

<http://poloclub.gatech.edu/cse6242>

CSE6242 / CX4242: **Data** & **Visual** Analytics

# Clustering

Duen Horng (Polo) Chau

Assistant Professor

Associate Director, MS Analytics

Georgia Tech

Partly based on materials by

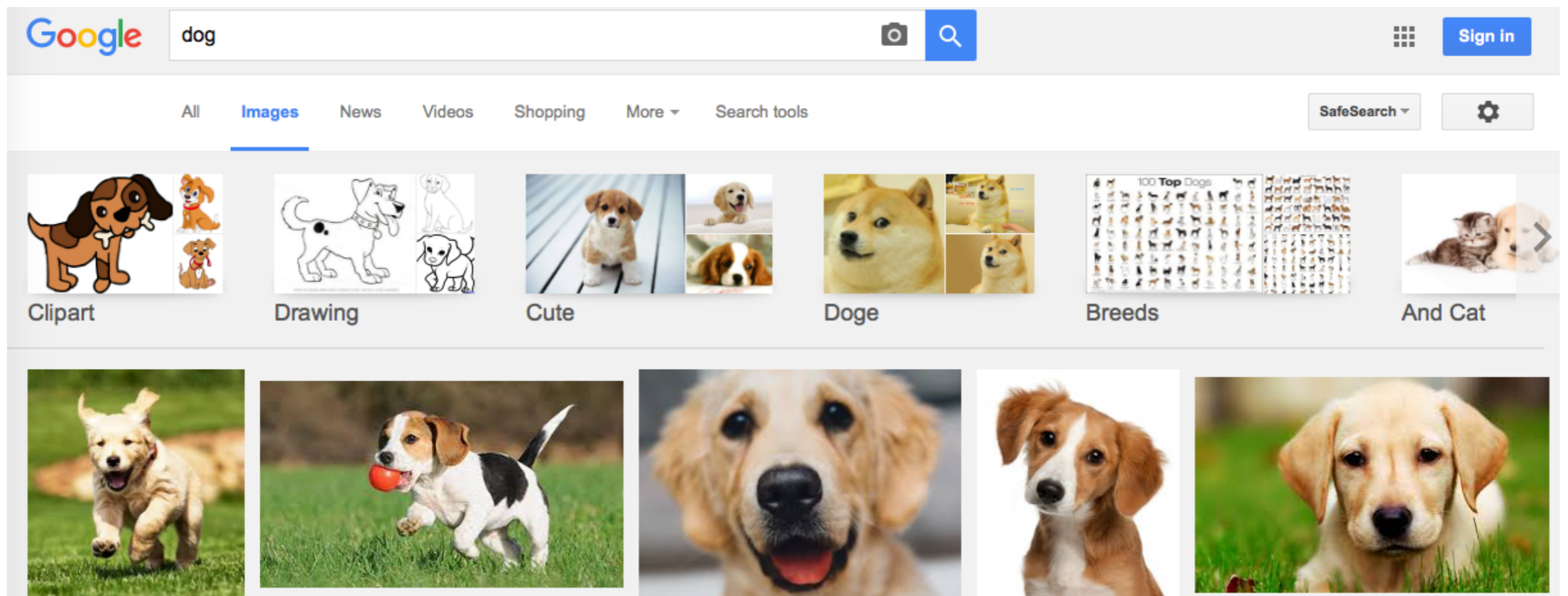
Professors Guy Lebanon, Jeffrey Heer, John Stasko, Christos Faloutsos, Parishit Ram (GT PhD alum; SkyTree), Alex Gray

# Clustering

The most common type of **unsupervised** learning

High-level idea: group **similar** things together

“**Unsupervised**” because clustering model is learned without any labeled examples



# Applications of Clustering

- Find similar patients subgroups
  - e.g., in healthcare
- Finding groups of similar text documents (topic modeling)
- ...

Clustering techniques you've got to know

**K-means**

**Hierarchical Clustering**

**DBSCAN**

# K-means (the “simplest” technique)

**Best D3 demo Polo could find:** <http://tech.nitoyon.com/en/blog/2013/11/07/k-means/>

## Algorithm Summary

- We tell K-means the value of **k** (#clusters we want)
- **Randomly** initialize the k cluster “means” (“centroids”)
- **Assign** each item to the the cluster whose mean the item is closest to (so, we need a **similarity function**)
- **Update/recompute** the new “means” of all k clusters.
- If all items’ assignments do not change, **stop**.

