Tools for Exploratory Network Analysis

André Panisson
Data Science Laboratory - ISI Foundation
andre.panisson@isi.it

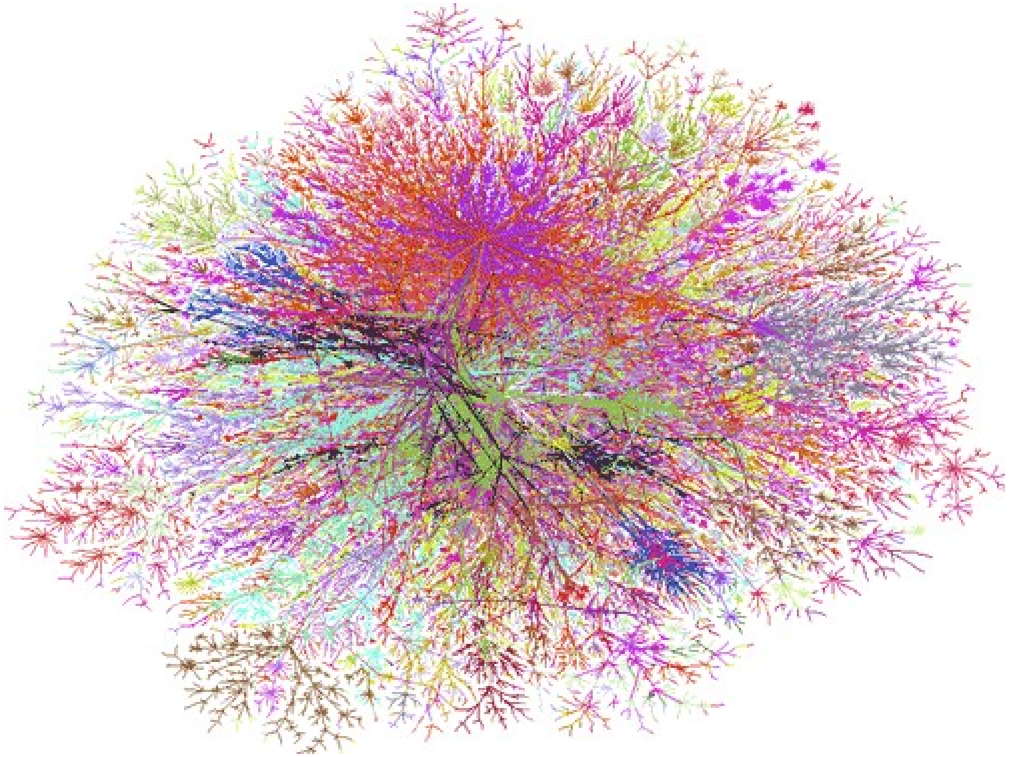
Complex Networks Thematic School, Les Houches, April 7-18, 2014

Networks are a mathematical tool to represent the backbone of complex systems that consists of many components.

Understanding the structure of the networks help us to understand the way the components interact with each other and the intrinsic properties of the complex system.

Challenge:

How to explore networks that are complex and big?



The Internet Backbone

Network Analysis Tools

Other Challenges:

- Network exploration

  Visualization is essential in data exploration
- Pattern matching

  How to efficiently create, render, and interact with complex networks?
- Scaling

  How to explore massive network data?

Information Visualization Mantra

“Overview first, zoom and filter, then details-on-demand.”

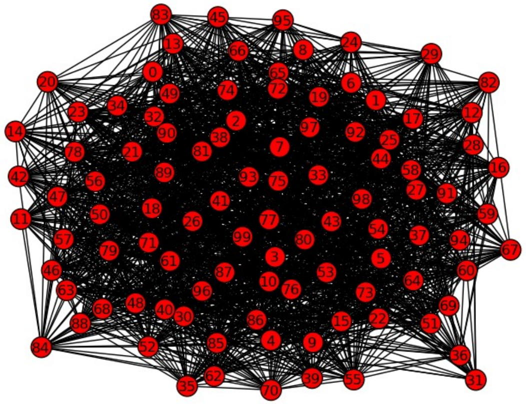
Ben Schneiderman

NetworkX

Implemented in Python

```python
import networkx as nx
import random

n = 100
p = 0.2

G = nx.Graph()

for node in range(100):
    G.add_node(node)

for source in range(100):
    for target in range(100):
        if source == target: continue
        if random.random() < p:
            G.add_edge(source, target)
```

```python
import matplotlib.pyplot as plt
nx.draw(G)
plt.show()
```



NetworkX

- Generative models:

```
K_5 = nx.complete_graph(5)
K_3_5 = nx.complete_bipartite_graph(3,5)


er = nx.erdos_renyi_graph(100,0.15)
ws = nx.watts_strogatz_graph(30,3,0.1)
ba = nx.barabasi_albert_graph(100,5)
```

- Basic Analysis:

