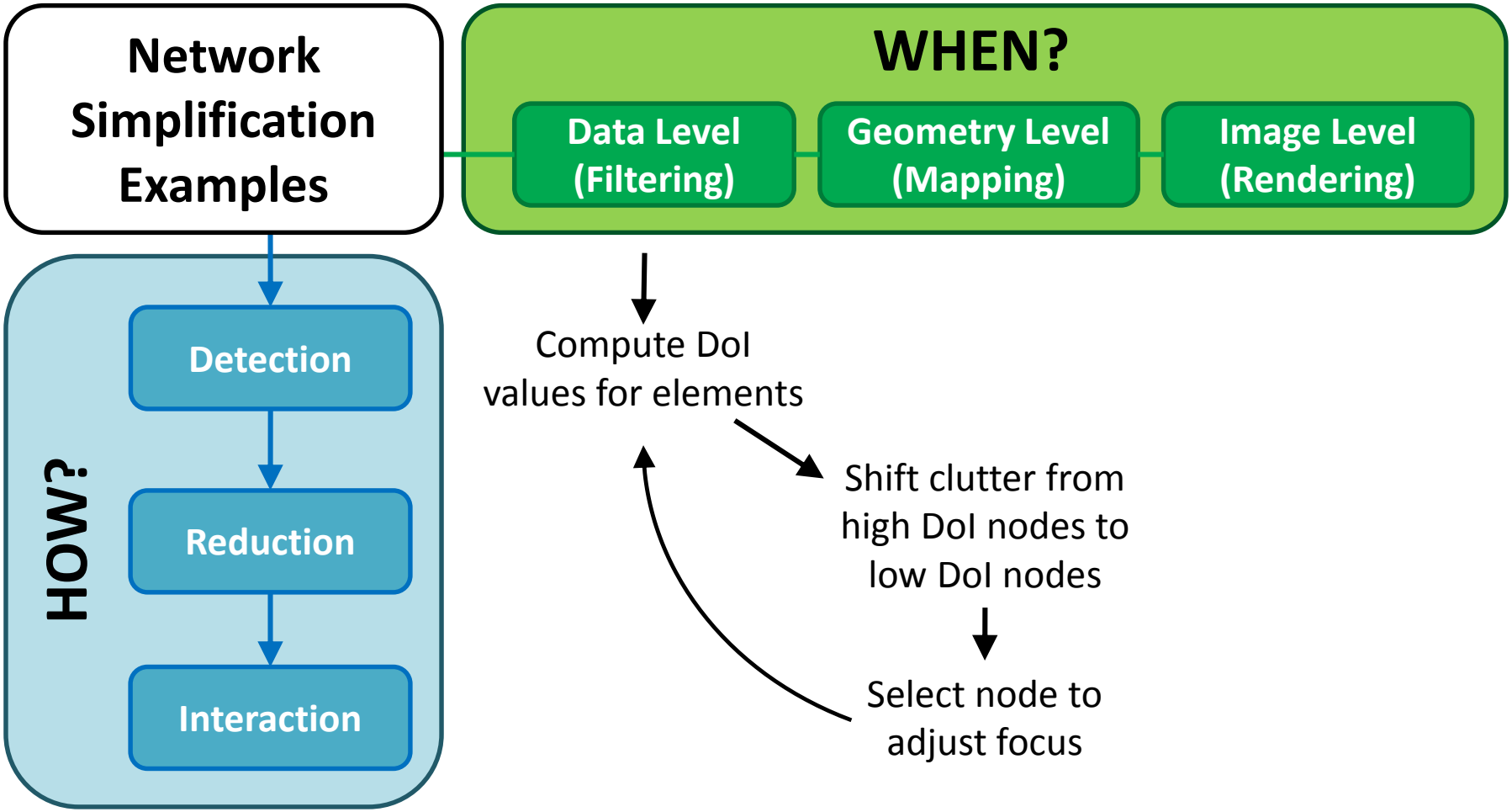
This also works for other scaling factors:



[images courtesy of Steffen Hadlak]

Generating Layouts on DoI-based Layout

Geometry Level (Mapping)



Generating Layouts on DoI-based Layout

Geometry Level (Mapping)

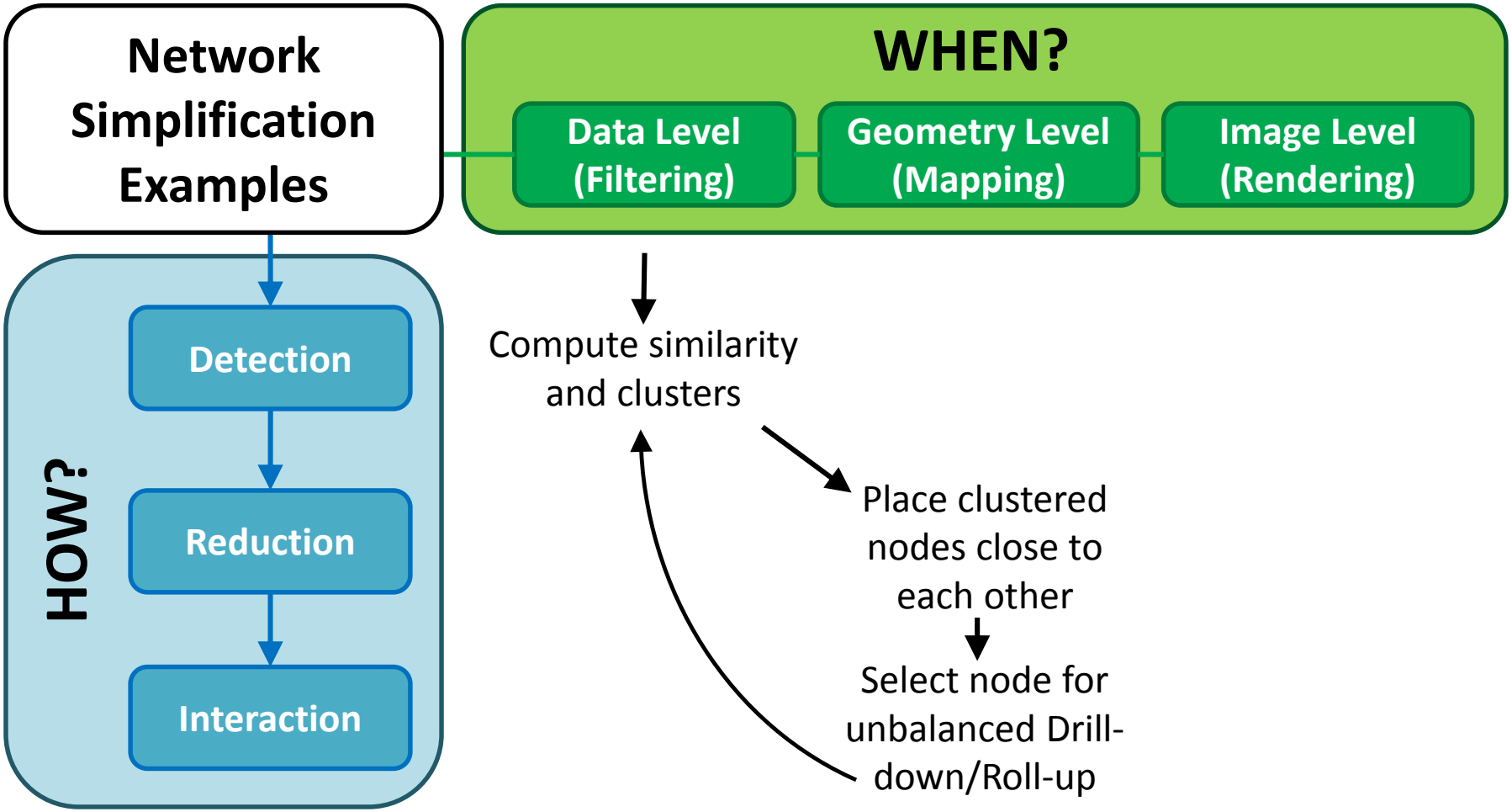
Shifting is done by using the DoI values as forces in a force-directed layout.

But there are many more uses for DoI values:

- Pinning weights in a dynamic layout
- Labeling priority
- Saturation of color-coding
- …and even Pseudohaptics

Generating Layouts on Similarity/Cluster-based Layout

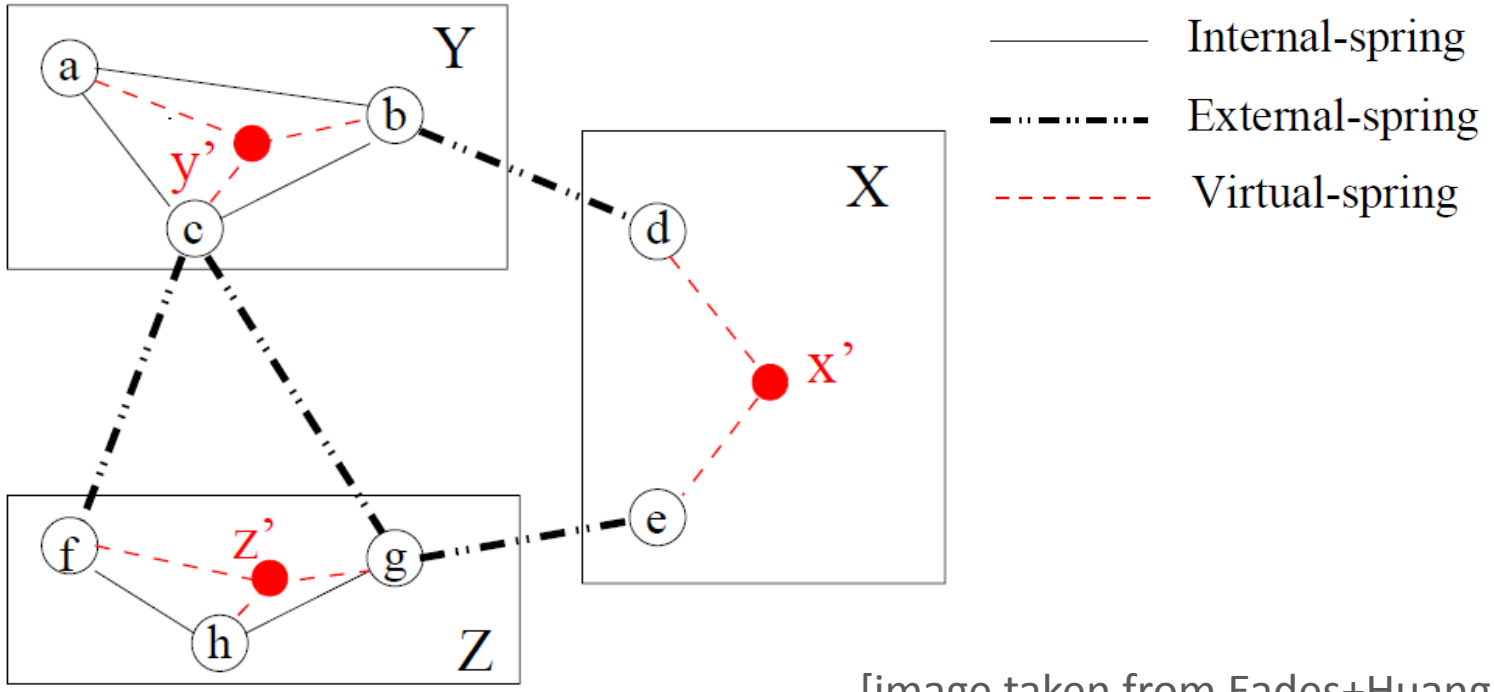
Geometry Level (Mapping)



Generating Layouts on Similarity/Cluster-based Layout

Geometry Level (Mapping)

Placing clustered nodes close to each other can be achieved by a multilevel layout approach:

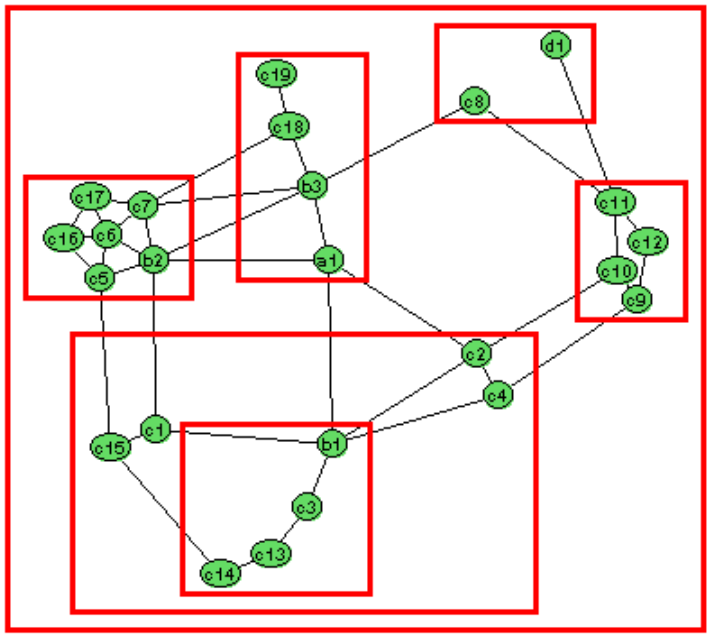


[image taken from Eades+Huang 2000]

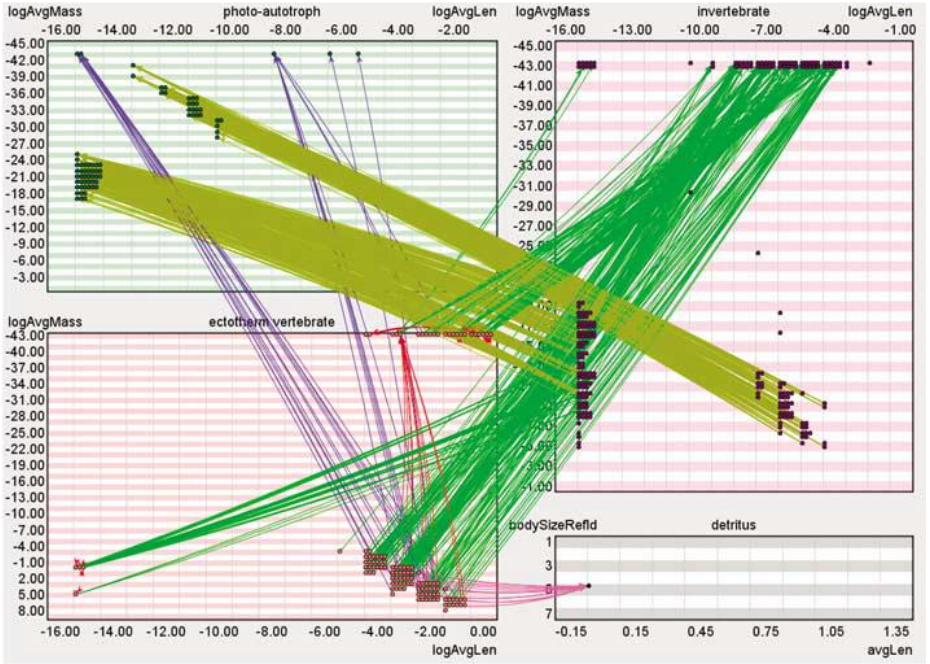
Generating Layouts on Similarity/Cluster-based Layout

Geometry Level (Mapping)

Note: The layout should focus on the node aspect according to which the network was clustered!