**YouTube video demo:** <https://youtu.be/luRb3y8qKX4?t=3m4s>

# K-means What's the catch?

<http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html>

## How to **decide k** (a hard problem)?

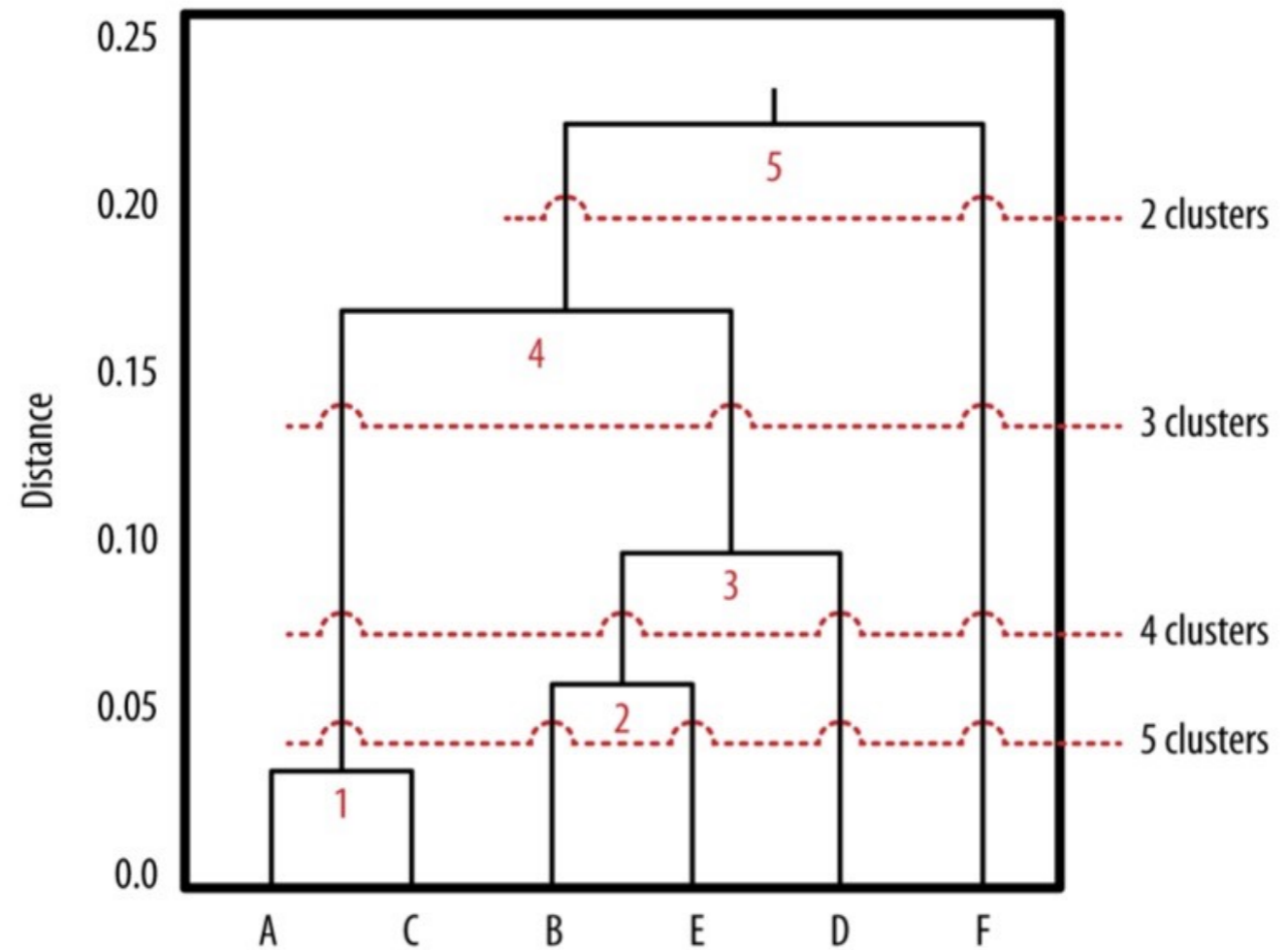
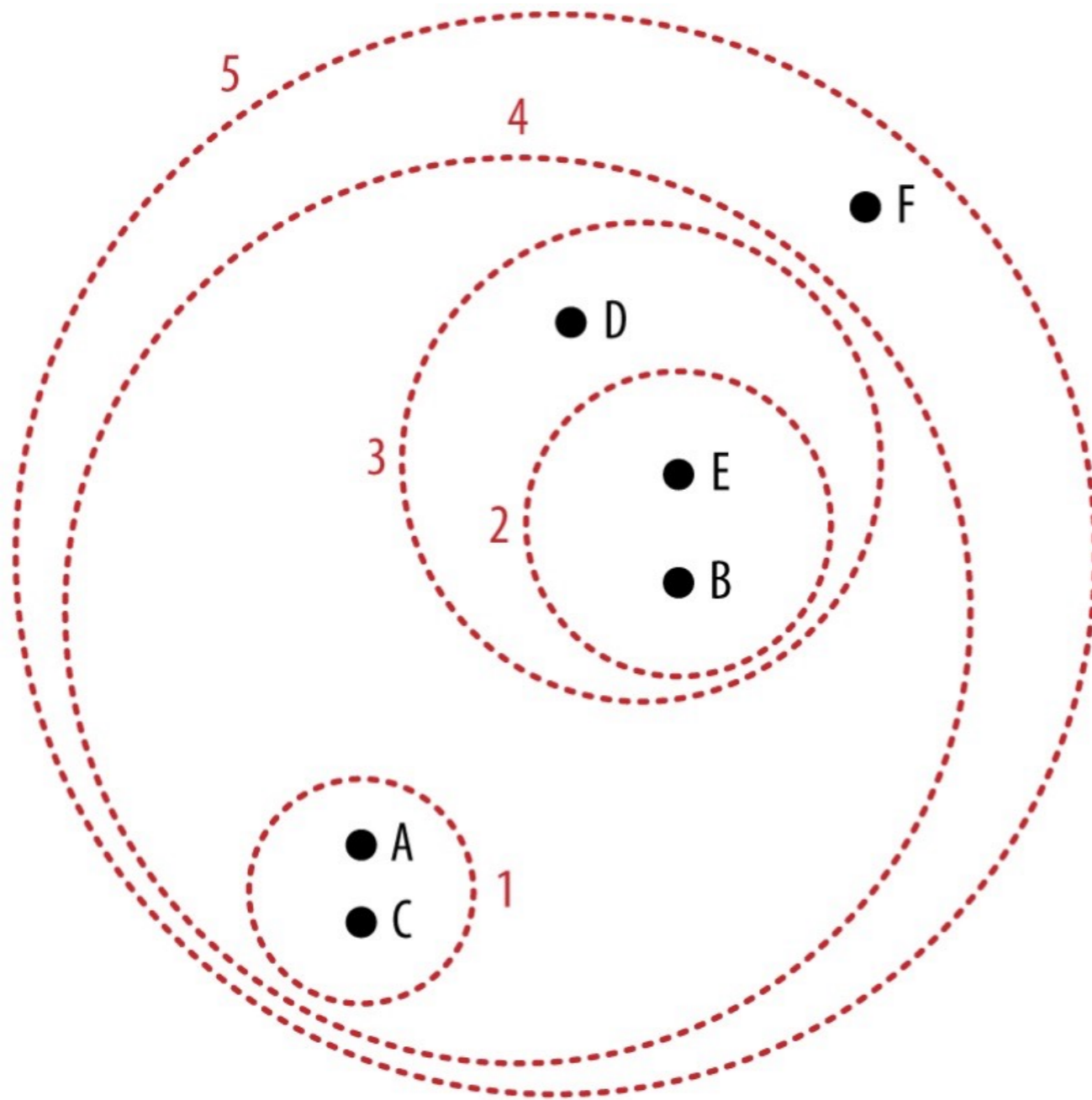
- A few ways; best way is to evaluate with real data  
(<https://www.ee.columbia.edu/~dpwe/papers/PhamDN05-kmeans.pdf>)

## Only locally optimal (vs global)

- Different initialization gives different clusters
  - How to “fix” this?
- “Bad” starting points can cause algorithm to converge slowly
- Can work for relatively large dataset
  - Time complexity  $O(d n \log n)$  per iteration  
(assumptions:  $n \gg k$ , dimension  $d$  is small)  
<http://www.cs.cmu.edu/~./dpelleg/download/kmeans.ps>

# Hierarchical clustering

High-level idea: build a tree (hierarchy) of clusters

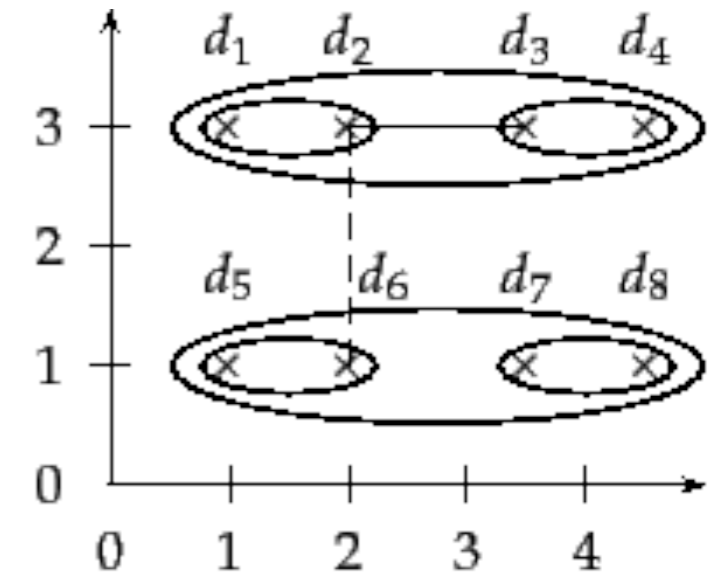


Dendrogram

# Ways to calculate distances between two clusters

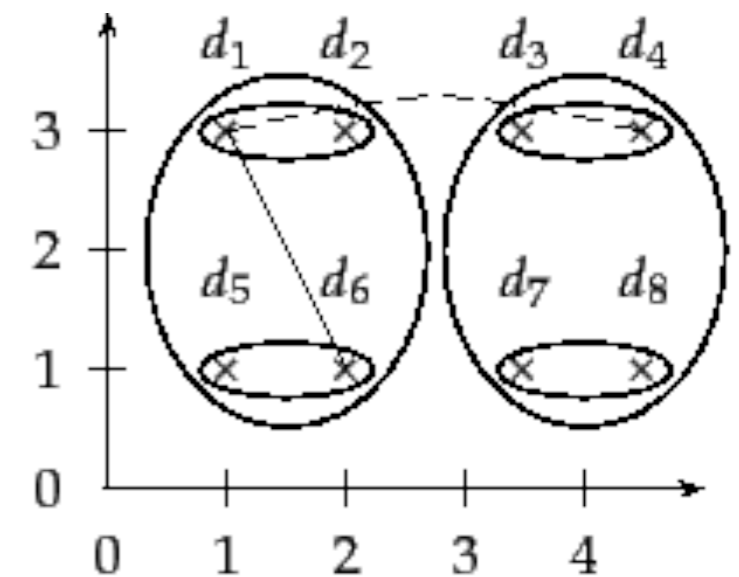
## Single linkage

- minimum of distance between clusters
- similarity of two clusters = similarity of the clusters' **most similar** members