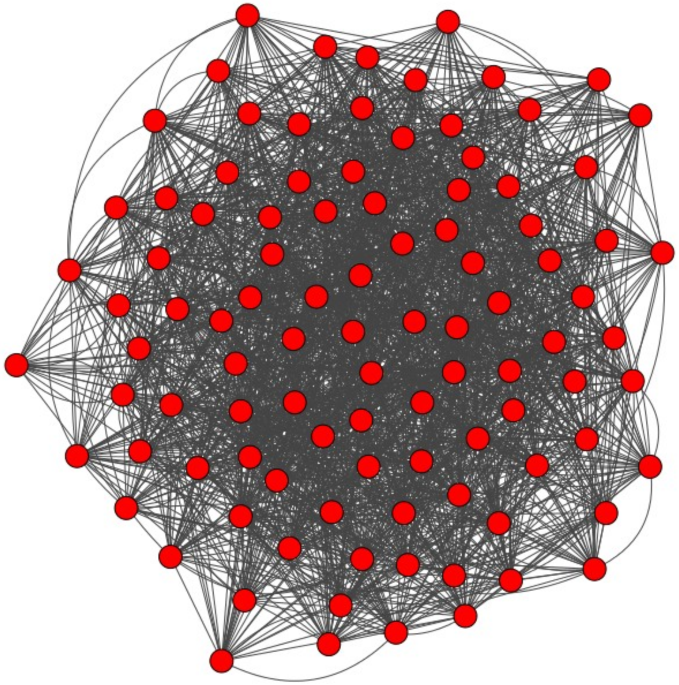
```
nx.degree(G)
nx.connected_components(G)
nx.clustering(G)
nx.betweenness_centrality(G)
nx.degree_assortativity_coefficient(G)
```

IGraph

Library written in C/C++

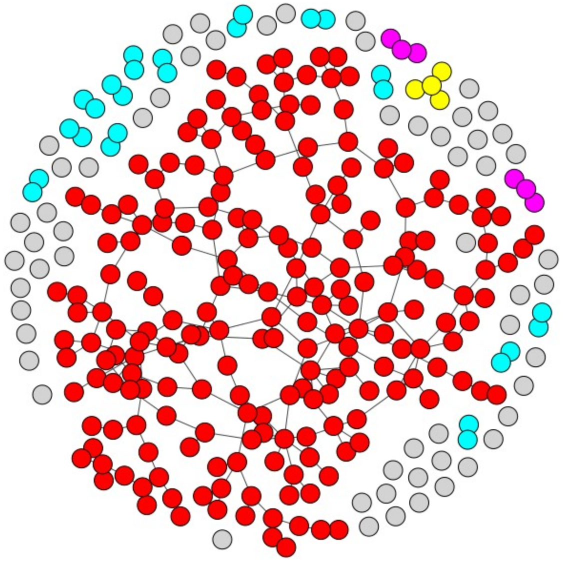
Mature interfaces to GNU R and Pythond Python

```
from igraph import *
import random

p = 0.2
n = 100

G = Graph(n)

for source in range(100):
    for target in range(100):
        if source == target: continue
        if random.random() < p:
            G.add_edges((source, target))

plot(G, layout="fr", vertex_label=None)
```



IGraph

```
from igraph import *

g = Graph.Erdos_Renyi(n=300, m=250)
colors = ["lightgray", "cyan", "magenta", "yellow", "blue", "green", "red"]
for component in g.components():
  color = colors[min(6, len(component)-1)]
  for vidx in component: g.vs[vidx]["color"] = color

plot(g, layout="fr", vertex_label=None)
```



```
g = Graph.Erdos_Renyi(100, 0.2)
g = Graph.Barabasi(100,2)

g.degree()
g.betweenness()
g.edge_betweenness()
g.pagerank()  [1]
c = g.community_infomap()  [2]
```

[1] Page, Lawrence and Brin, Sergey and Motwani, Rajeev and Winograd, Terry (1999) The PageRank Citation Ranking: Bringing Order to the Web. Technical Report. Stanford InfoLab.

[2] M. Rosvall and C. T. Bergstrom, Maps of information flow reveal community structure in complex networks, PNAS 105, 1118 (2008)