[Eades+Huang 2000]



[Aris+Shneiderman 2007]

Readjusted Layout on

Geometry Level (Mapping)

If nodes produce clutter in a layout, there are three ways to counter that on geometry level:

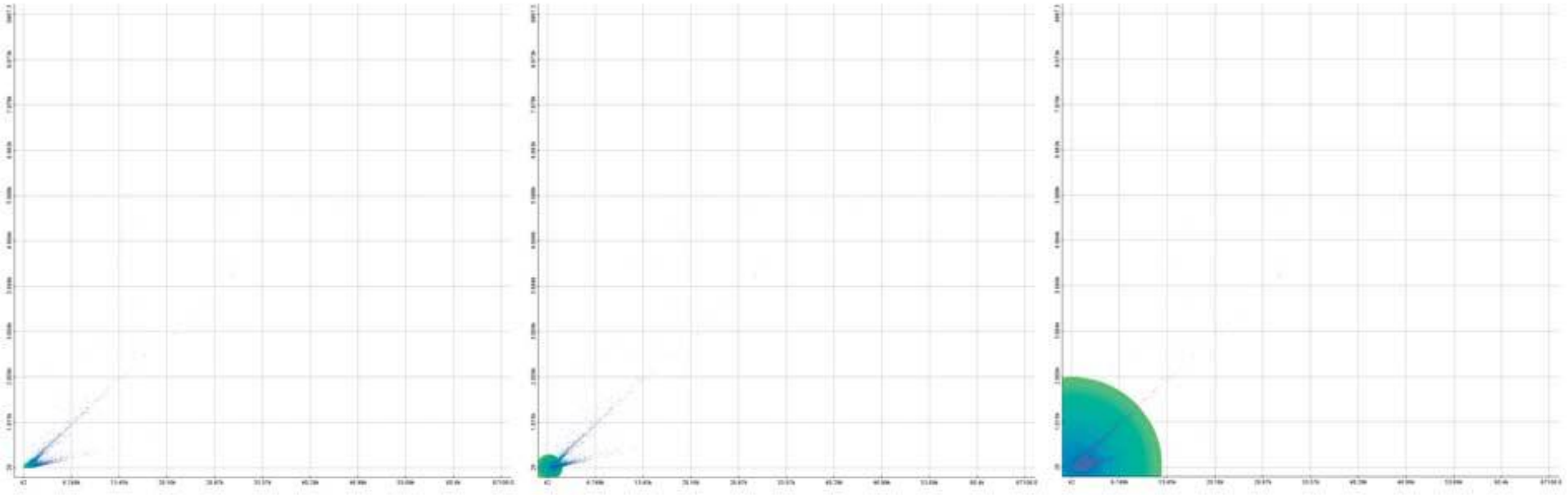
1. Distribute the nodes – e.g., overlap removal
2. Remove the nodes – e.g., layout coarsening
3. Abstract the nodes – e.g., use of metanodes

Readjusting Layouts on

Geometry Level (Mapping)

Distribution: Generalized Scatter Plots [Keim et al. 2010]

⟶ **displacement** + distortion

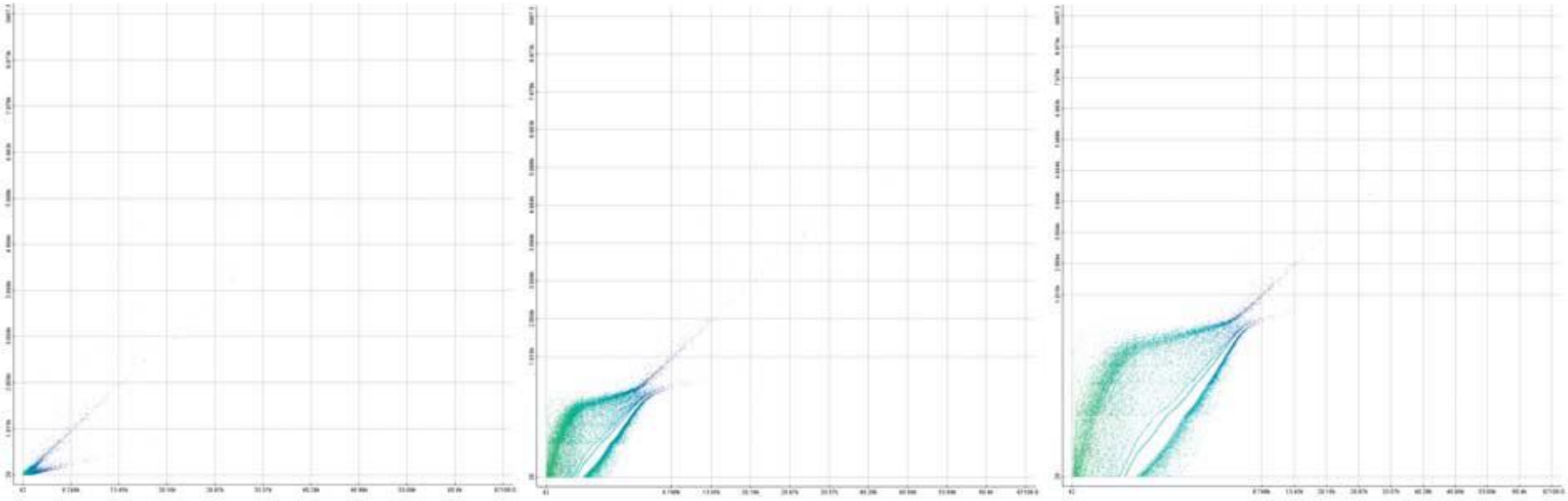


Readjusting Layouts on

Geometry Level (Mapping)

Distribution: Generalized Scatter Plots [Keim et al. 2010]

⟶ displacement + **distortion**

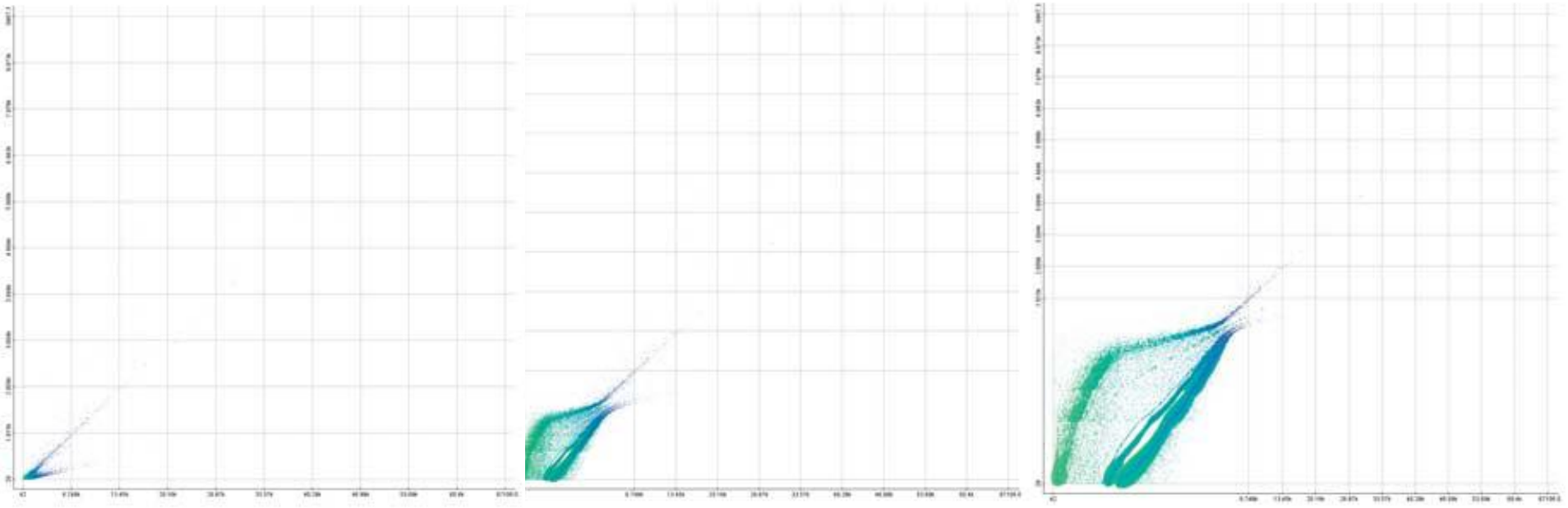


Readjusting Layouts on

Geometry Level (Mapping)

Distribution: Generalized Scatter Plots [Keim et al. 2010]

→ **displacement** + **distortion**

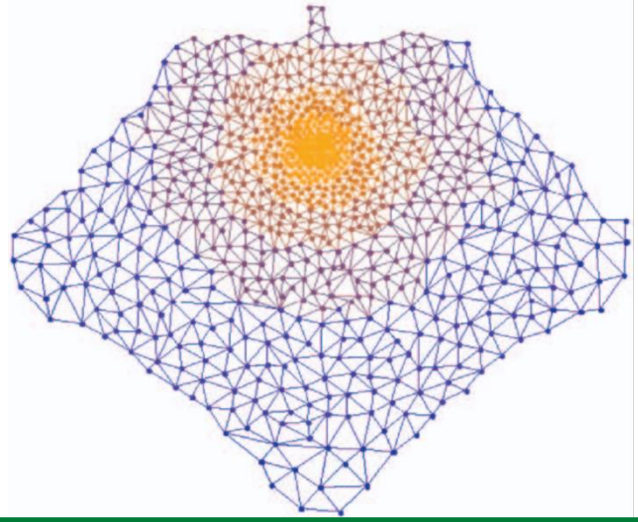


Readjusting Layouts on

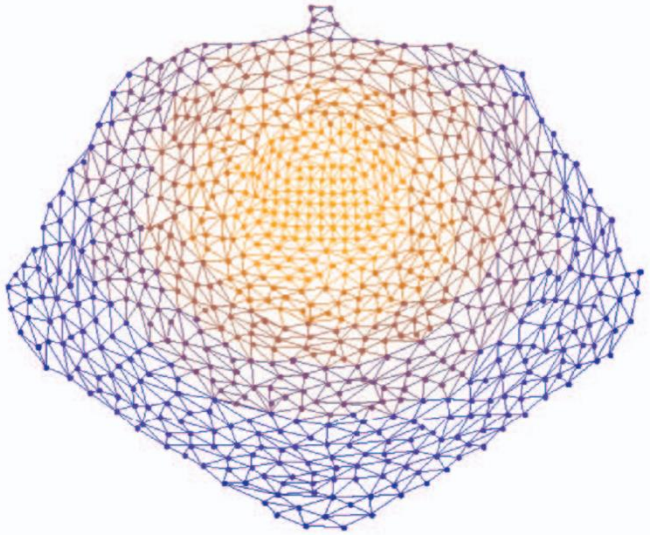
Geometry Level (Mapping)

Coarsening: Topological Fisheye [Ganser et al. 2005]