## Complete linkage

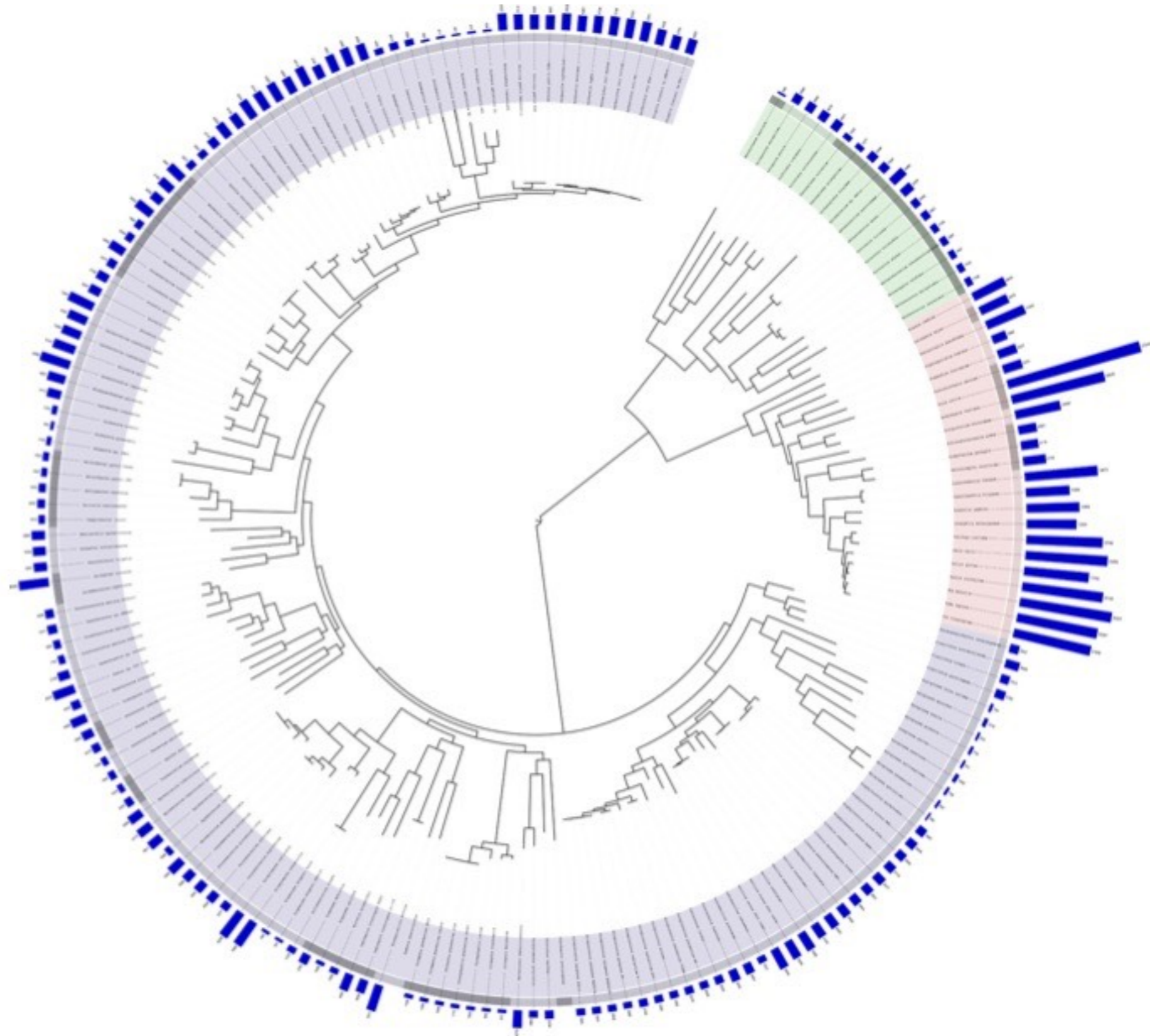
- maximum of distance between clusters
- similarity of two clusters = similarity of the clusters' **most dissimilar** members



## Average linkage

- distance between cluster centers





<https://bl.ocks.org/mbostock/4063570>  
<https://bl.ocks.org/mbostock/4339607>

# Hierarchical clustering for large datasets?

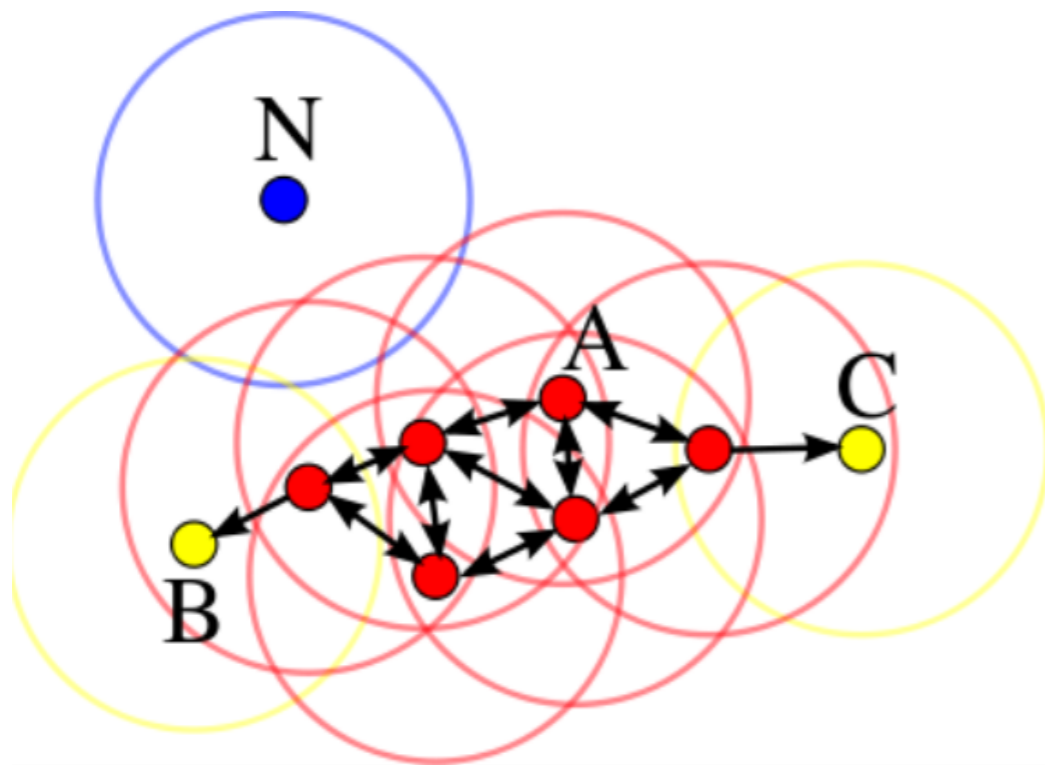
- OK for small datasets (e.g., <10K items)
- Time complexity between  $O(n^2)$  to  $O(n^3)$  where  $n$  is the number of data items
- Not good for millions of items or more
- But great for understanding concept of clustering

# DBSCAN

“Density-based spatial clustering with noise”