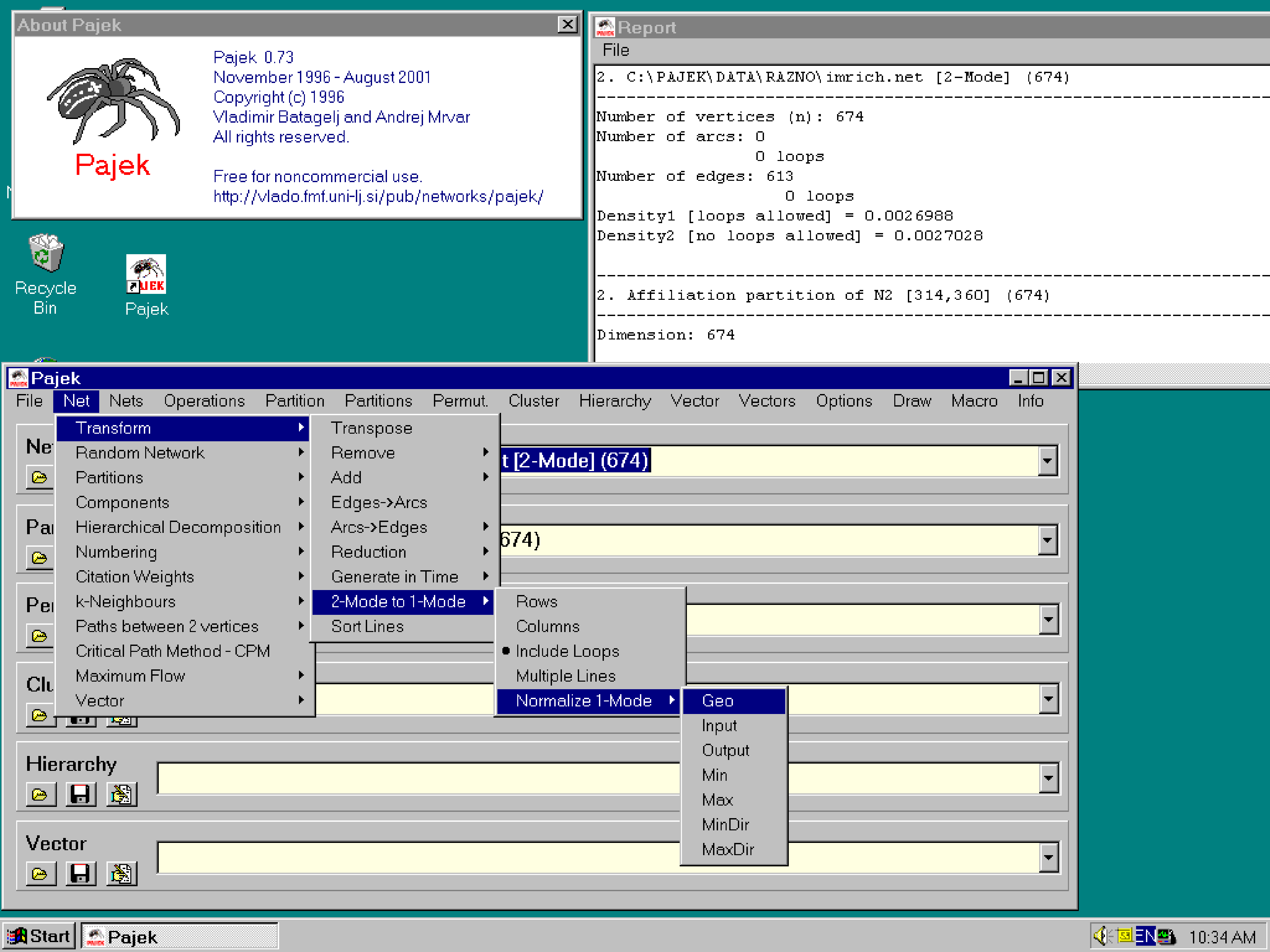
Pajek

Pajek is one of the first visual exploratory tools for graph visualization and analysis

It is freely available for noncommercial use

Available for Windows

http://vlado.fmf.uni-lj.si/pub/networks/pajek/



About Pajek

Pajek 0.73
November 1996 - August 2001
Copyright (c) 1996
Vladimir Batagelj and Andrej Mrvar
All rights reserved.

Free for noncommercial use.
http://vlado.fmf.uni-lj.si/pub/networks/pajek/

Report

```
2. C:\PAJEK\DATA\RAZNO\imrich.net [2-Mode] (674)
------------------------------------------------------------------
Number of vertices (n): 674
Number of arcs: 0
                0 loops
Number of edges: 613
                   0 loops
Density1 [loops allowed] = 0.0026988
Density2 [no loops allowed] = 0.0027028

------------------------------------------------------------------
2. Affiliation partition of N2 [314,360] (674)
------------------------------------------------------------------
Dimension: 674
```