Original:
Crack Graph
V= 10240
E= 30380



LOD Coarsening:
fine-grained at focus node,
coarse-grained the further away

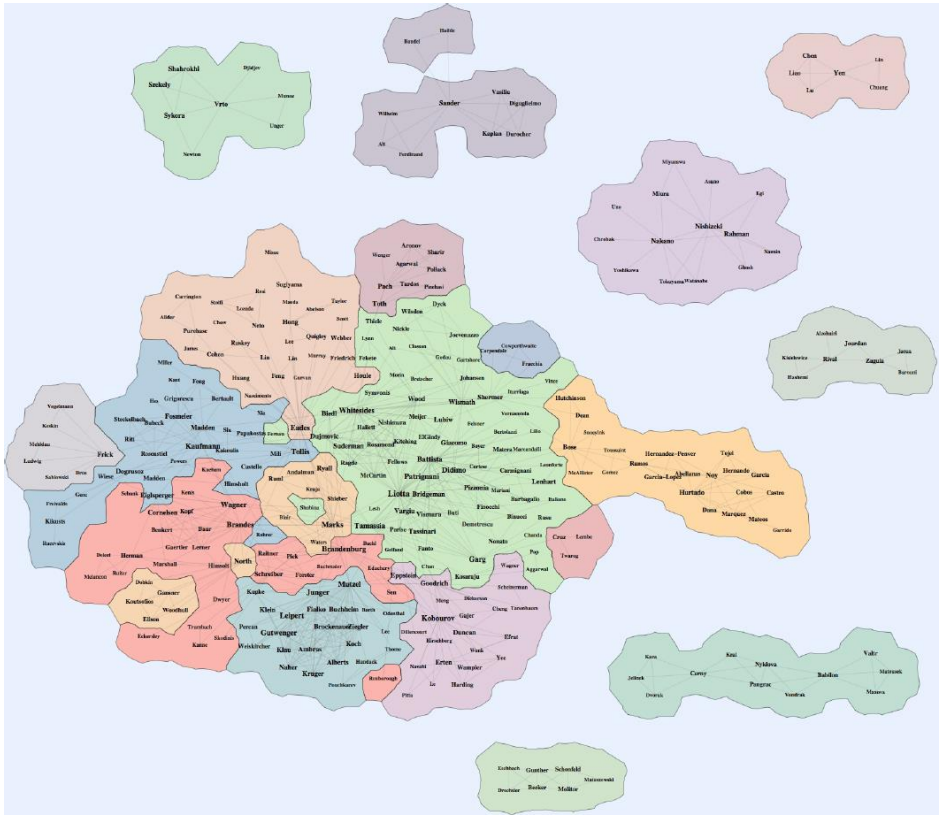


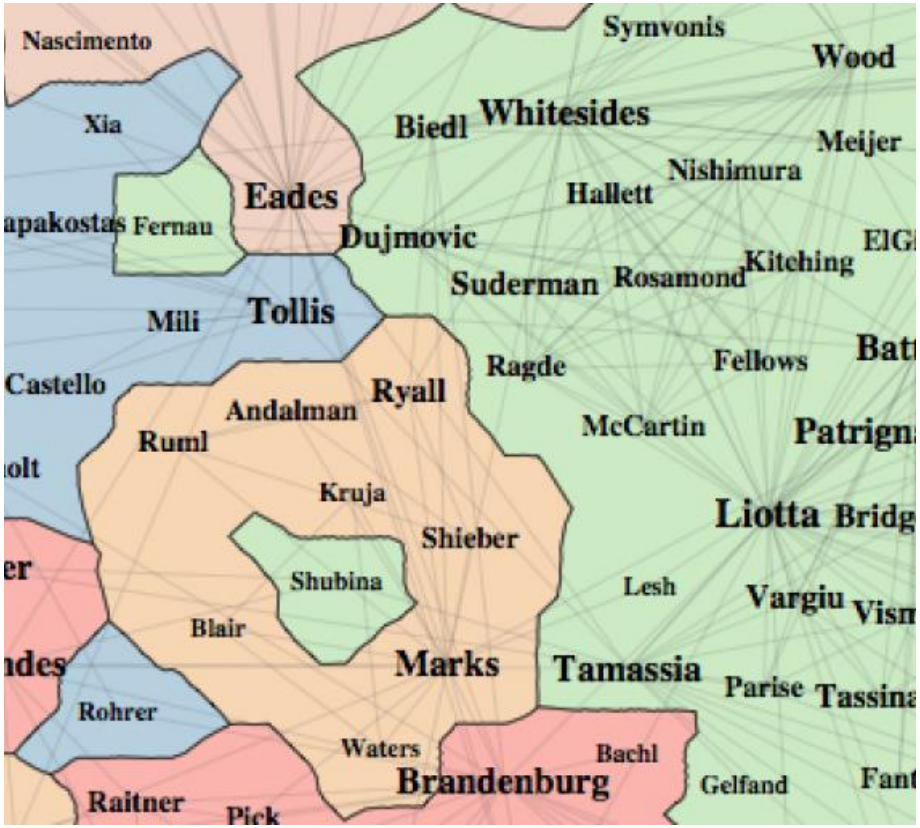
Radial Distortion
to achieve uniform
node density

Readjusting Layouts on Cluster-based Readjustment

Geometry Level (Mapping)

2D Abstraction: Gmap [Gansner et al. 2010]

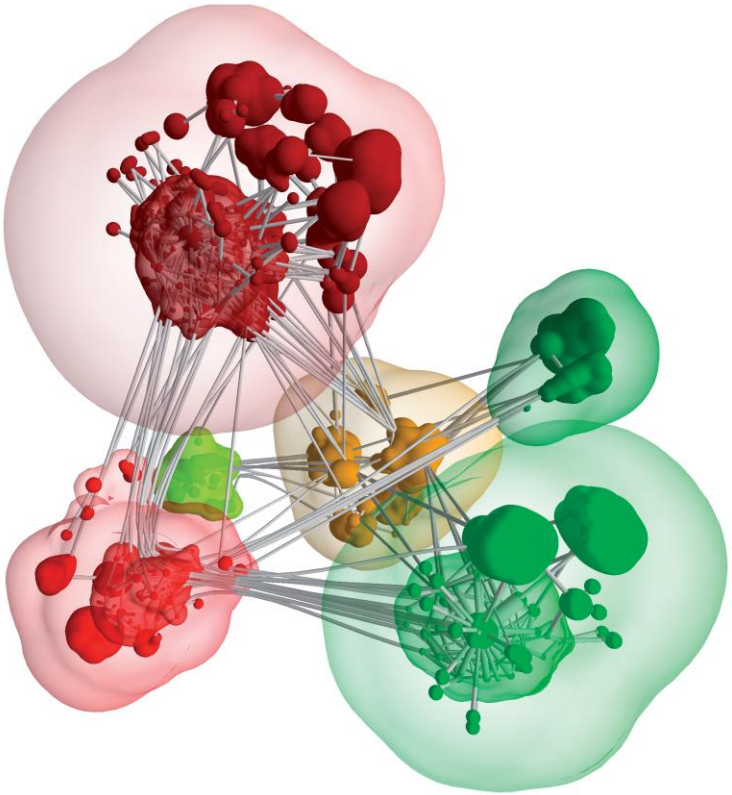


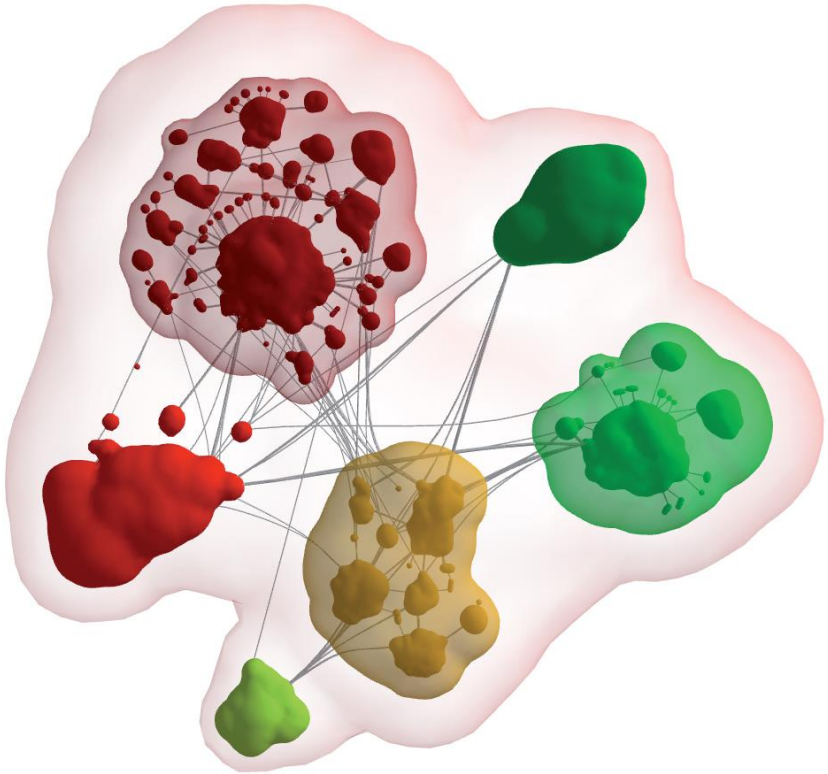


Readjusting Layouts on Cluster-based Readjustment

Geometry Level (Mapping)

3D Abstraction: Graph Cluster Surfaces [Balzer+Deussen 2007]





Readjusting Layouts on Interactive Readjustment

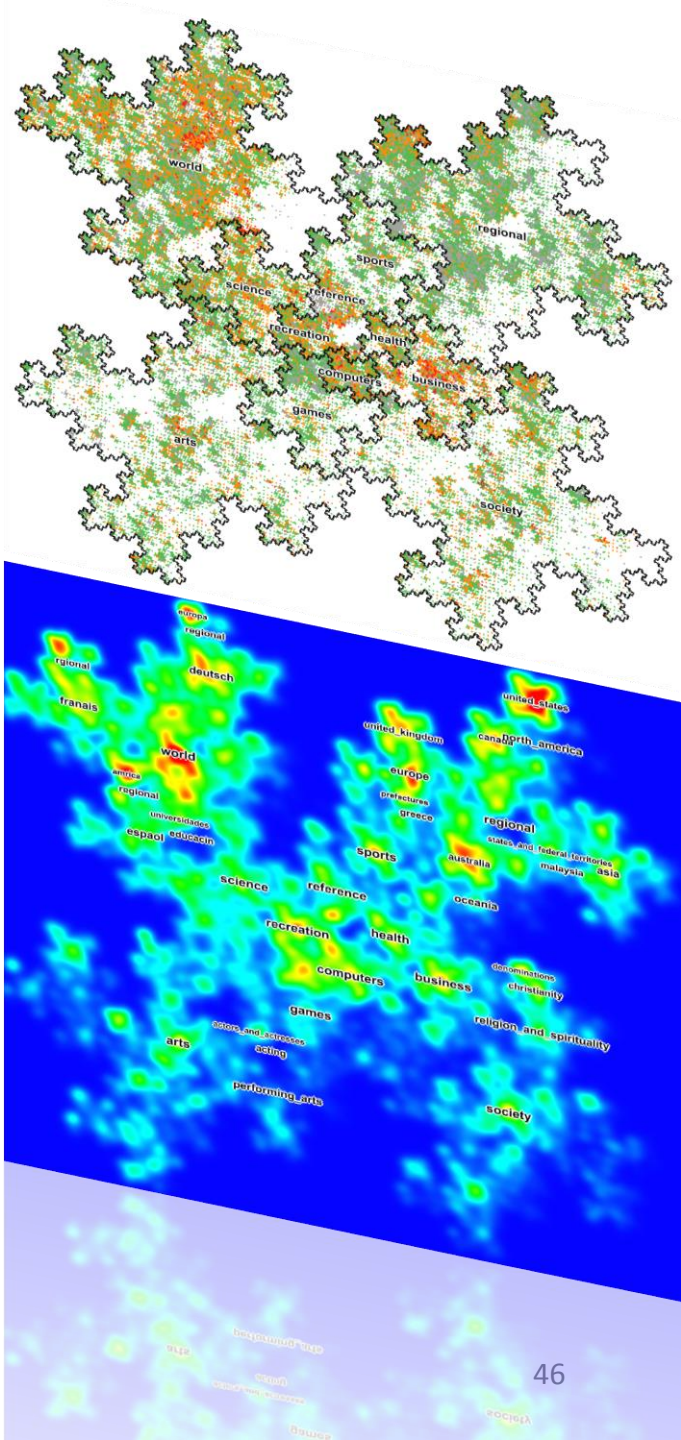
Geometry Level (Mapping)

In situ Visualization [Hadlak et al. 2011]

Hadlak2011 - InSitu 1.avi

Node Set Simplification on

IMAGE LEVEL (RENDERING)

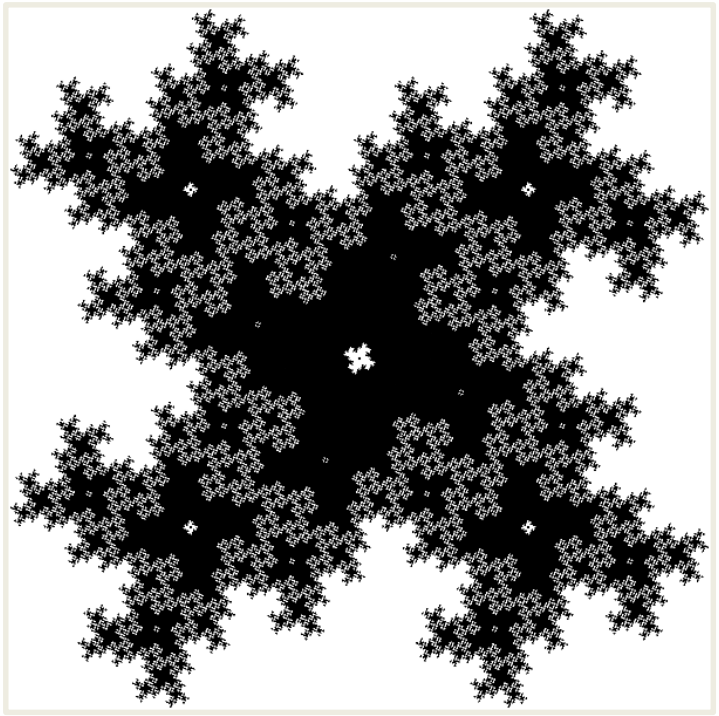


Reducing Clutter on Image Level (Rendering)