<https://en.wikipedia.org/wiki/DBSCAN>

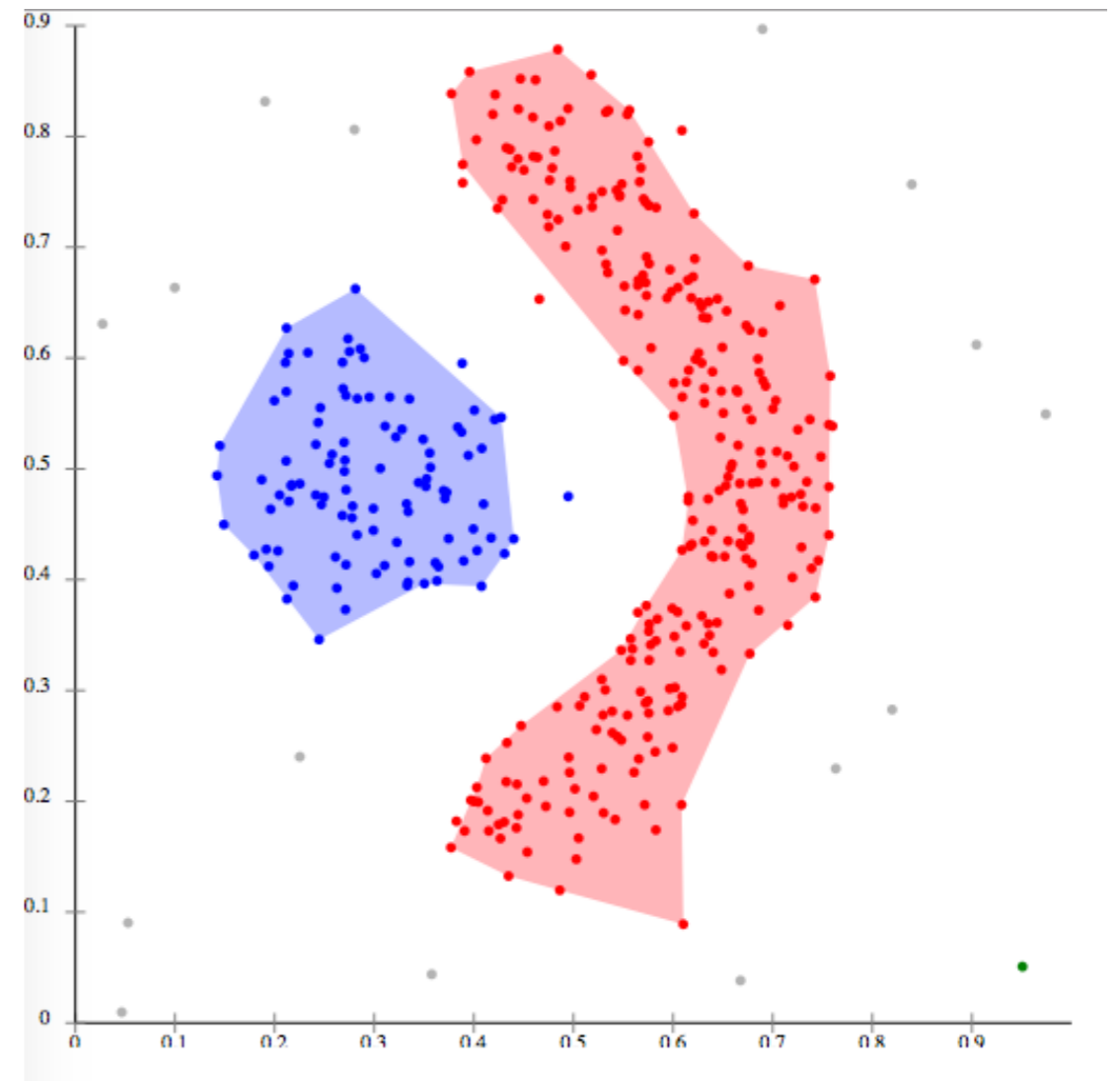
Received “test-of-time award” at KDD’14 — an extremely prestigious award.



Only need two parameters:

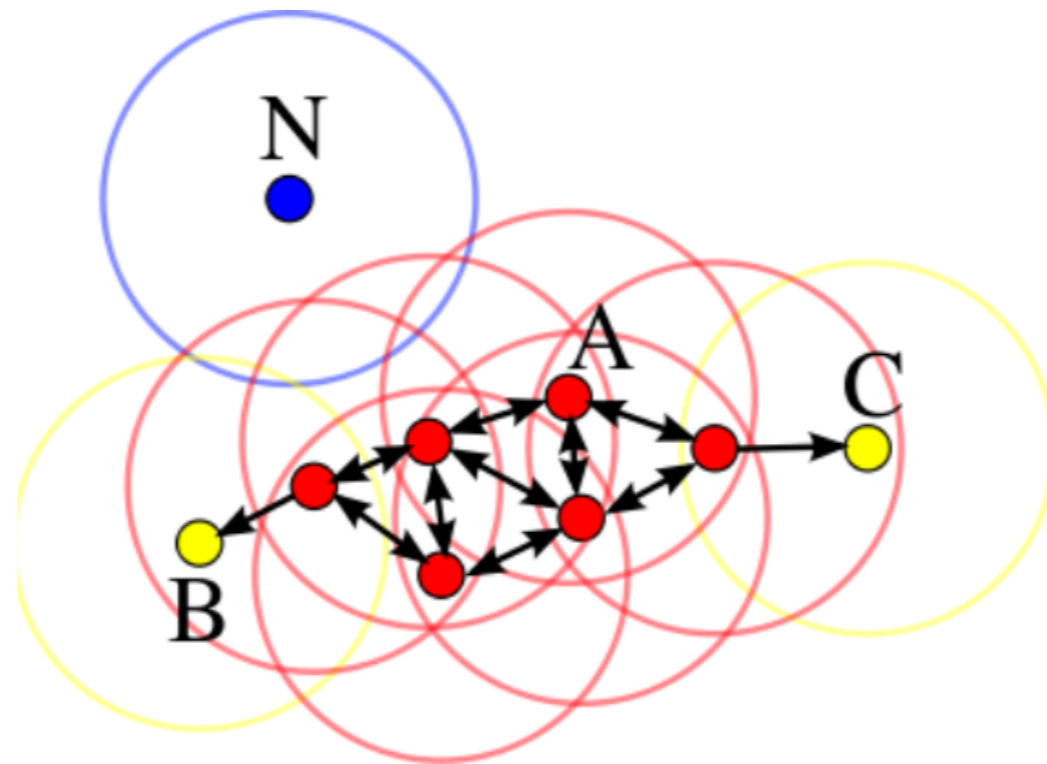
1. “radius” epsilon
2. minimum number of points (e.g., 4) required to form a dense region

Yellow “border points” are **density-reachable** from red “core points”, but not vice-versa.



# Interactive DBSCAN Demo

<https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/>



Only need two parameters:

1. “radius” epsilon

2. minimum number of points (e.g., 4) required to form a dense region

Yellow “border points” are **density-reachable** from red “core points”, but not vice-versa.