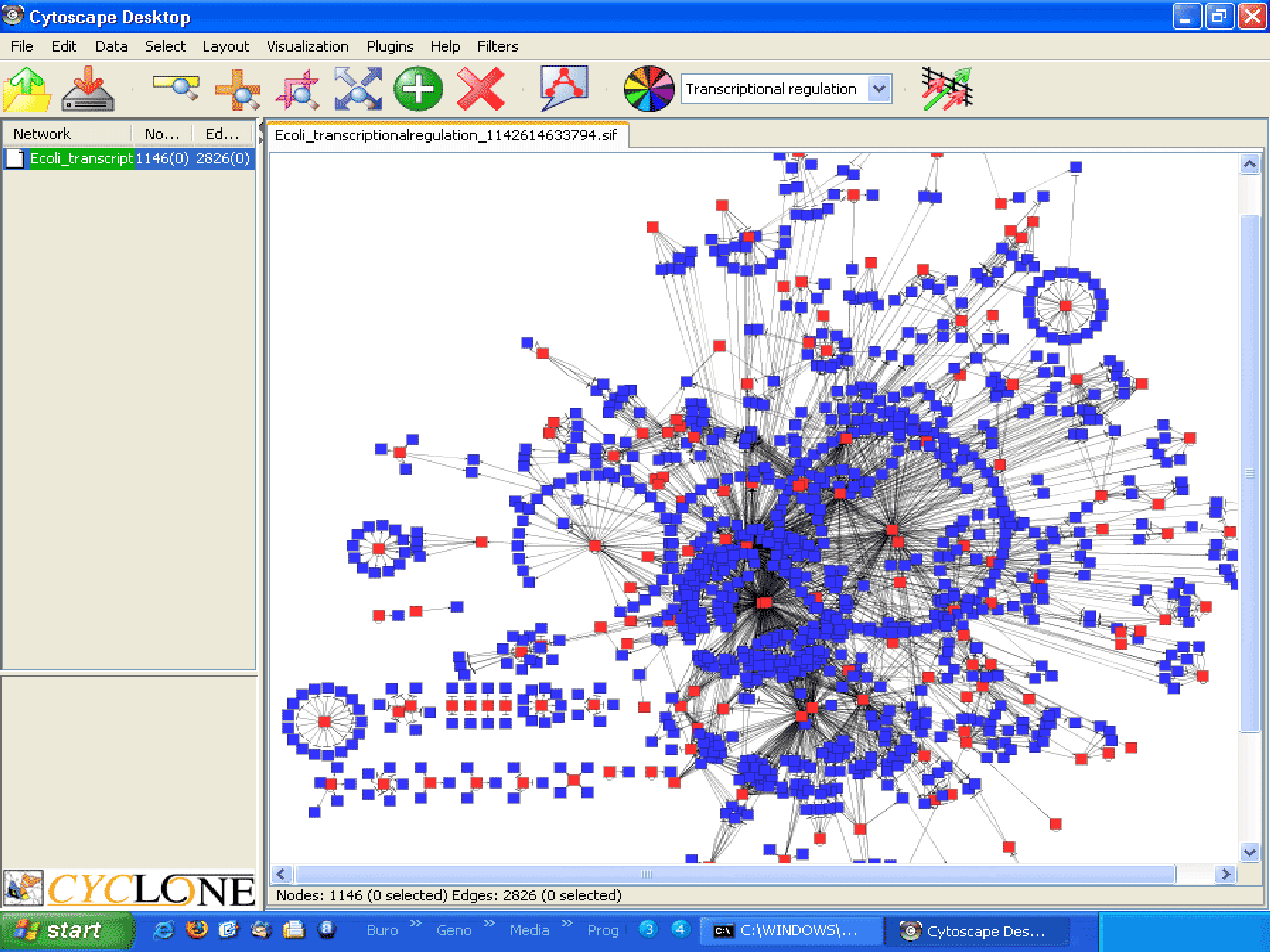
Cytoscape

Open source software, mainly used for manipulation of biomolecular interaction networks

Cytoscape support large databases of protein-protein, protein-DNA, and genetic interactions, available for humans and model organisms

It is also an extensible software, with a plug-in architecture, allowing the development of additional features.

http://www.cytoscape.org



Network	No...	Ed...
Ecoli_transcript	1146(0)	2826(0)

Ecoli_transcriptionalregulation_1142614633794.sif

Nodes: 1146 (0 selected) Edges: 2826 (0 selected)