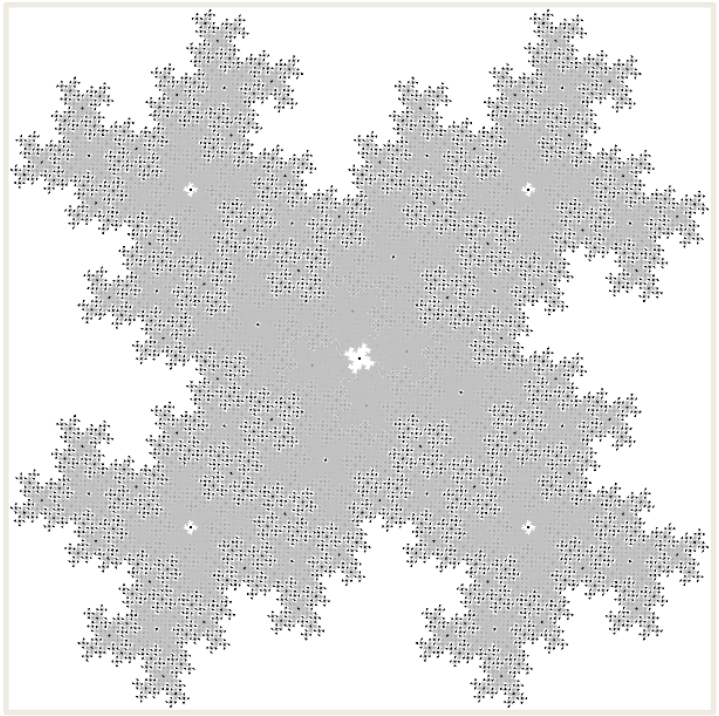
Idea: Normalization

Example: Lightness Adaptation

the same regular, balanced tree

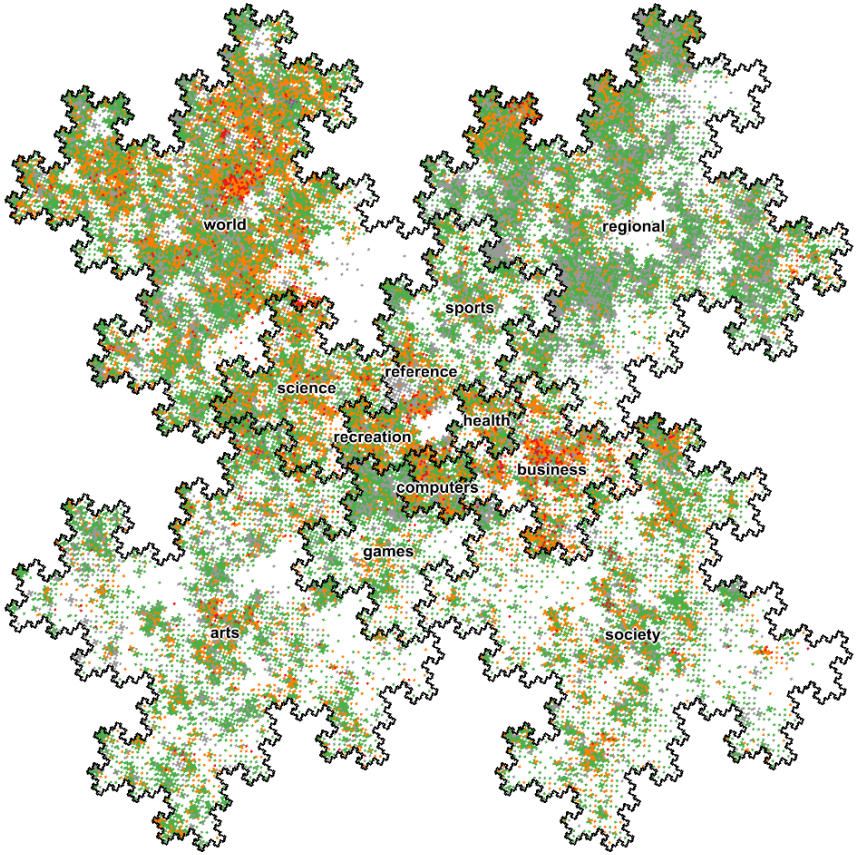


without adaptation



with adaptation

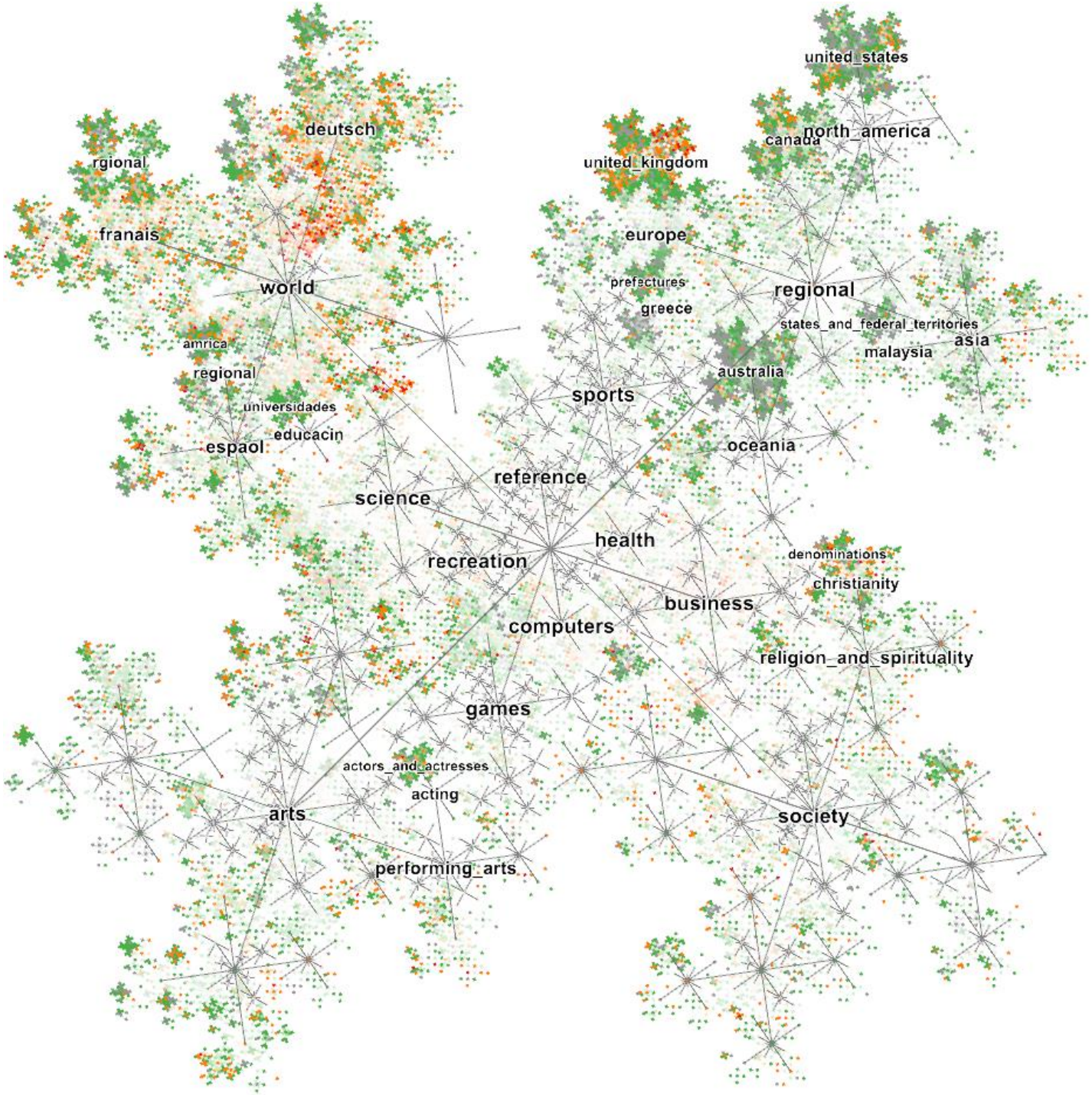
Reducing Clutter on Image Level (Rendering)



without adaptation

with adaptation


europa
regional
deutsch
rgional
franais
world
amrica
regional
universidades
educacin
espaol
science
recreation
arts
performing_arts
actors_and_actresses
acting
games
computers
reference
health
sports
europe
prefectures
greece
united_kingdom
united_states
canada
north_america
regional
states_and_federal_territories
asia
malaysia
australia
oceania
denominations
christianity
business
religion_and_spirituality
society


deutsch
rgional
franais
world
amrica
regional
universidades
educacin
espaol
science
reference
recreation
health
sports
computers
games
arts
actors_and_actresses
acting
performing_arts
united_states
canada
north_america
united_kingdom
europe
prefectures
greece
regional
states_and_federal_territories
asia
malaysia
australia
oceania
denominations
christianity
business
religion_and_spirituality
society

Interacting with Clutter on Image Level (Rendering)

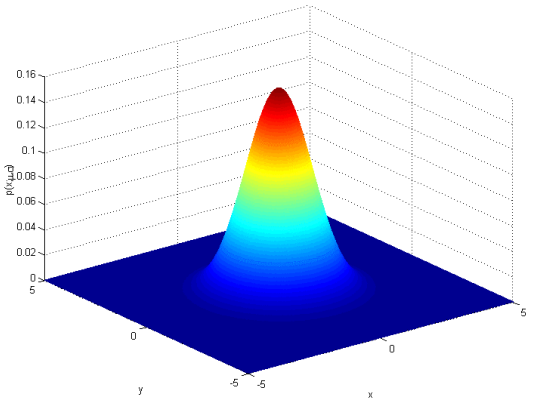
Example: Graph Splatting [van Liere+de Leeuw 2003]

0. Prepare an array M(i,j) of the size of the drawing area

For each node plotted at position (x,y):

1. compute Gaussian centered at (x,y) with a given variance *v*

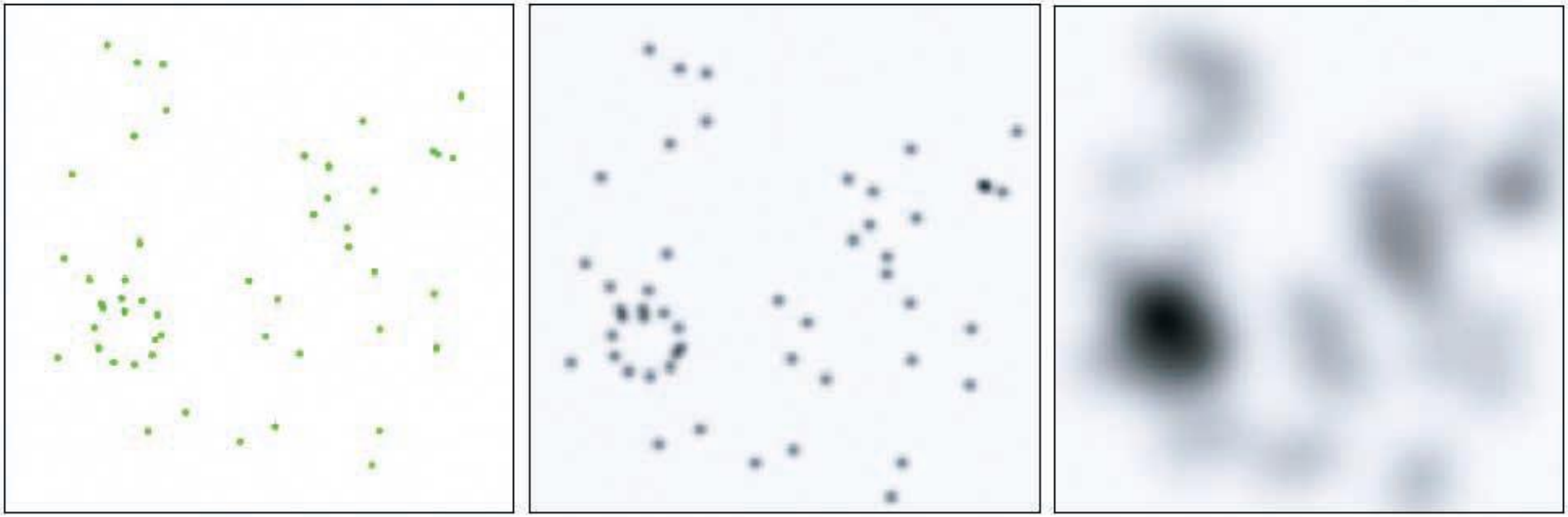
2. add to each cell of M(i,j) the value of the Gaussian at that position



radially symmetric Gaussian distribution

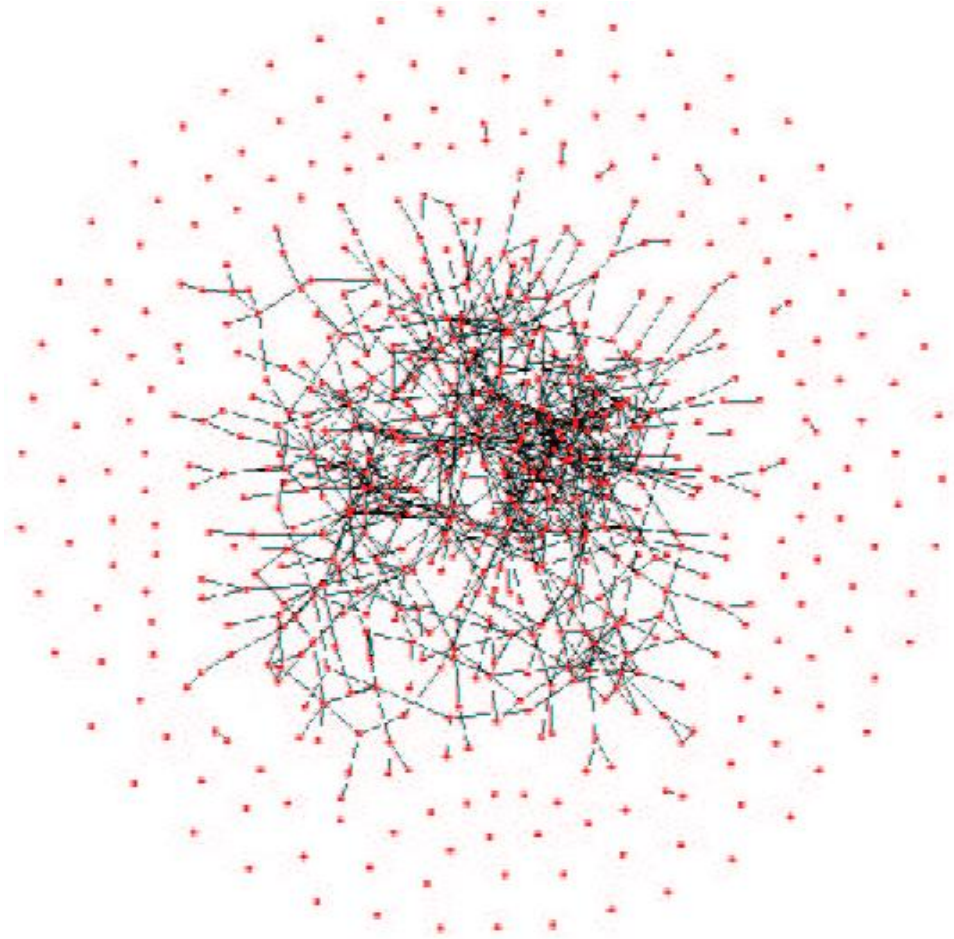
Interacting with Clutter on Image Level (Rendering)