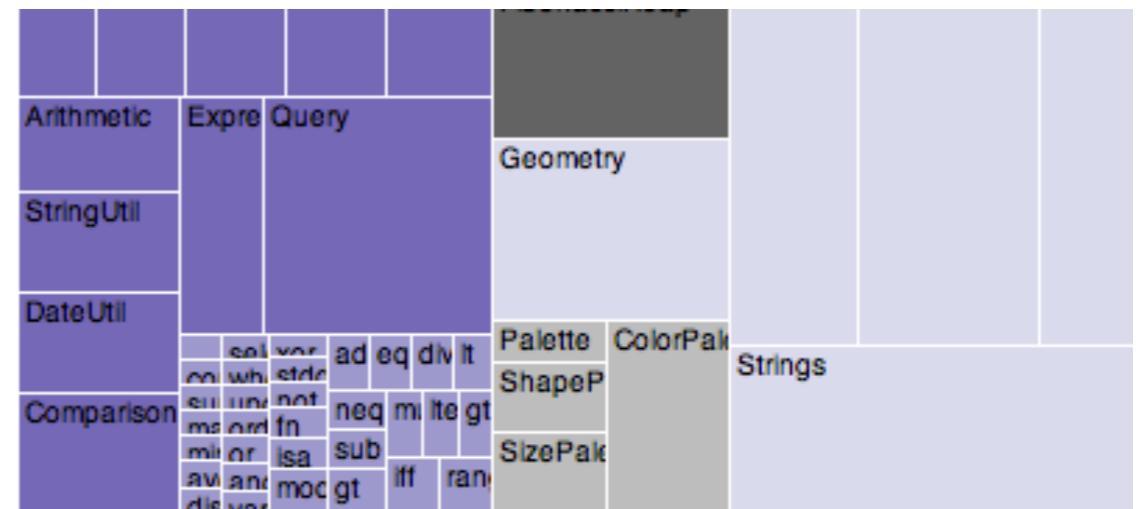
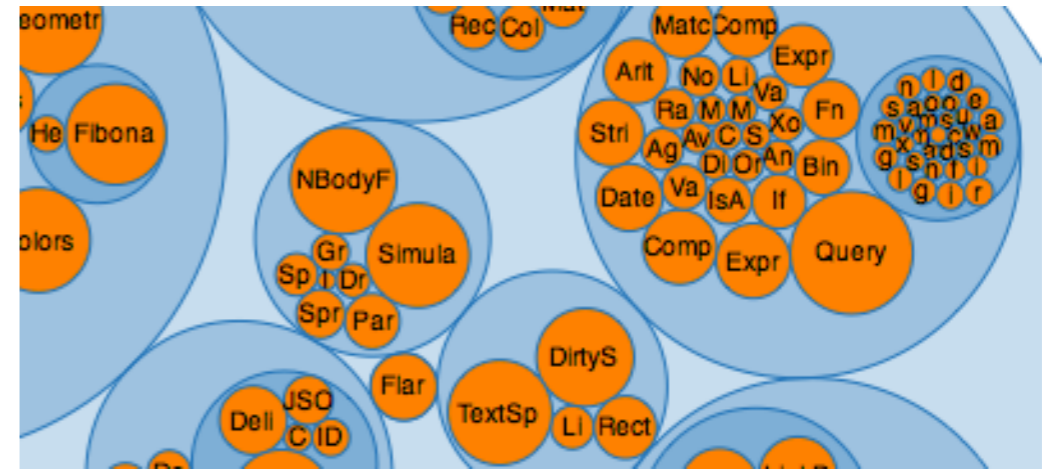
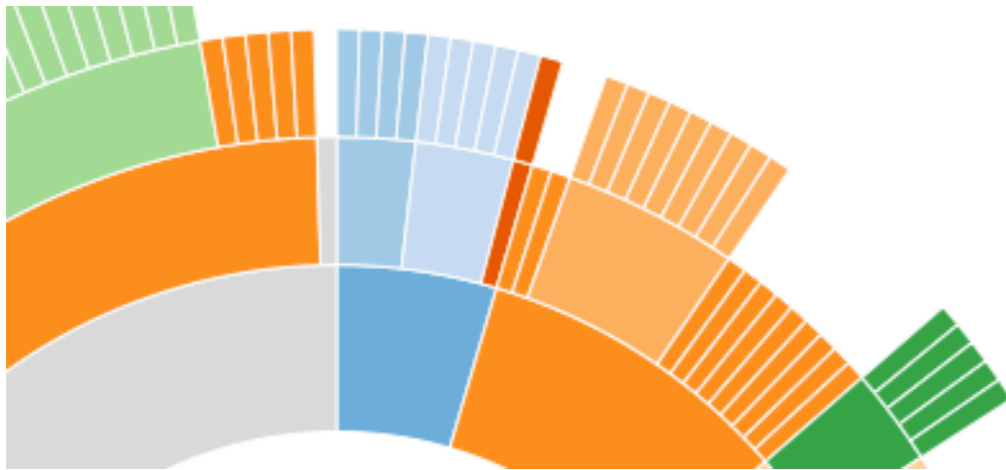
# You can use DBSCAN now.

[http://scikit-learn.org/stable/auto\\_examples/cluster/plot\\_dbscan.html](http://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html)