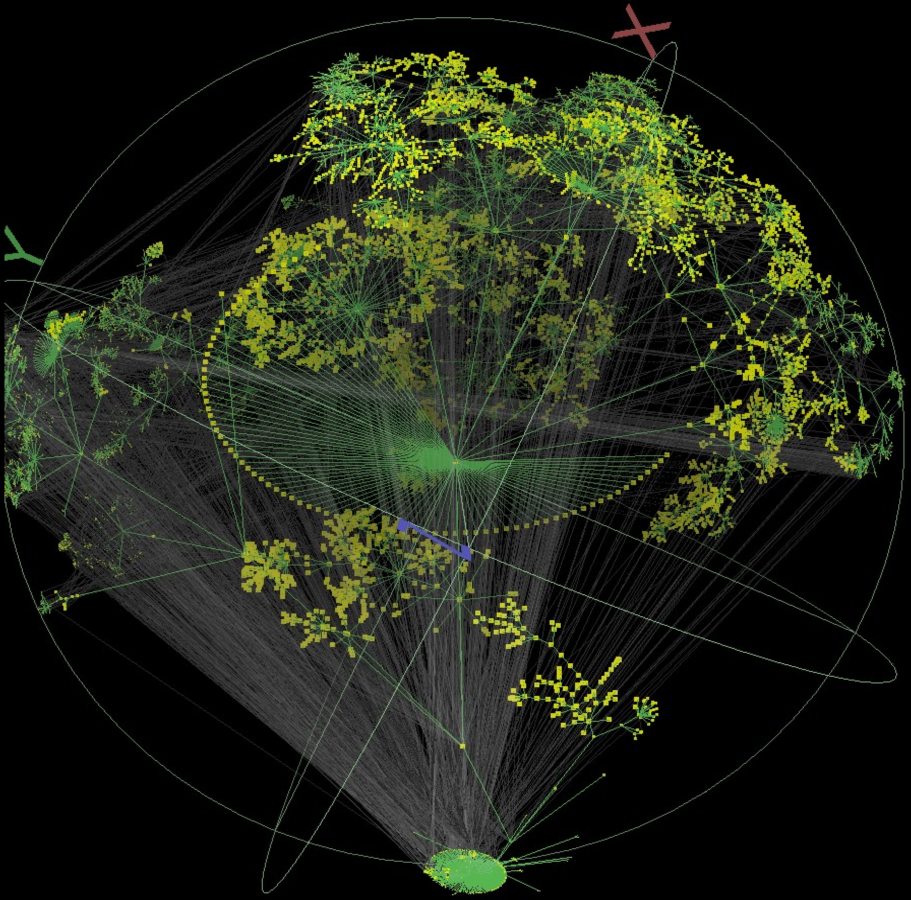
Walrus

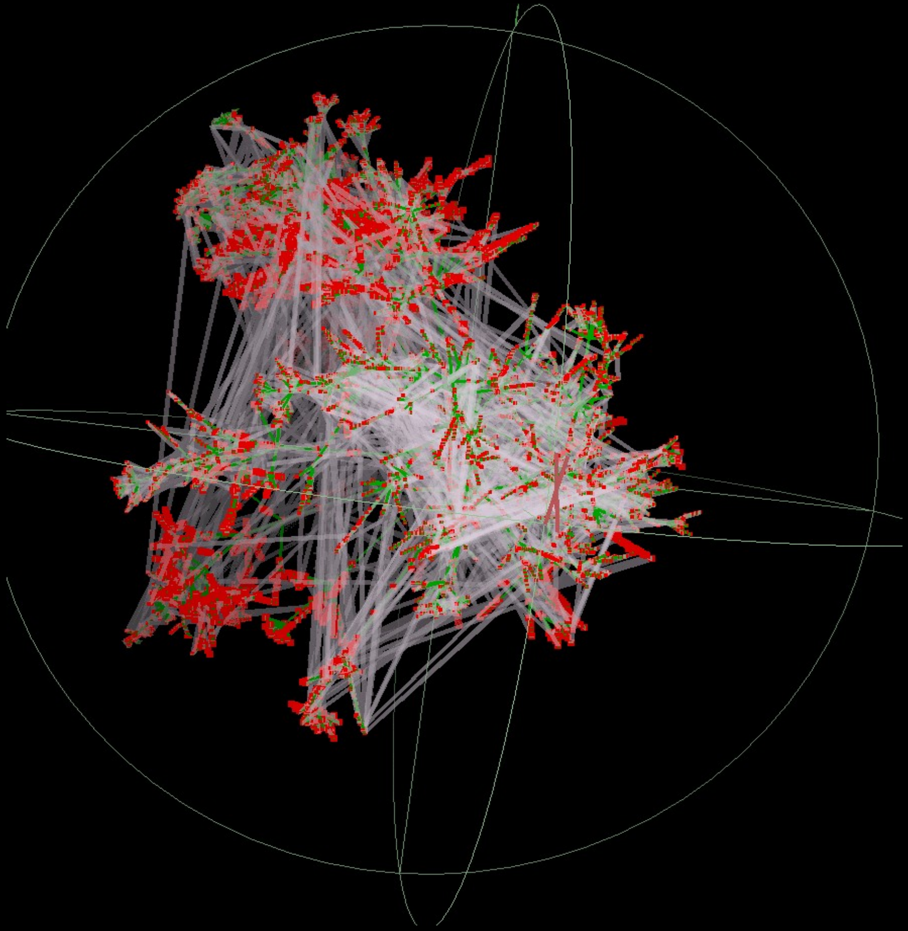
A tool that visualizes graphs using a 3D engine, based on their spanning tree representation.

Walrus is best suited to visualizing graphs that are nearly trees

It is also an extensible software, with a plug-in architecture, allowing the development of additional features

http://www.caida.org/tools/visualization/walrus/





GUESS

A visualization and analysis tool based on Gython

Gython: an extension of Python; a domain-specific language that supports operators that can deal directly with graph structures