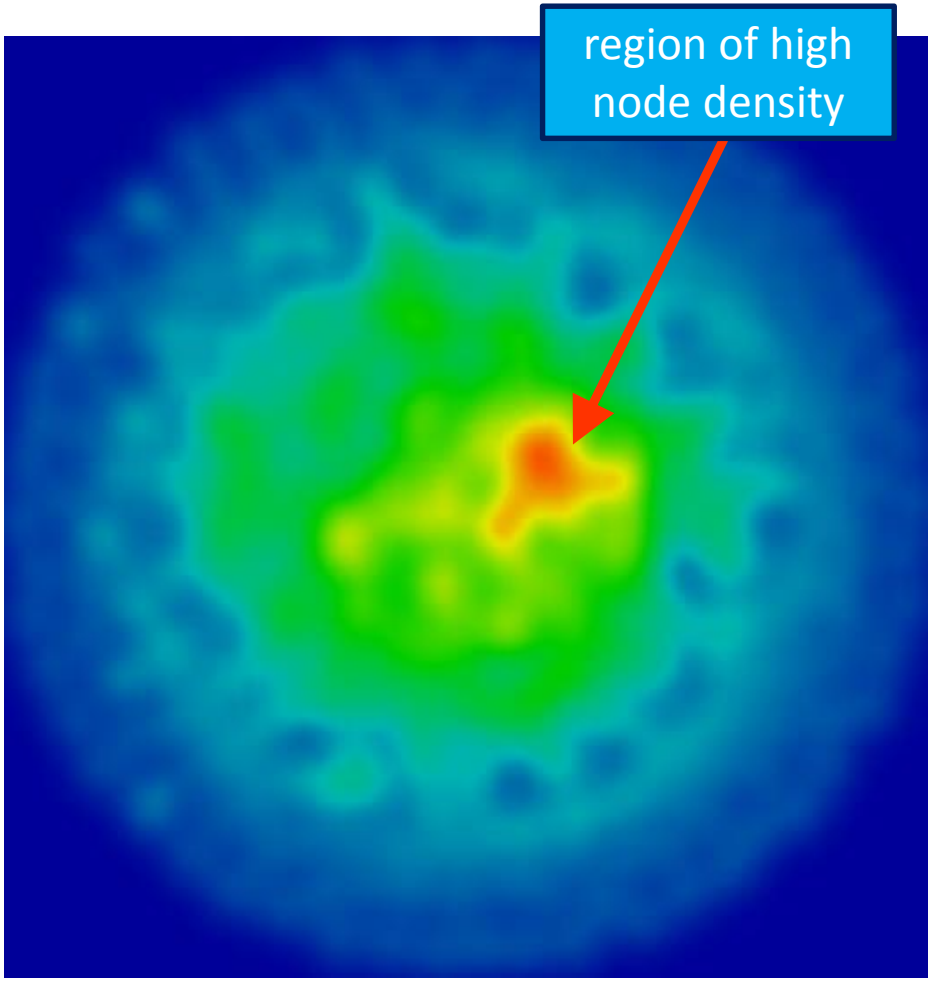
Example: Graph Splatting [van Liere+de Leeuw 2003]



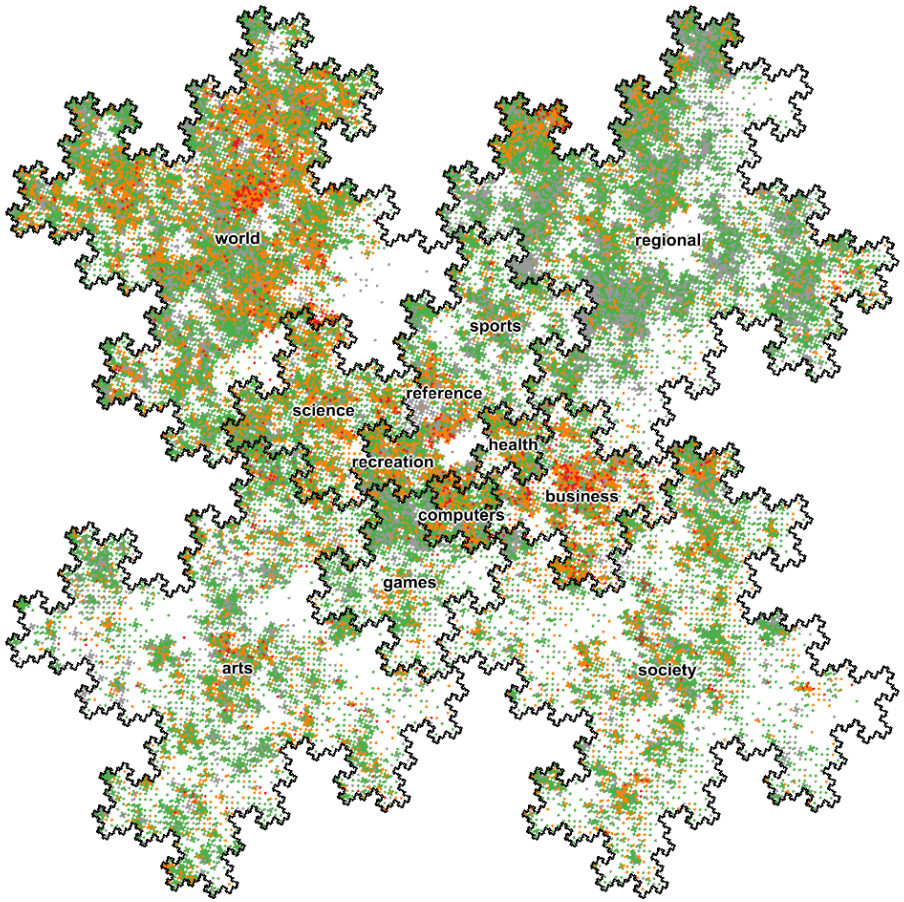
Effect of different variances: left = original layout, middle = low variance, right = high variance

Interacting with Clutter on Image Level (Rendering)

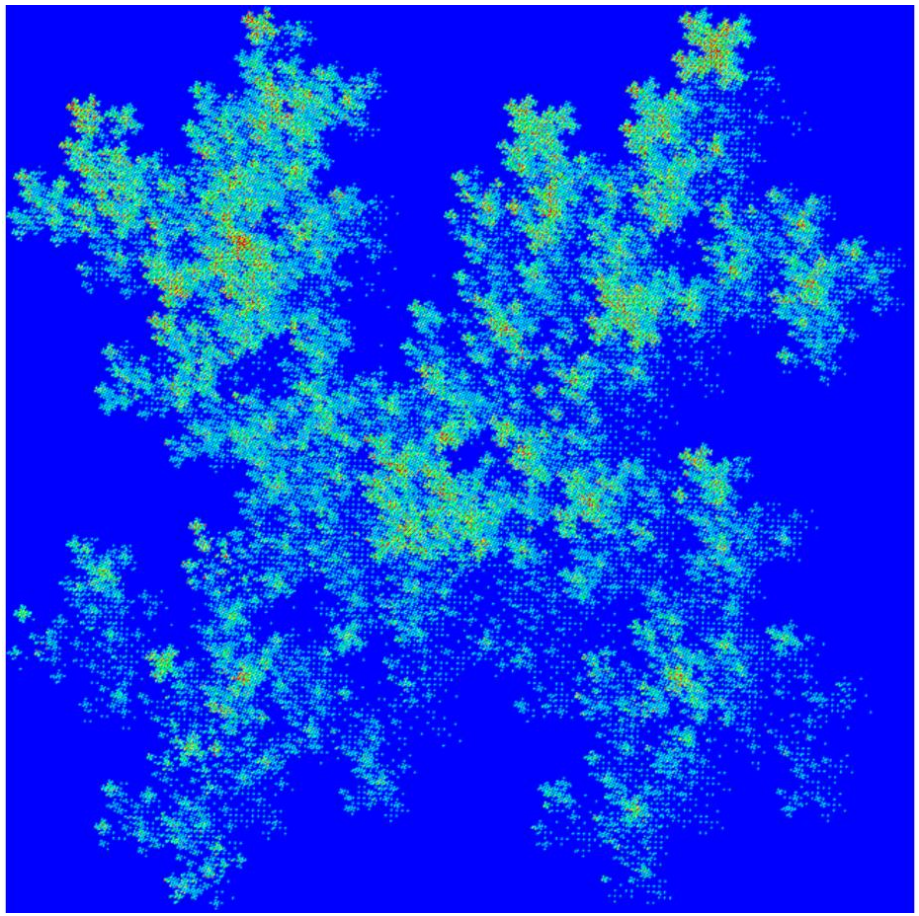




Interacting with Clutter on Image Level (Rendering)

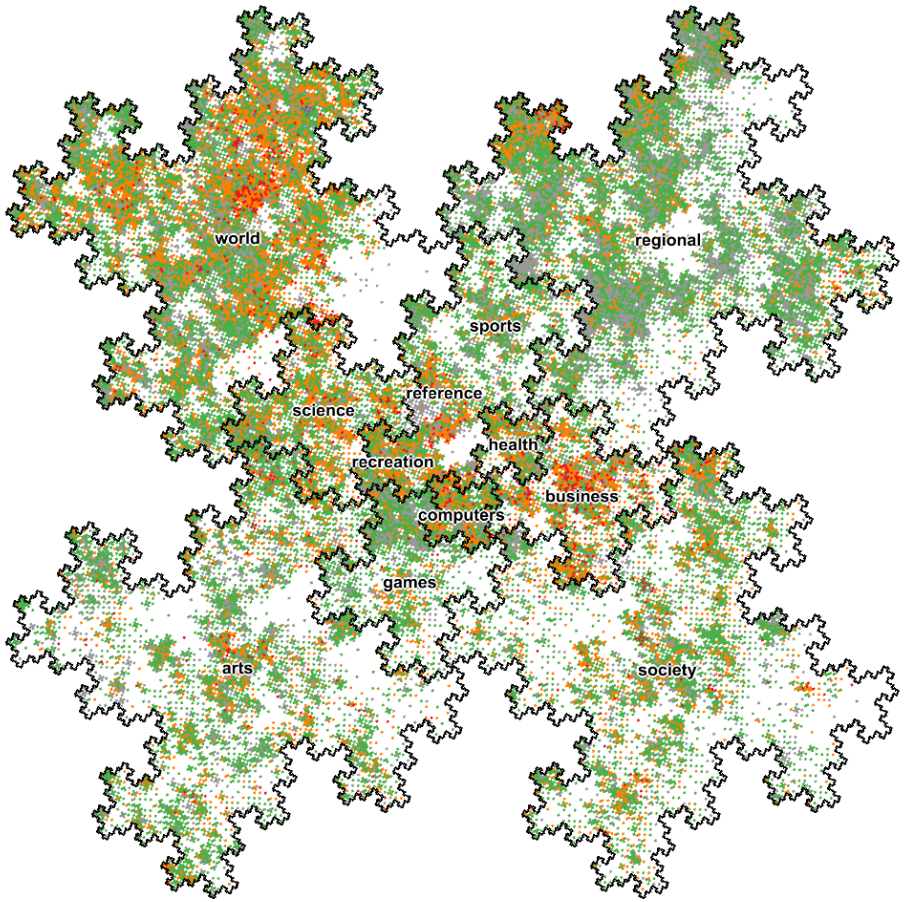


σ = 0.001

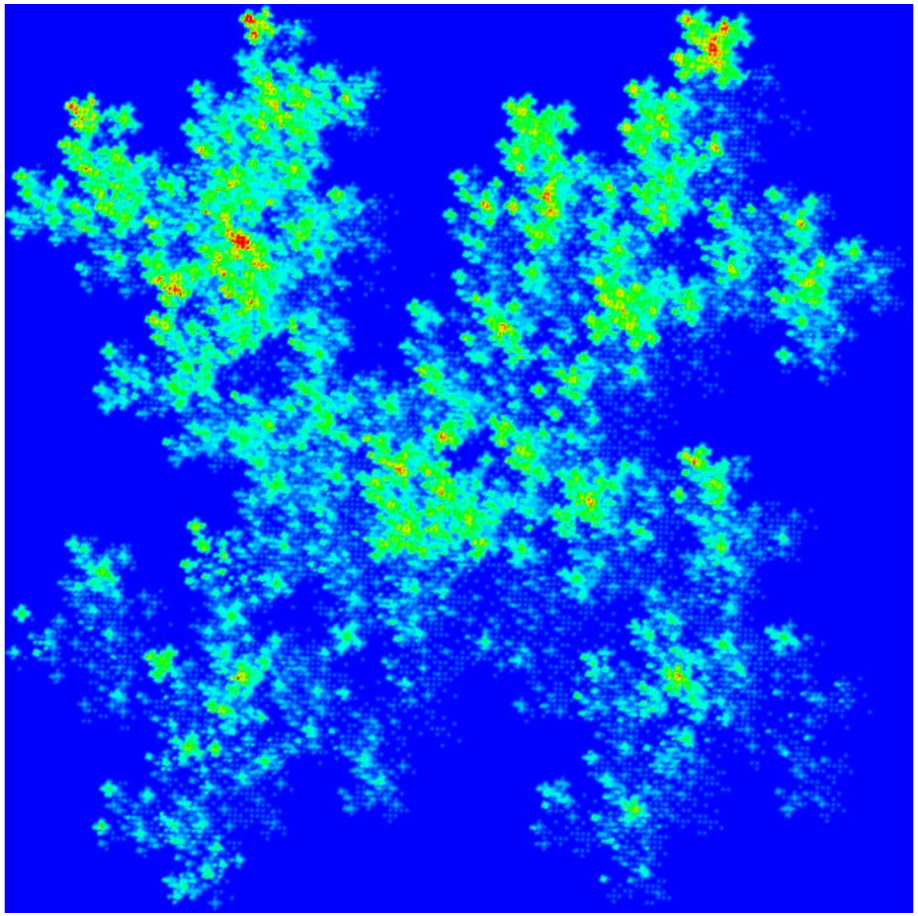


[images courtesy of Steffen Hadlak]