# Visualizing Clusters

# D3 has some built-in techniques

<https://github.com/mbostock/d3/wiki/Hierarchy-Layout>





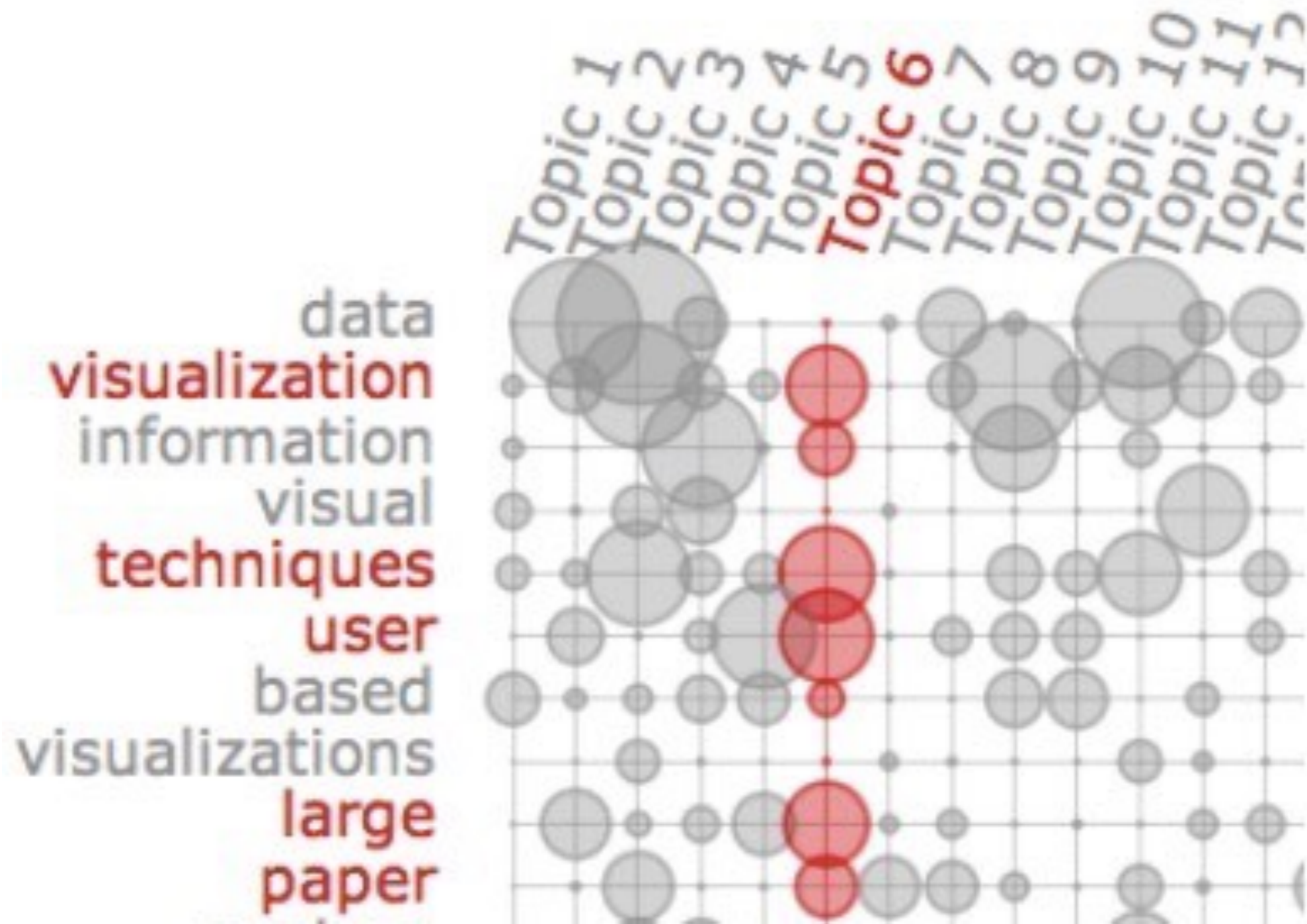


# Visualizing **Topics** as Matrix

**Termite: Visualization Techniques for Assessing Textual Topic Models**

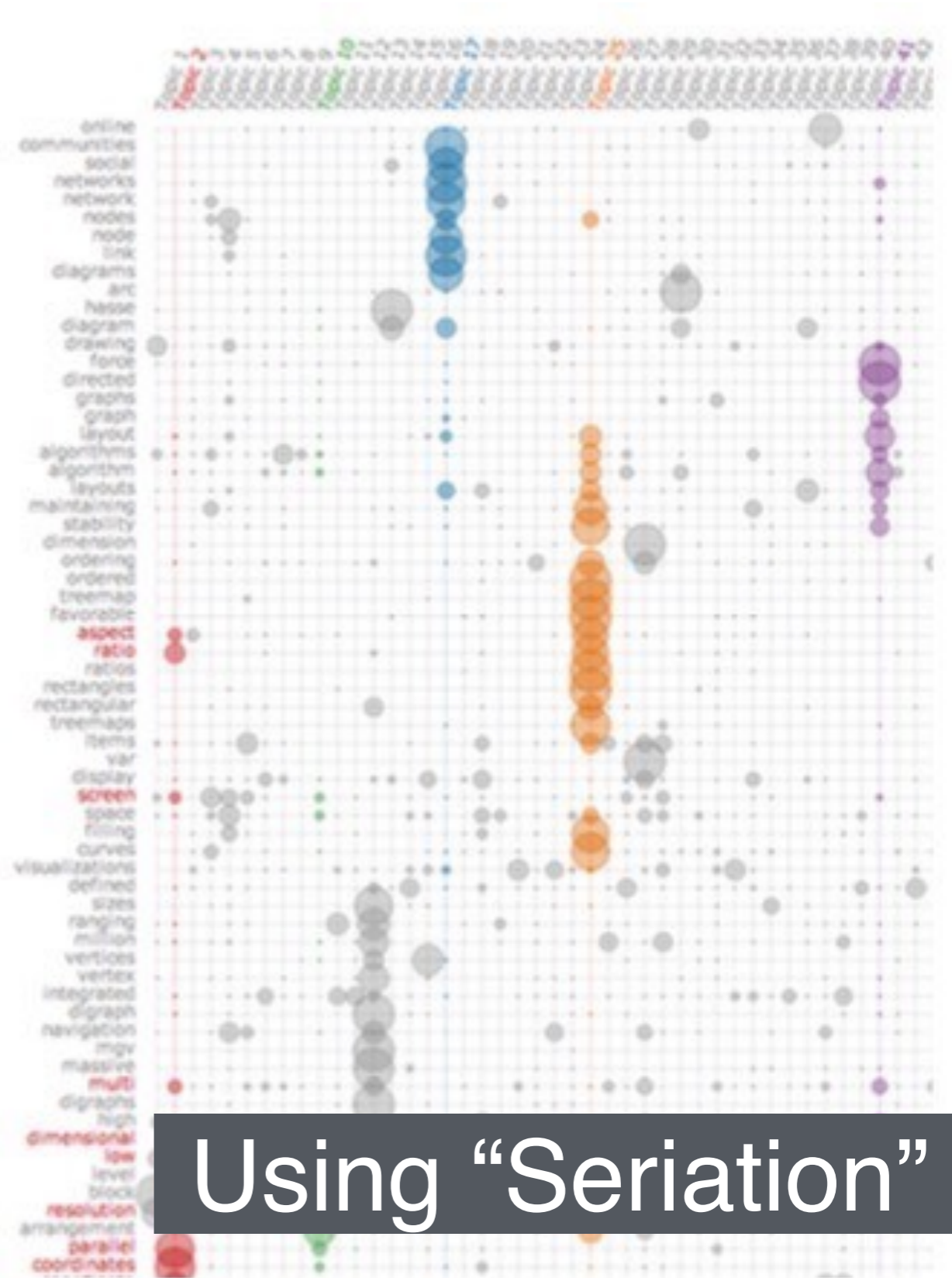
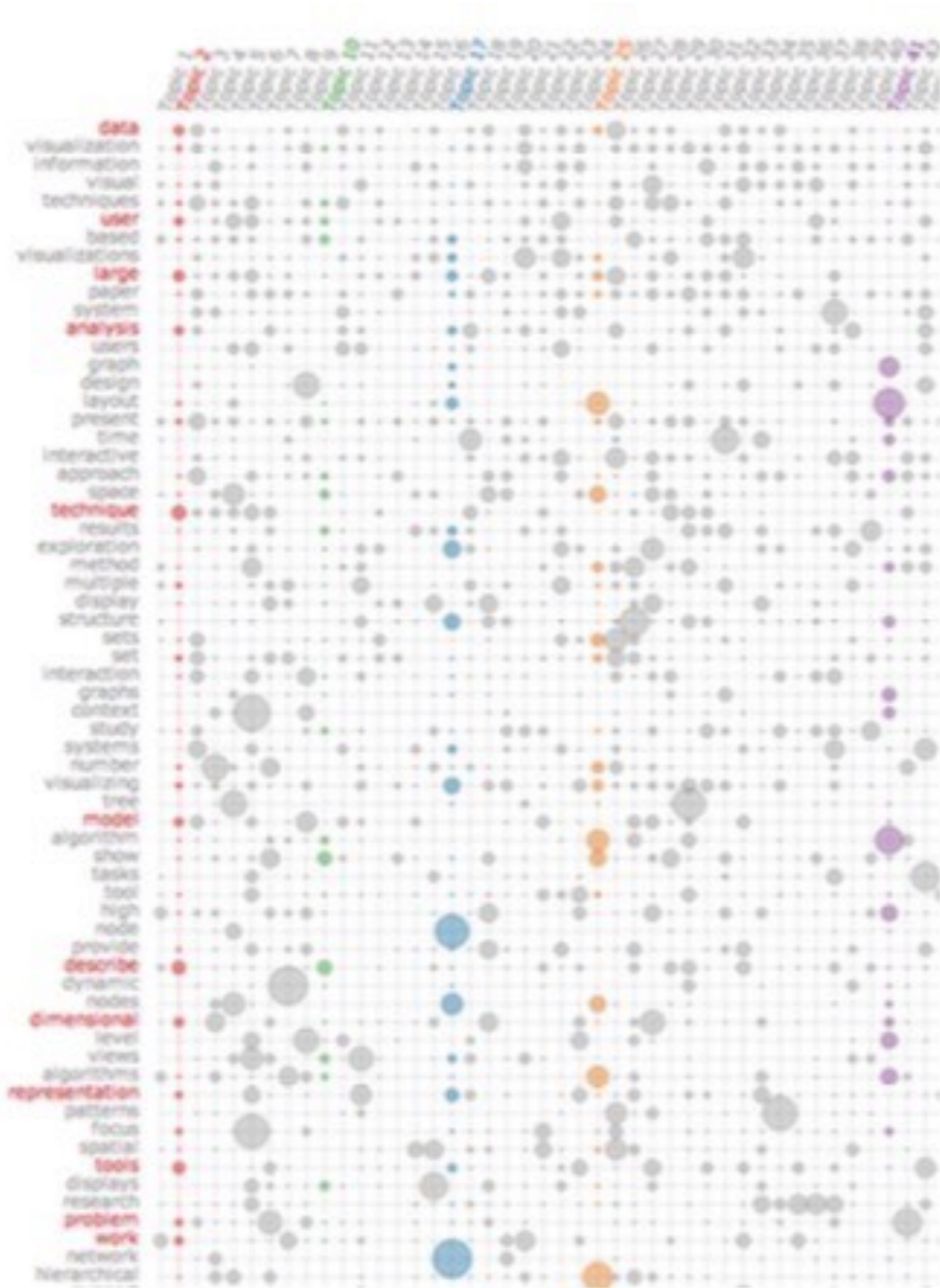
Jason Chuang, Christopher D. Manning, Jeffrey Heer. AVI 2012.