http://graphexploration.cond.org/

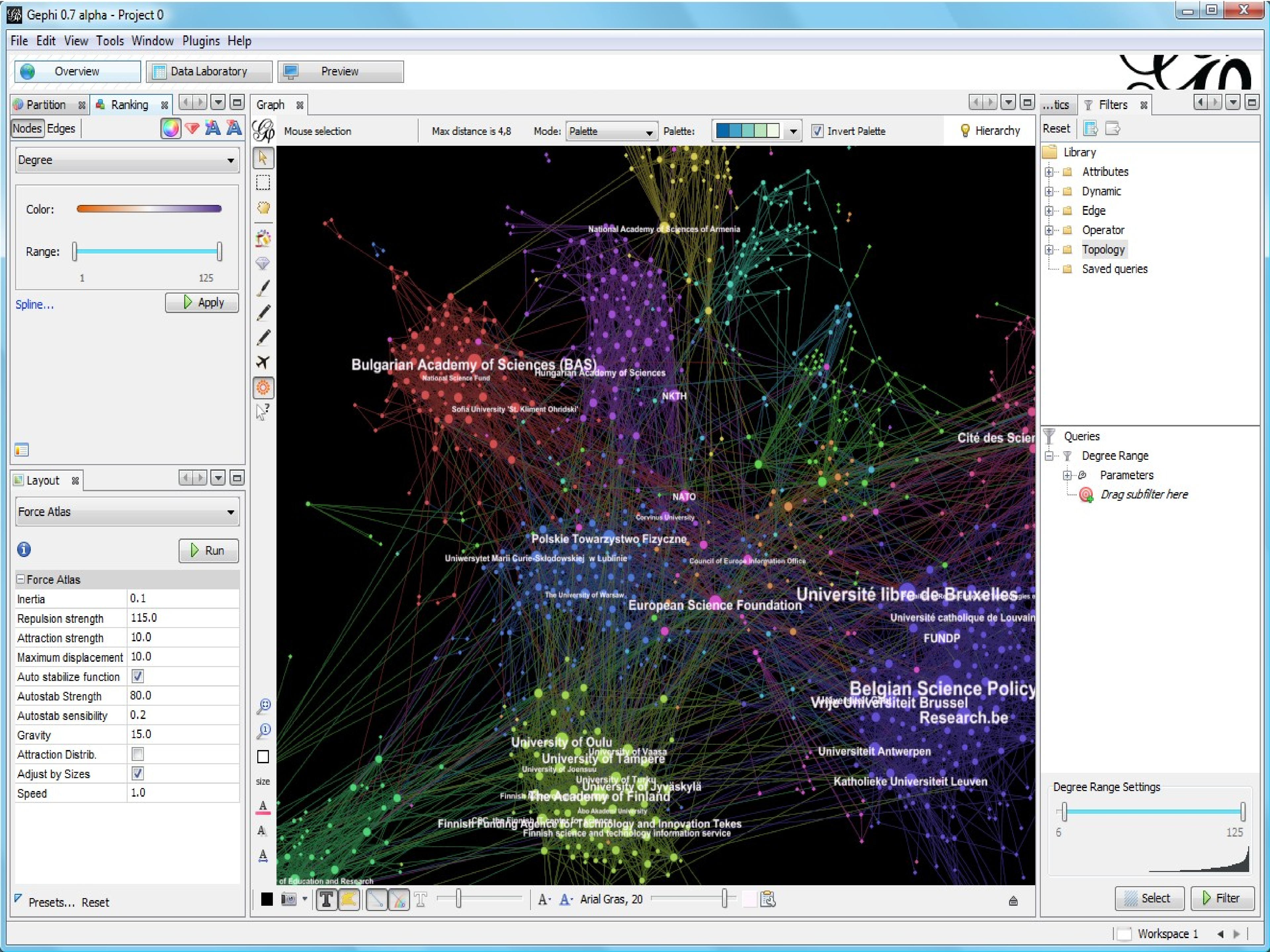
Gephi

http://gephi.org

An open source software based on the Netbeans platform, specialized in graph analysis and visualization.

For visualization of large networks, it uses Java OpenGL, that speeds up the exploration and realtime rendering.

The key features allow spatializing, filtering, navigating, manipulating and clustering graphs.



Gephi 0.7 alpha - Project 0
File Edit View Tools Window Plugins Help
Overview
Data Laboratory
Preview
Partition
Ranking
Nodes
Edges
Spline...
Apply
Layout
Run
Presets... Reset
Graph
Hierarchy
...tics
Filters
Reset
Library
Attributes
Dynamic
Edge
Operator
Topology
Saved queries
Queries
Degree Range
Parameters
Drag subfilter here
Select
Filter
Workspace 1
size
Arial Gras, 20
National Academy of Sciences of Armenia
Bulgarian Academy of Sciences (BAS)
National Science Fund
Hungarian Academy of Sciences
Sofia University 'St. Kliment Ohridski'
NKTH
Cité des Scien
NATO
Corvinus University
Polskie Towarzystwo Fizyczne
Uniwersytet Marii Curie-Skłodowskiej w Lublinie
Council of Europe Information Office
The University of Warsaw
European Science Foundation
Université libre de Bruxelles
Université catholique de Louvain
FUNDP
Belgian Science Policy
Vrije Universiteit Brussel
Research.be
University of Oulu
University of Vaasa
University of Tampere
University of Joensuu
University of Turku
University of Jyväskylä
The Academy of Finland
Åbo Akademi University
Universiteit Antwerpen
Katholieke Universiteit Leuven
Finnish Funding Agency for Technology and Innovation Tekes
CSC, the Finnish IT center for science
Finnish science and technology information service
of Education and Research

Degree

Color:

Range: 1 125

Force Atlas

Force Atlas	
Inertia	0.1
Repulsion strength	115.0
Attraction strength	10.0
Maximum displacement	10.0
Auto stabilize function	☑
Autostab Strength	80.0
Autostab sensibility	0.2
Gravity	15.0
Attraction Distrib.	☐
Adjust by Sizes	☑
Speed	1.0