Interacting with Clutter on Image Level (Rendering)

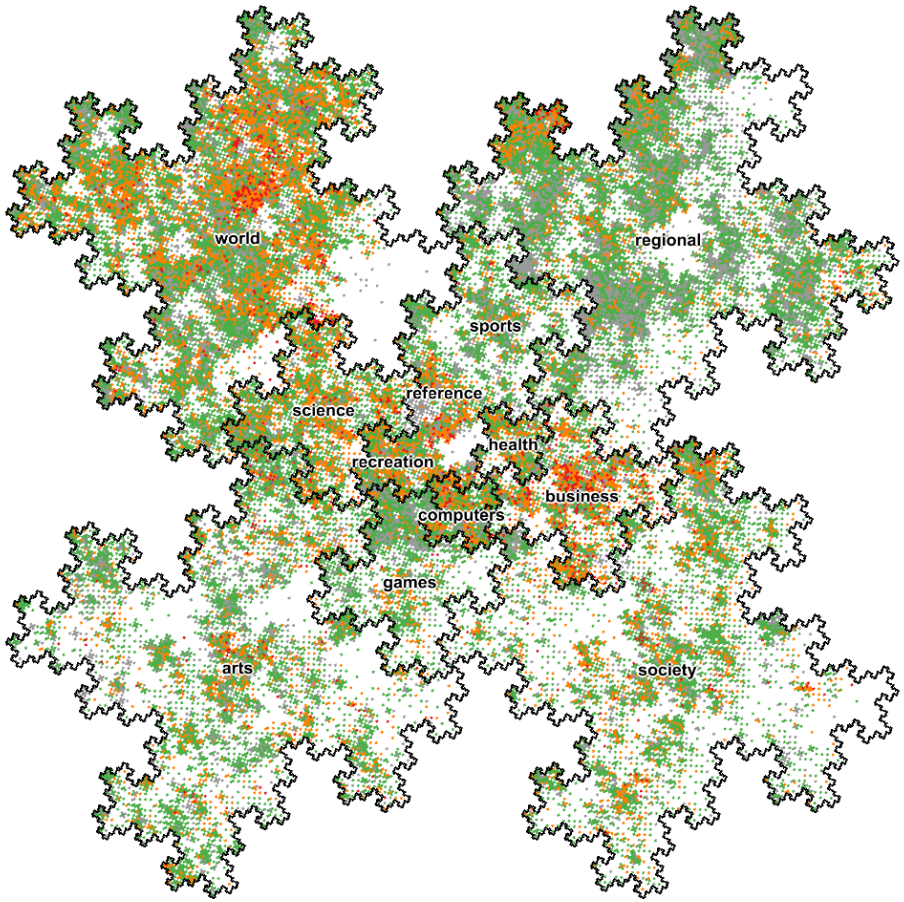


σ = 0.002

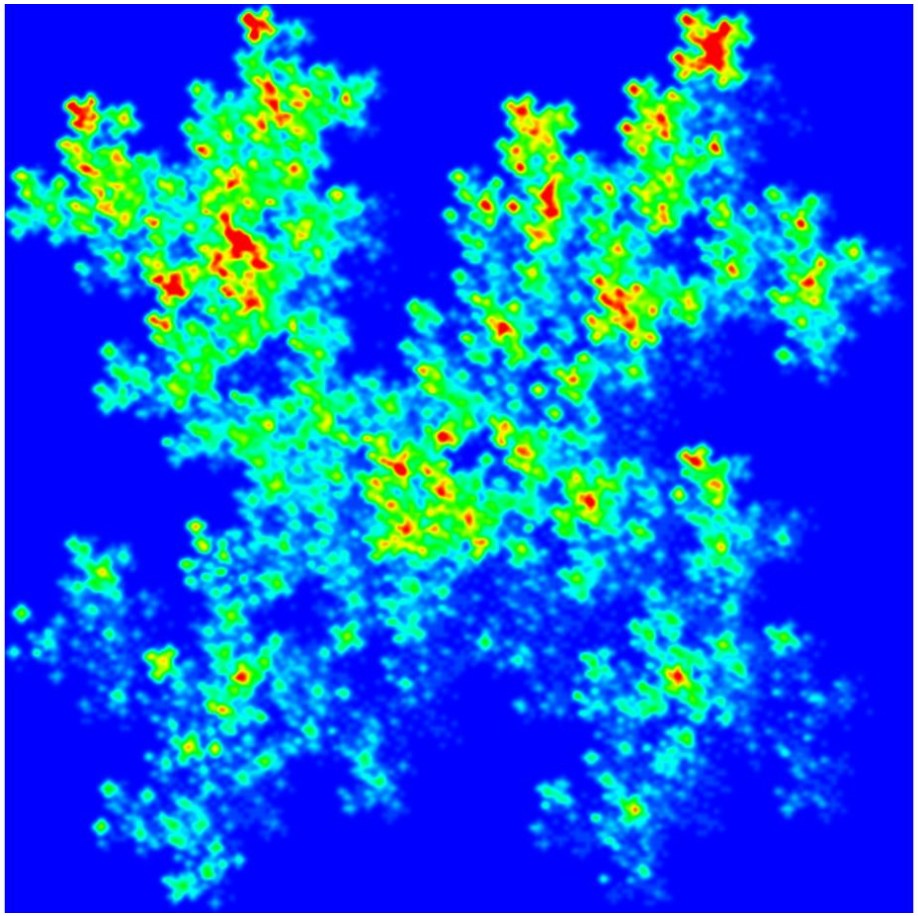


[images courtesy of Steffen Hadlak]

Interacting with Clutter on Image Level (Rendering)

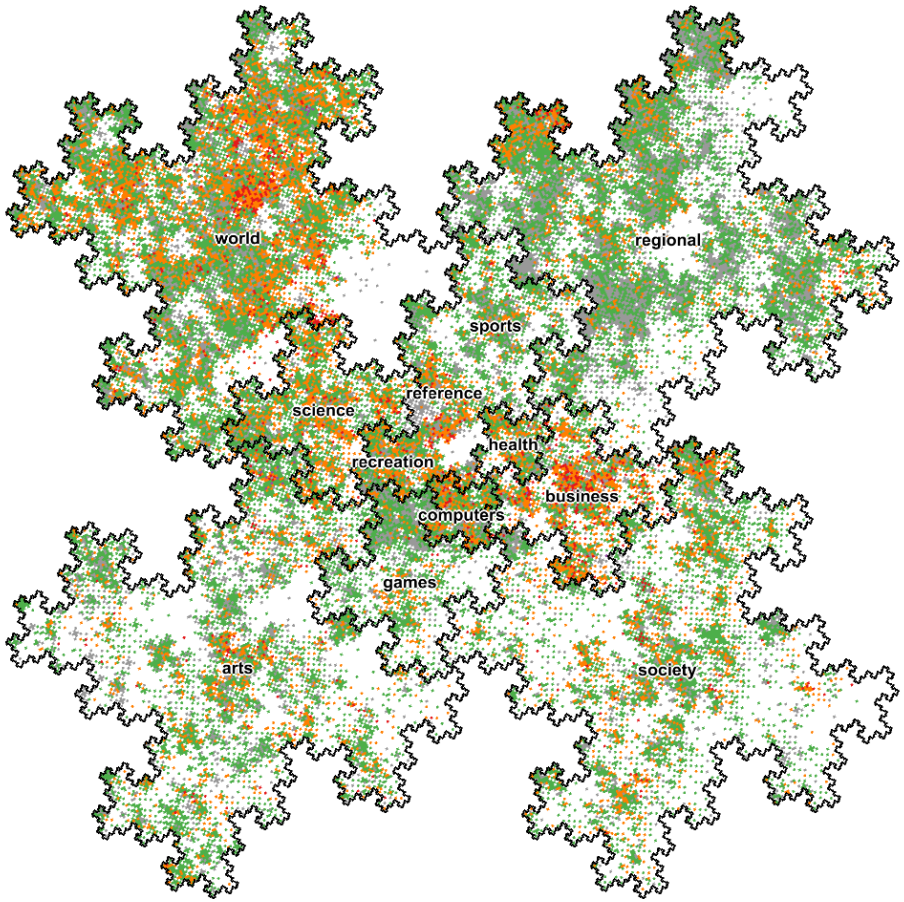


σ = 0.004

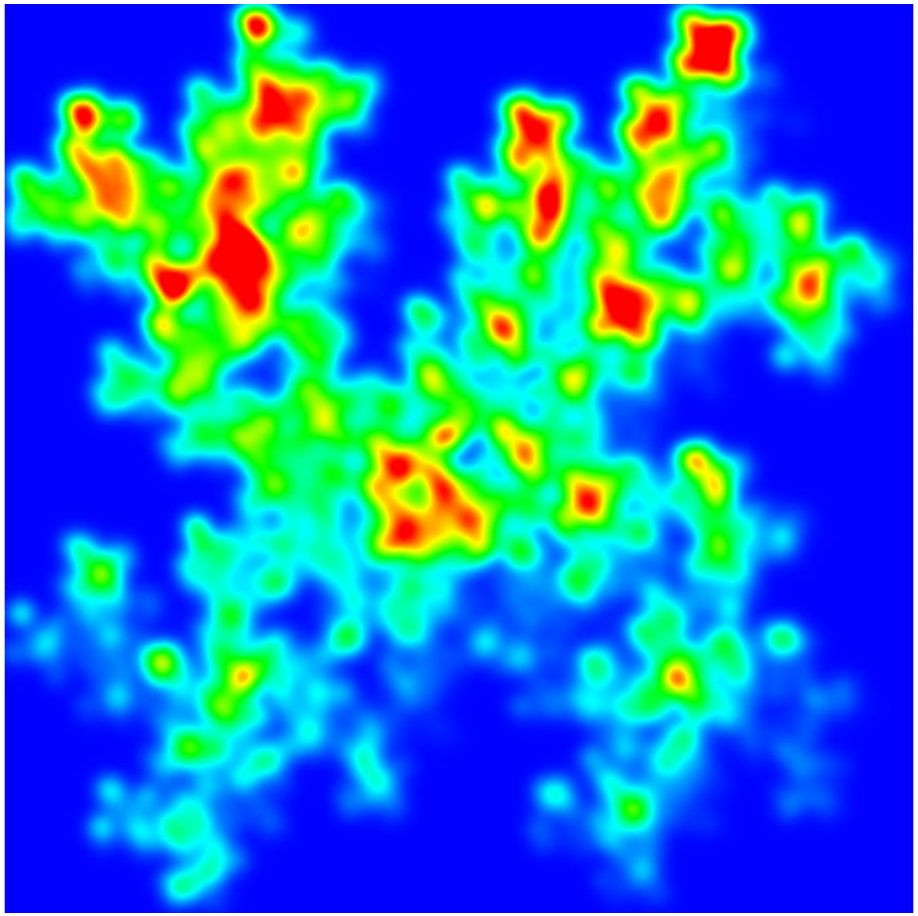


[images courtesy of Steffen Hadlak]

Interacting with Clutter on Image Level (Rendering)