<http://vis.stanford.edu/papers/termite>



# Termite: Topic Model Visualization Analy

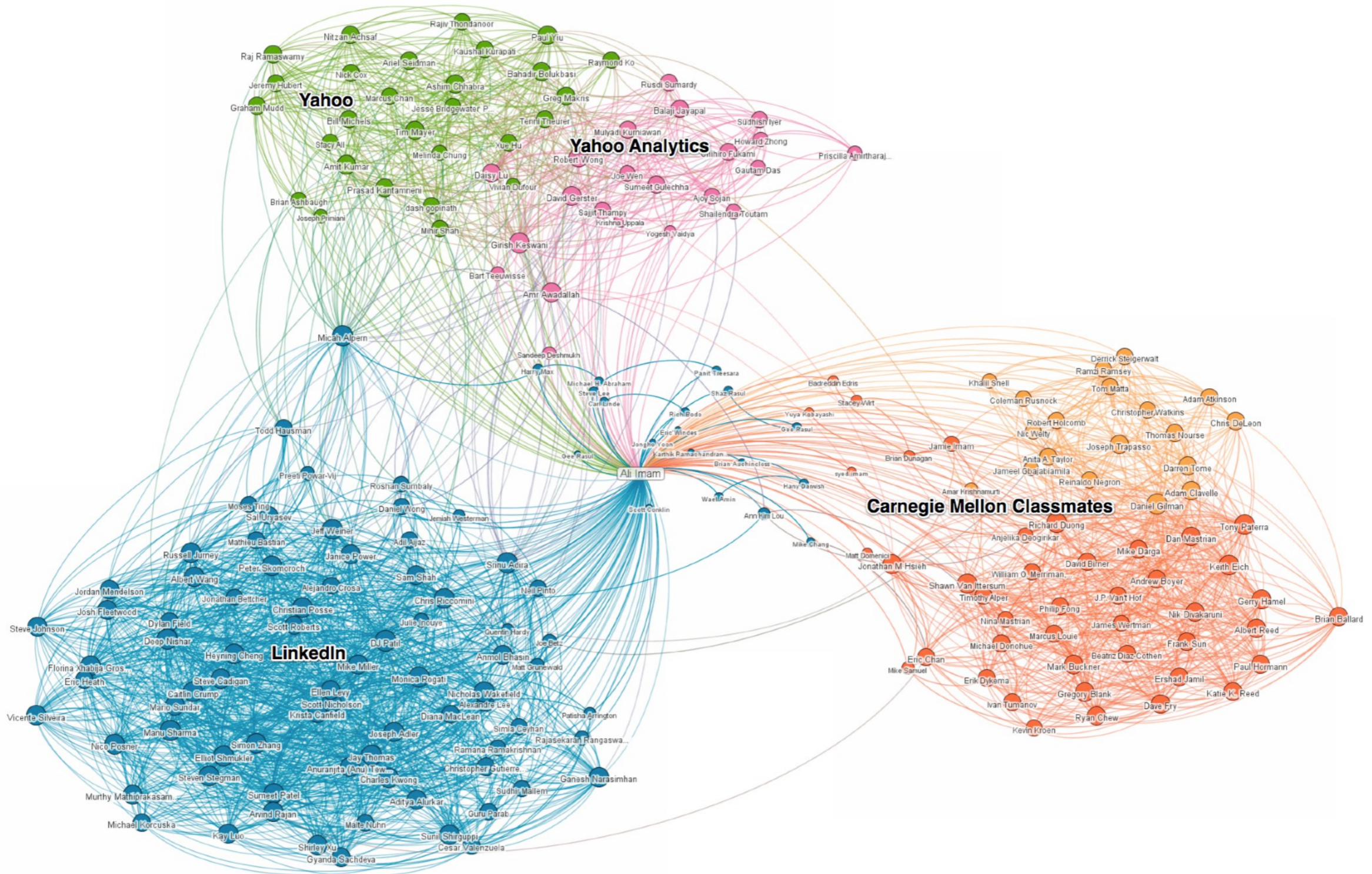
<http://vis.stanford.edu/papers/termite>



Using "Seriation"

# Visualizing Graph Communities

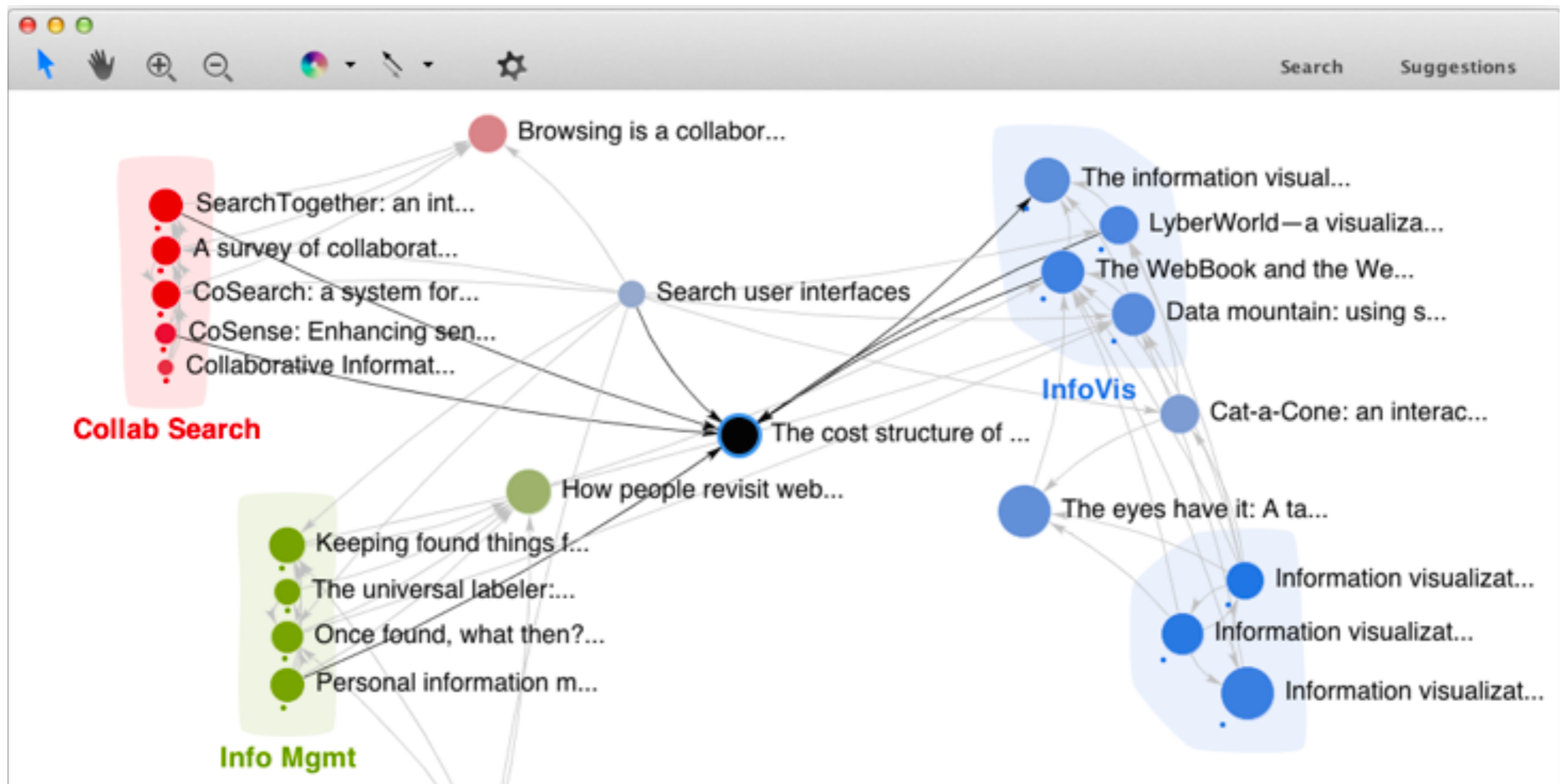
(using colors)