Mouse selection Max distance is 4,8 Mode: Palette Palette: ☑ Invert Palette

Degree Range Settings

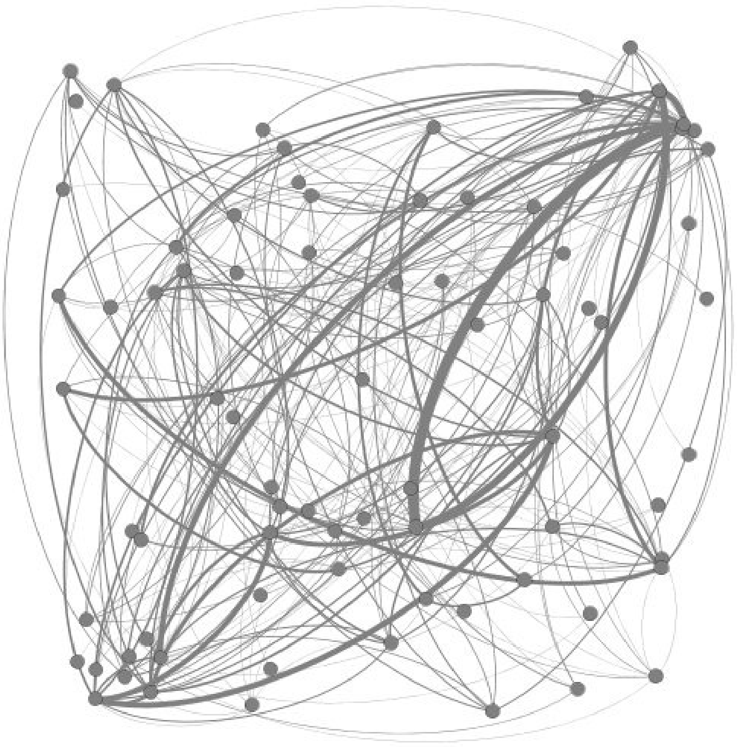
6 125

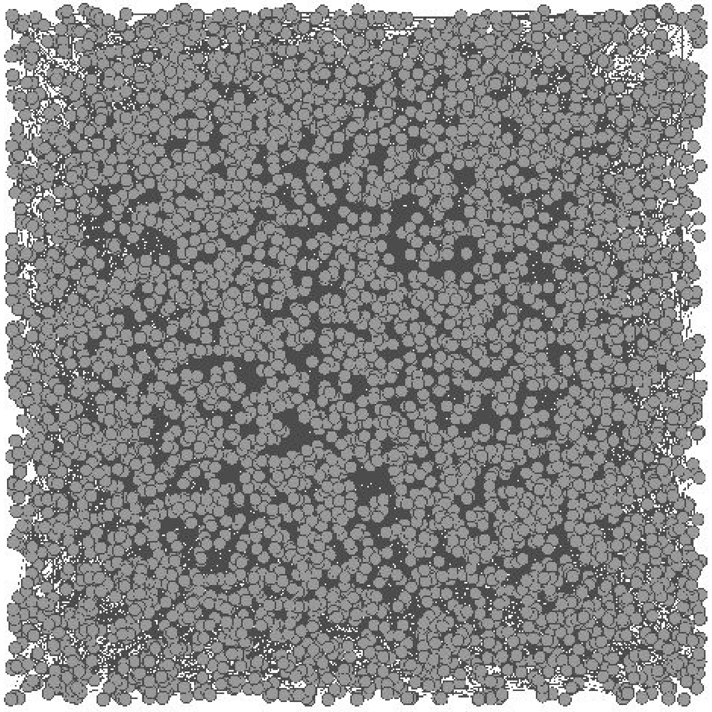
Gephi Examples

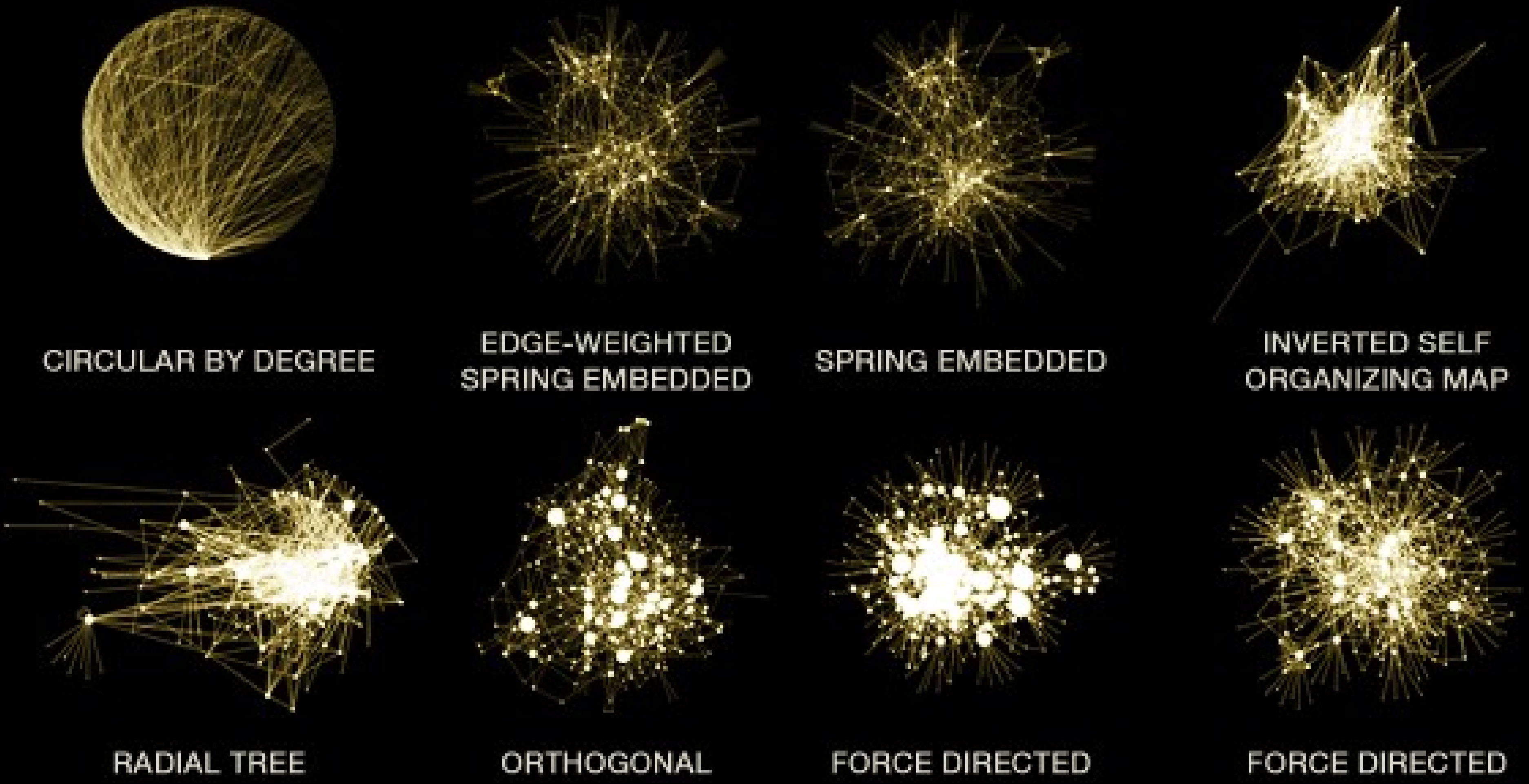
- Working with large networks
  - Layout
  - Filtering
  - Colors
  - Size