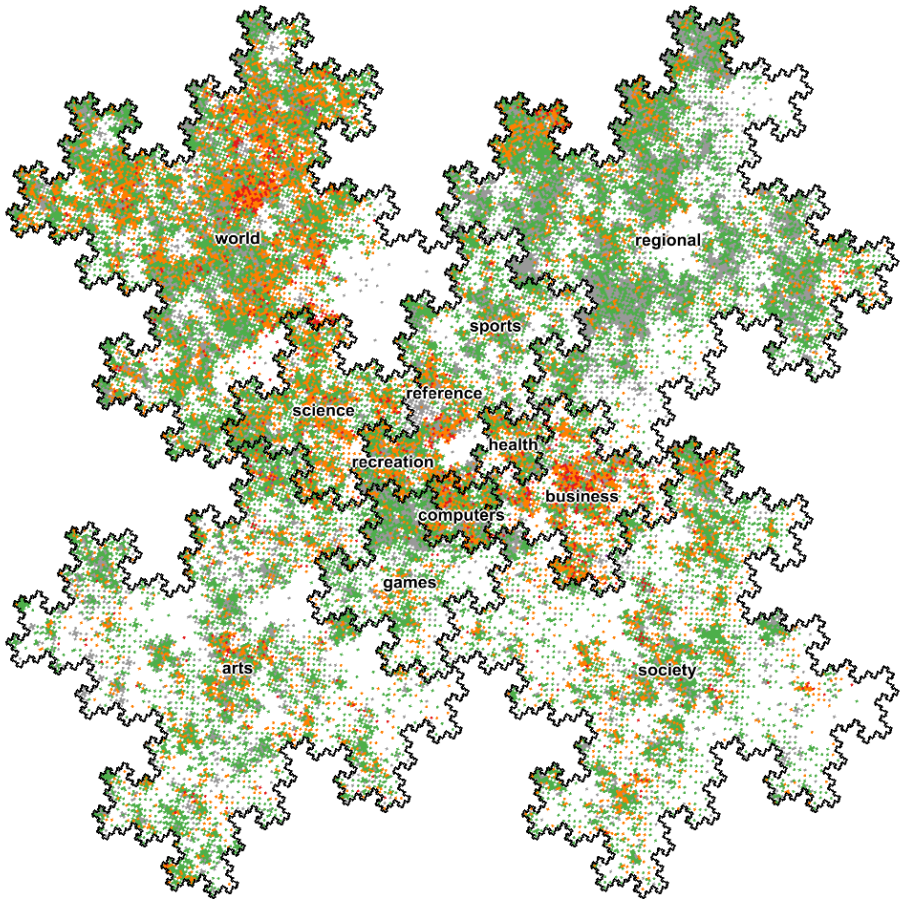


σ = 0.015

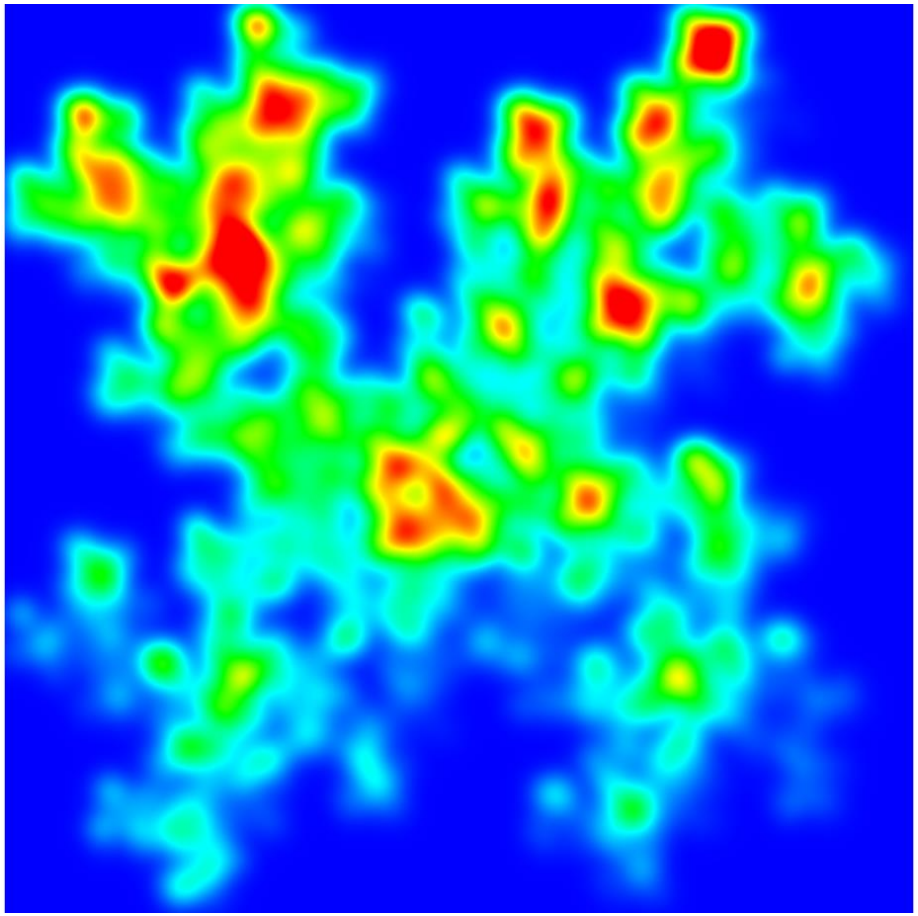


[images courtesy of Steffen Hadlak]

Interacting with Clutter on Image Level (Rendering)



σ = 0.020



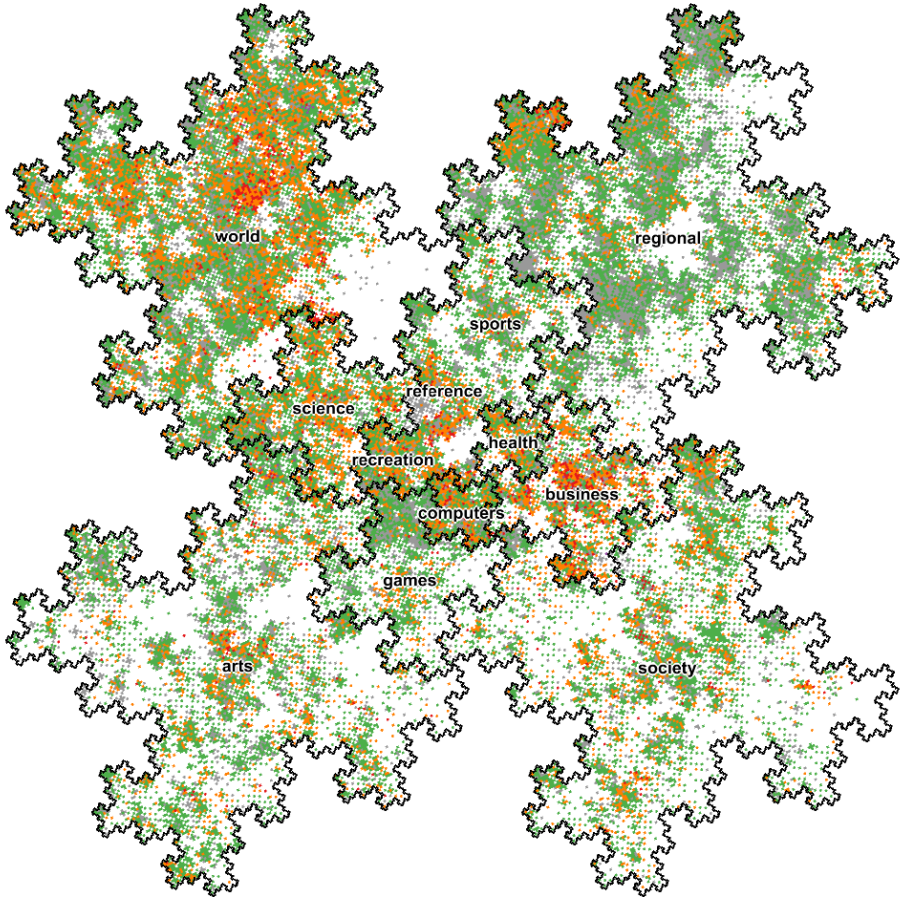
[images courtesy of Steffen Hadlak]

Interacting with Clutter on Image Level (Rendering)

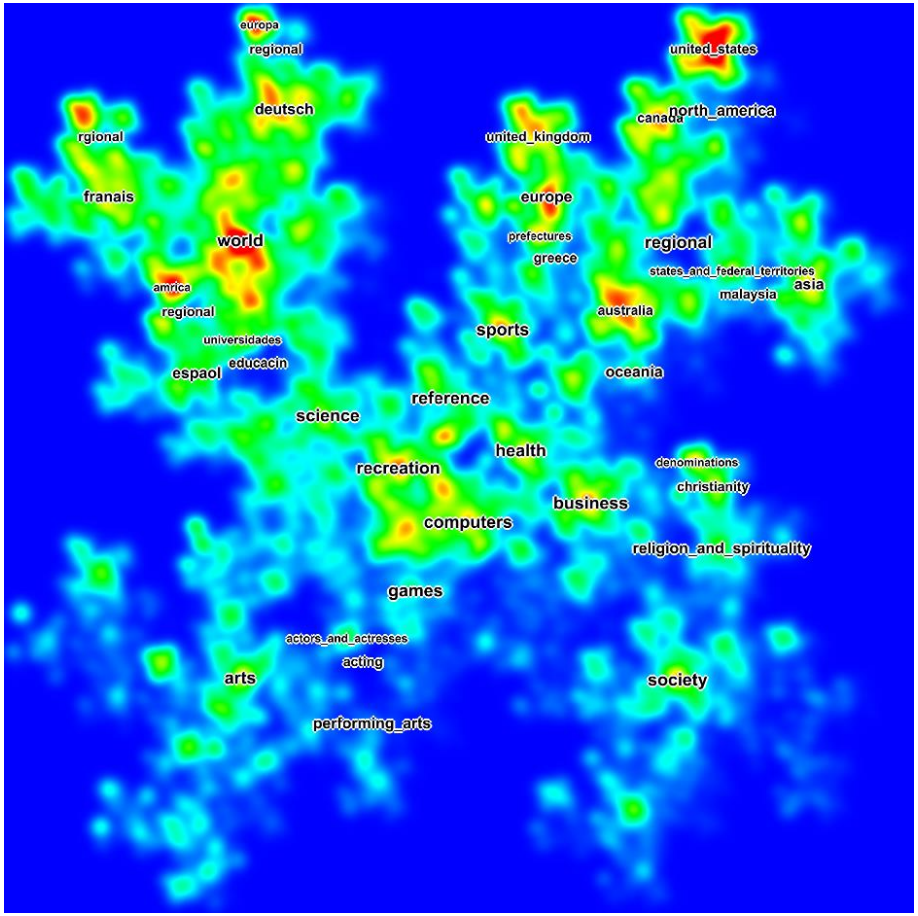
Example: Graph Splatting [van Liere+de Leeuw 2003]

Schulz2011 - Pointbased (TVCG).avi

Interacting with Clutter on Image Level (Rendering)



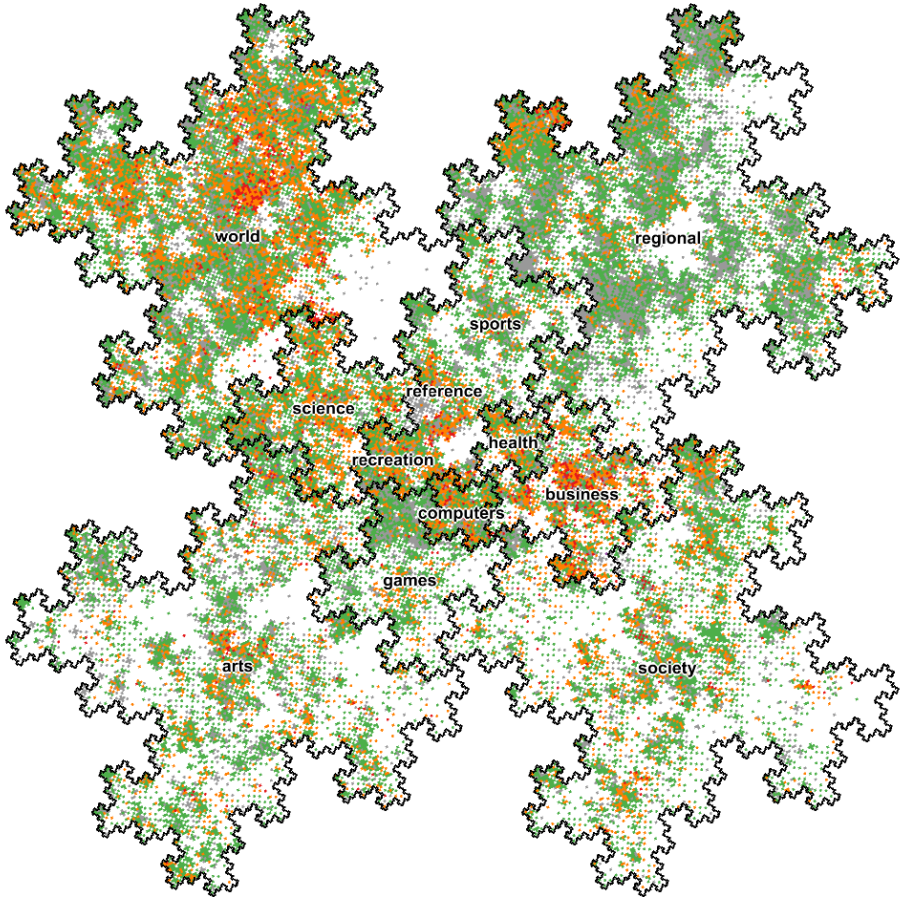
all nodes are equal

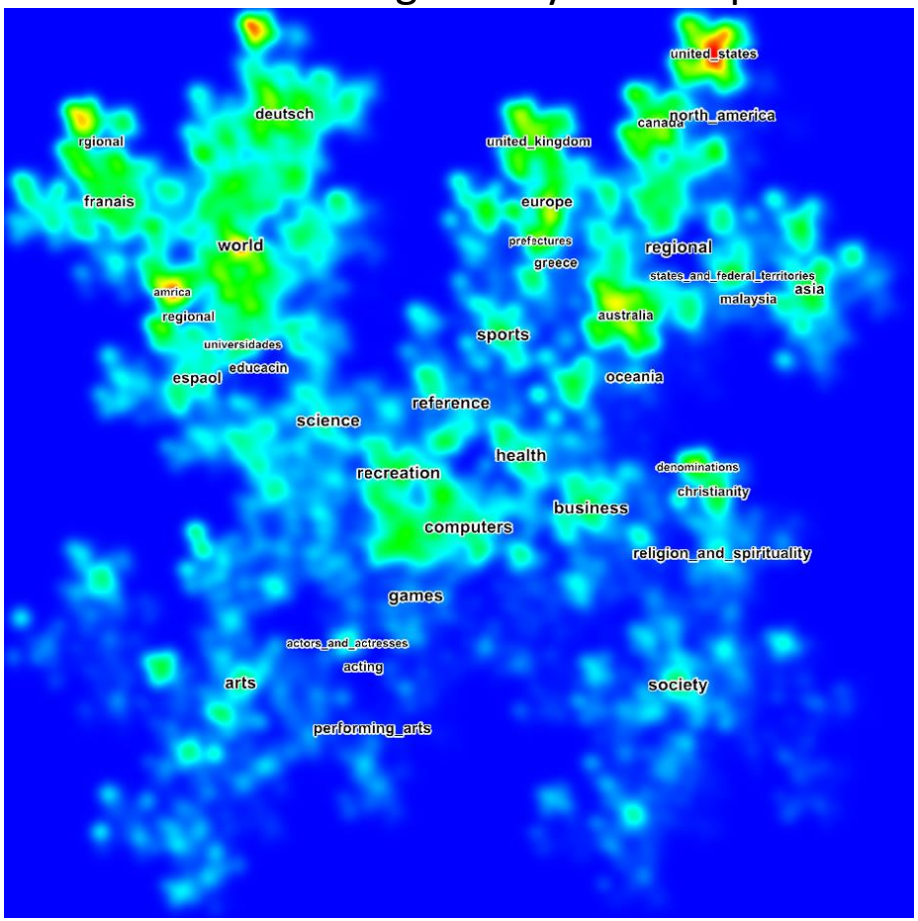


[images courtesy of Steffen Hadlak]