# Visualizing Graph Communities

(using colors and convex hulls)

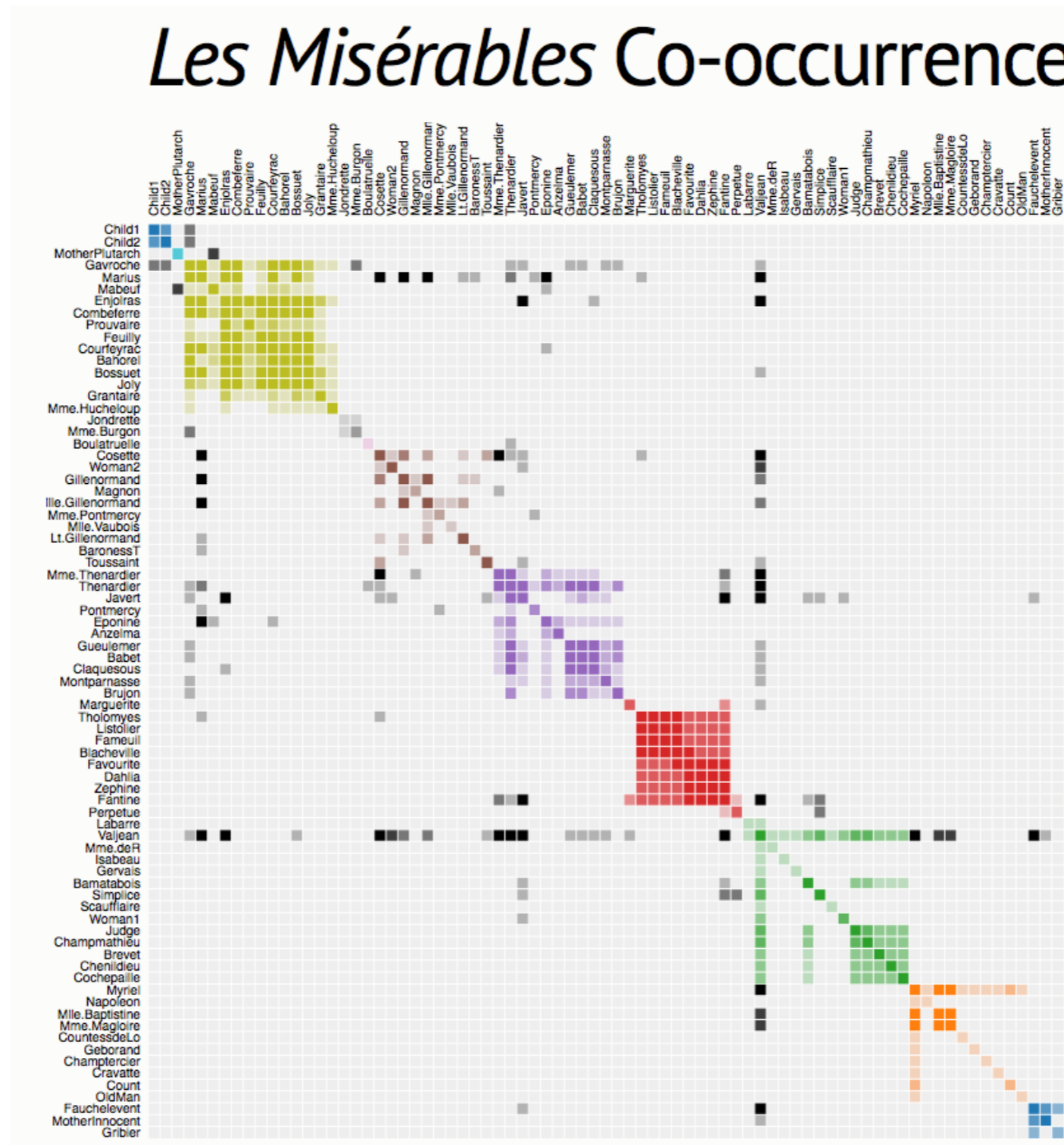
<http://www.cc.gatech.edu/~dchau/papers/11-chi-apolo.pdf>



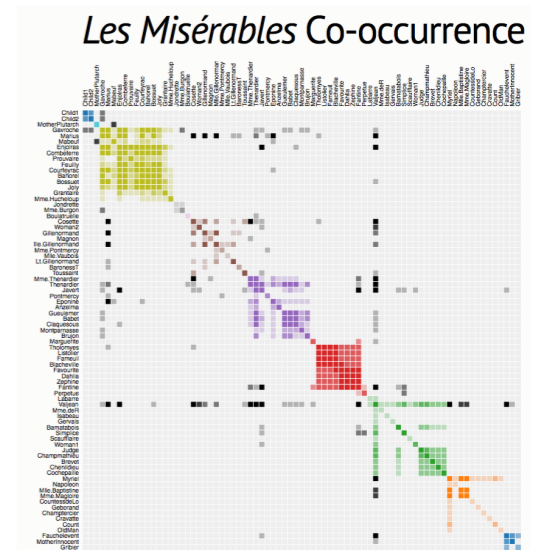
# Visualizing Graph Communities as Matrix

<https://bost.ocks.org/mike/miserables/>

Require good node ordering!



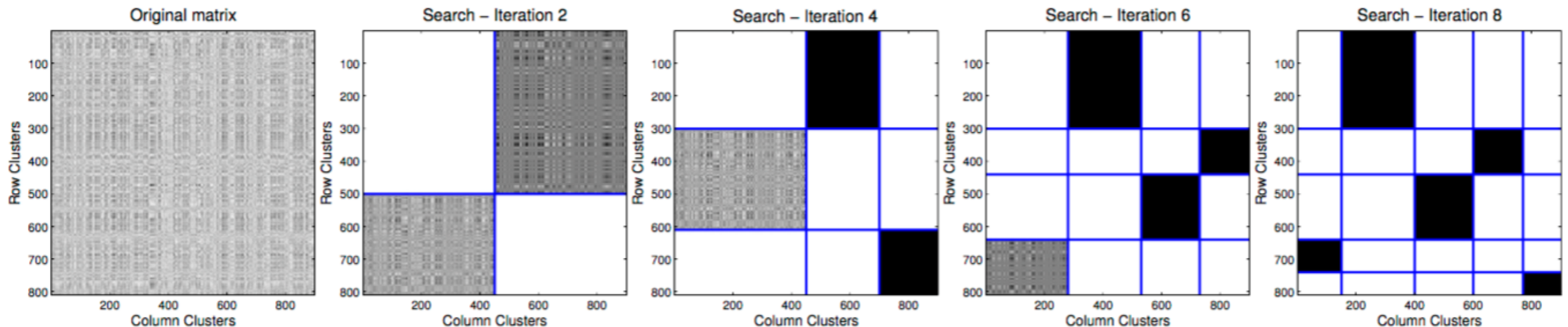
# Visualizing Graph Communities as Matrix



Require good node ordering!

Fully-automated way: “**Cross-associations**”

<http://www.cs.cmu.edu/~christos/PUBLICATIONS/kdd04-cross-assoc.pdf>



(a) Original matrix

(b) Iteration pair 1

(c) Iteration pair 2

(d) Iteration pair 3

(e) Iteration pair 4

# Graph Partitioning

If you know, or want to, specify #communities, use **METIS**, the most popular graph partitioning tools

<http://glaros.dtc.umn.edu/gkhome/views/metis>