Interacting with Clutter on Image Level (Rendering)

nodes are weighted by their depth

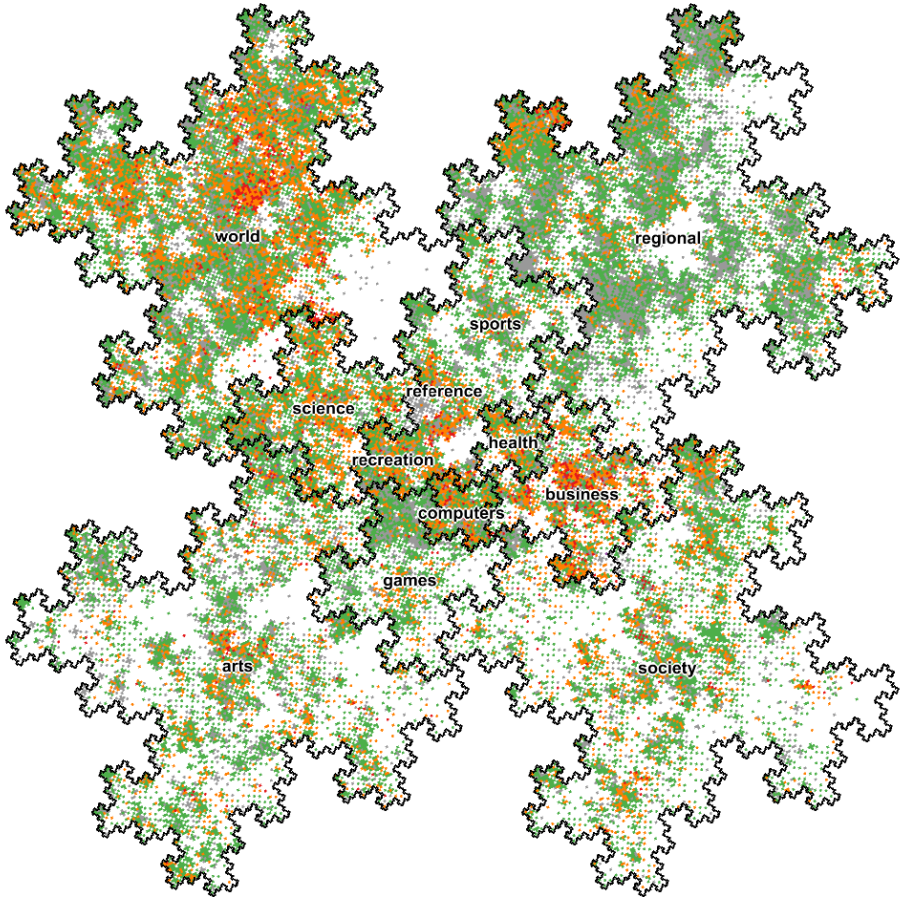


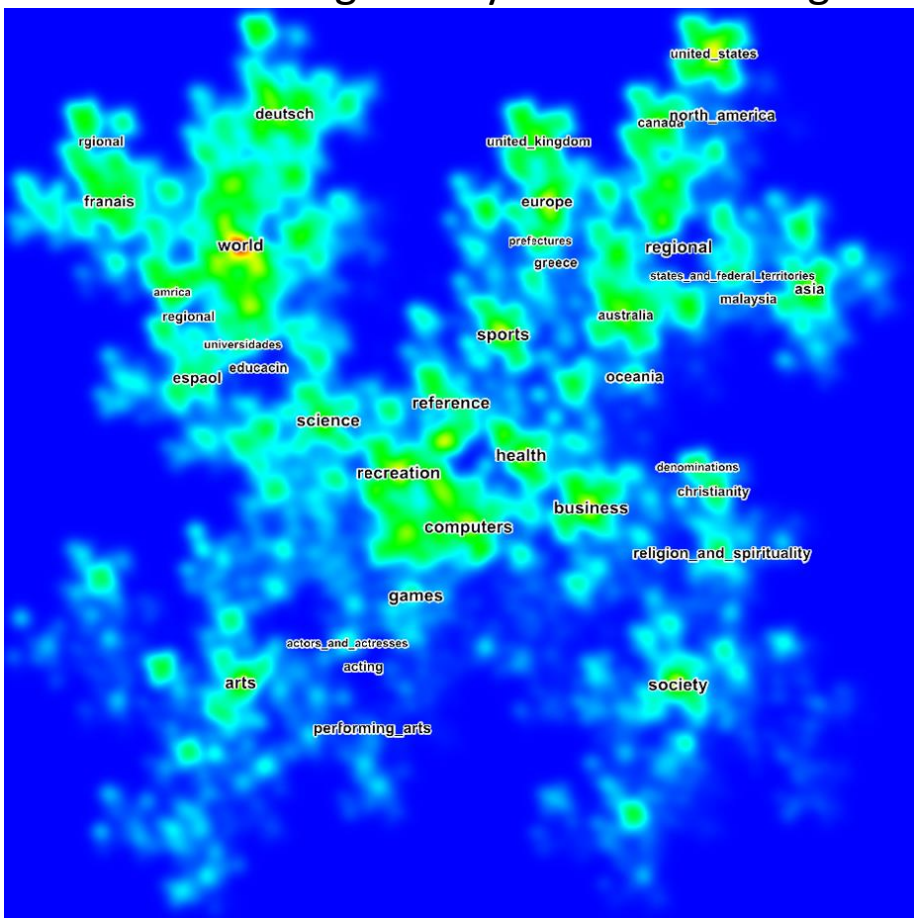


[images courtesy of Steffen Hadlak]

Interacting with Clutter on Image Level (Rendering)

nodes are weighted by their # of siblings

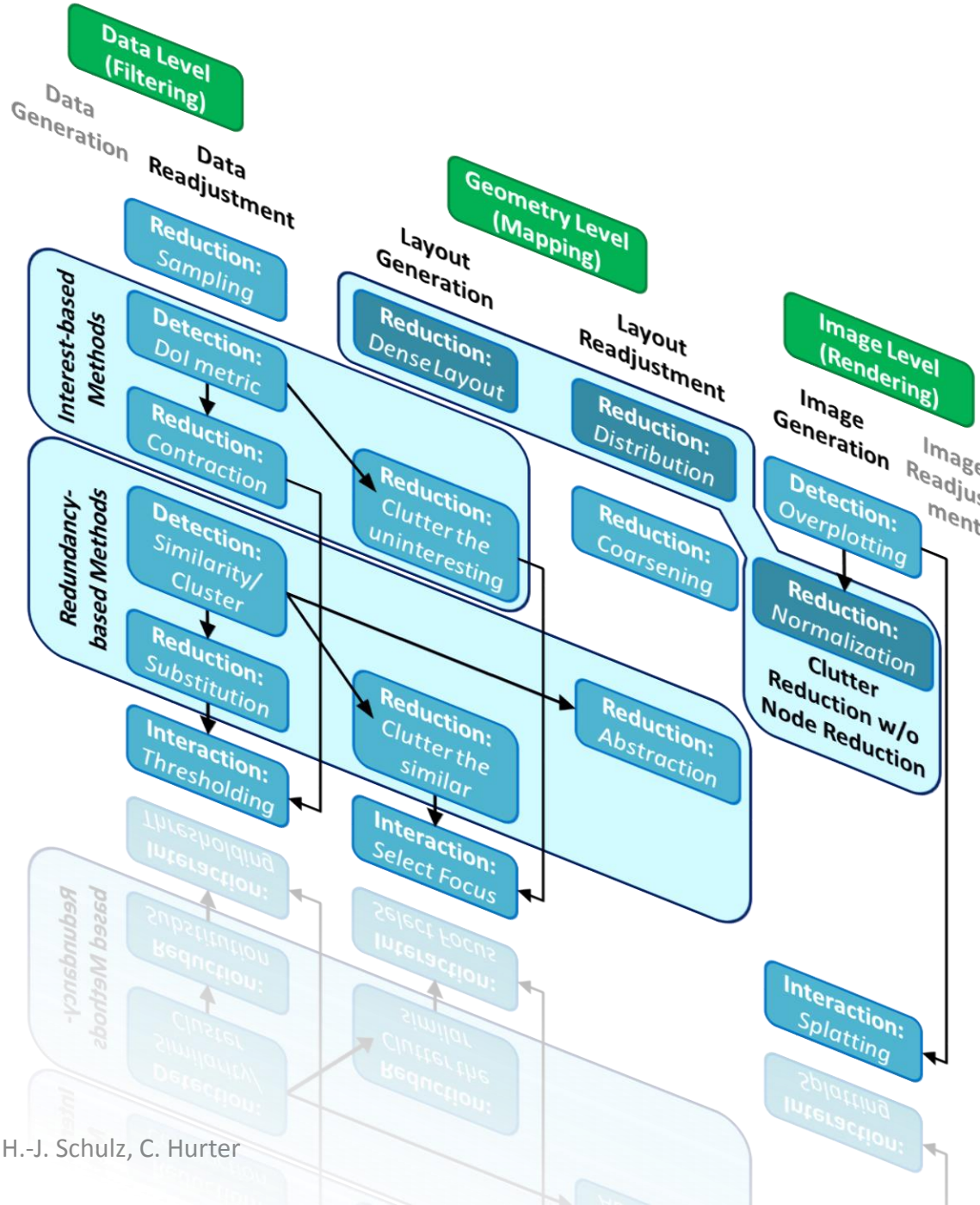


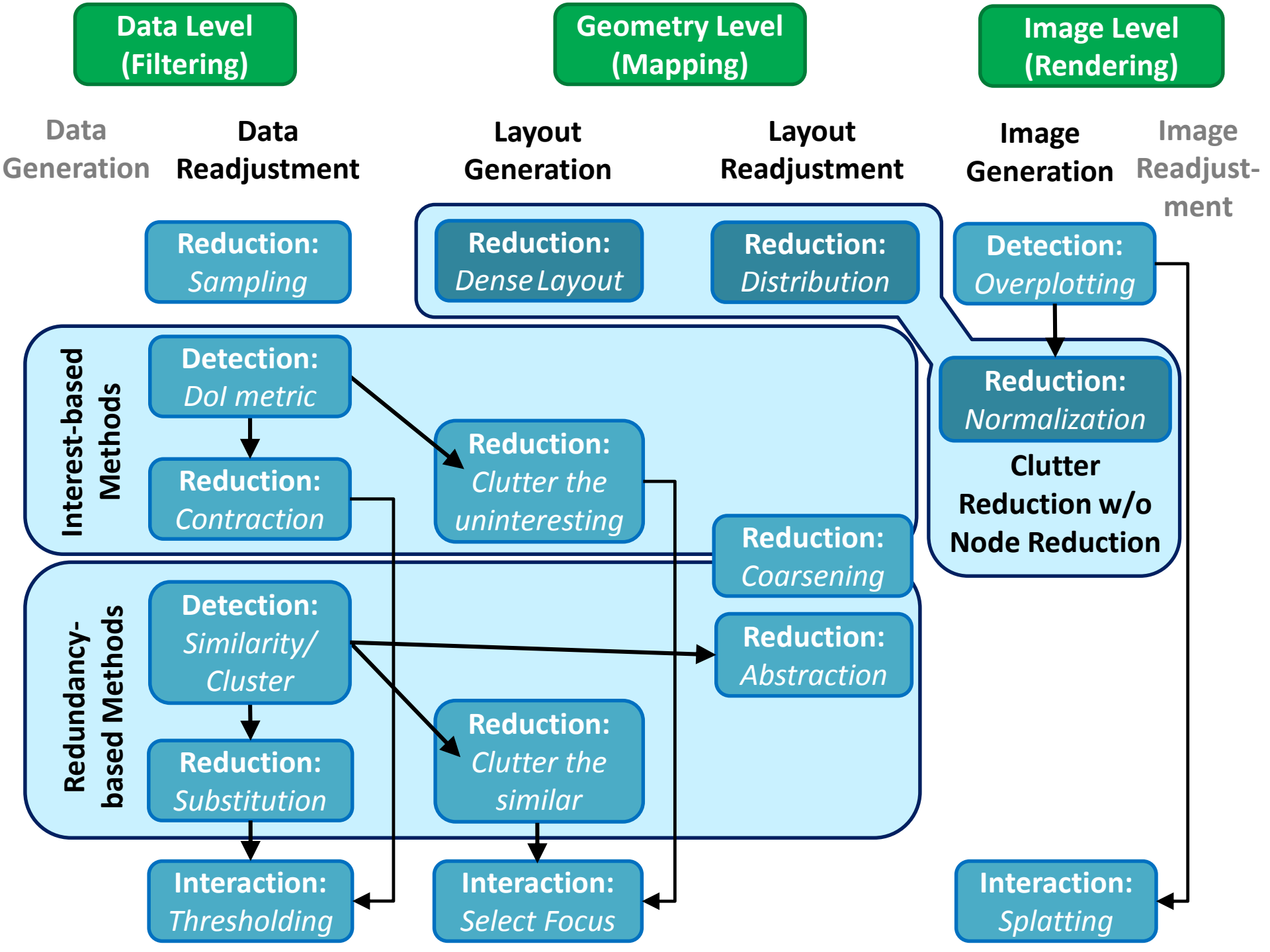


[images courtesy of Steffen Hadlak]

Node Set Simplification

SUMMARY




Data Level (Filtering)
Geometry Level (Mapping)
Image Level (Rendering)
Data Generation
Data Readjustment
Layout Generation
Layout Readjustment
Image Generation
Image Readjust-ment
Reduction: Sampling
Reduction: Dense Layout
Reduction: Distribution
Detection: Overplotting
Interest-based Methods
Detection: DoI metric
Reduction: Contraction
Reduction: Clutter the uninteresting
Reduction: Normalization
Clutter Reduction w/o Node Reduction
Reduction: Coarsening
Redundancy-based Methods
Detection: Similarity/ Cluster
Reduction: Abstraction
Reduction: Substitution
Reduction: Clutter the similar
Interaction: Thresholding
Interaction: Select Focus
Interaction: Splatting