Layouts

Random Layout

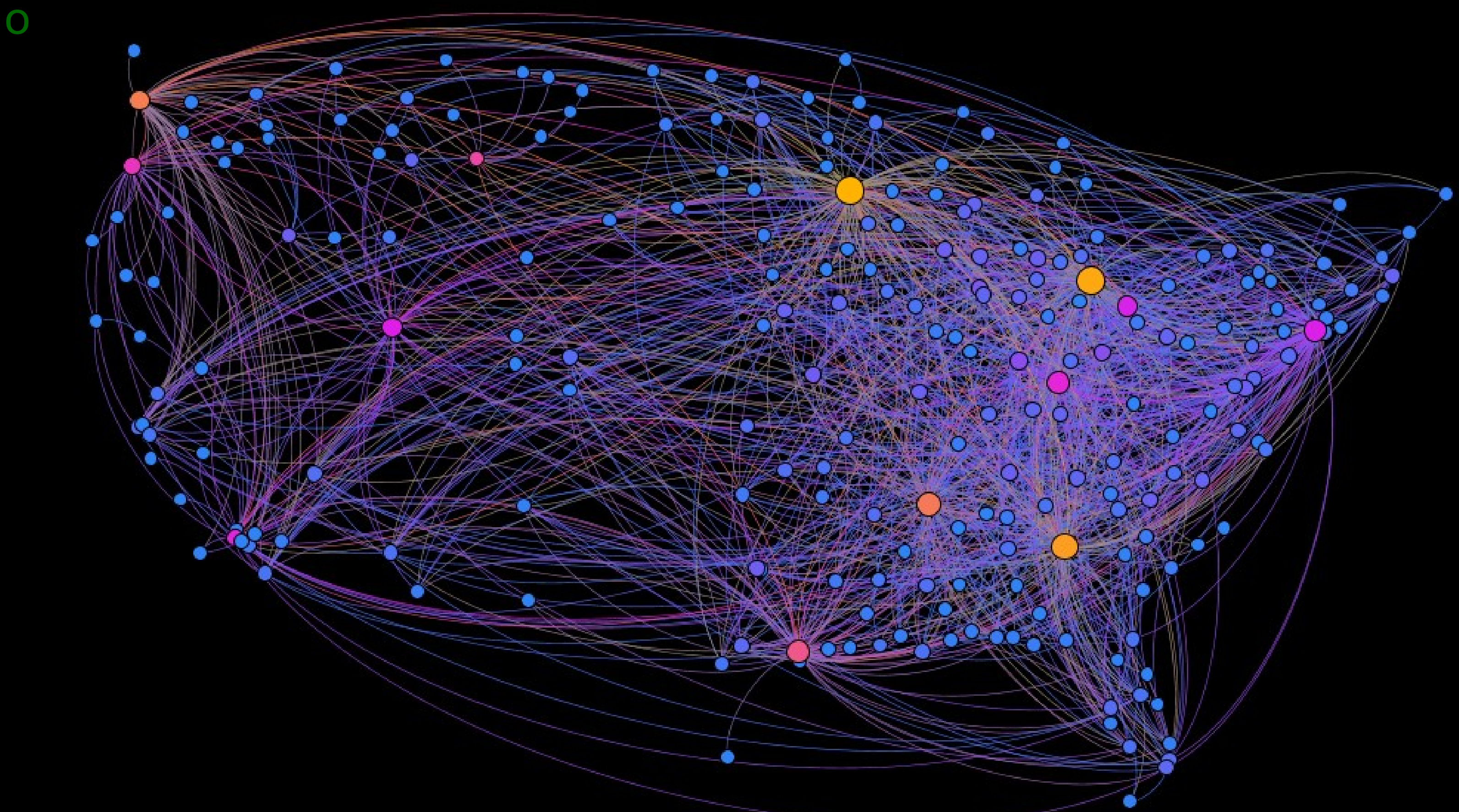


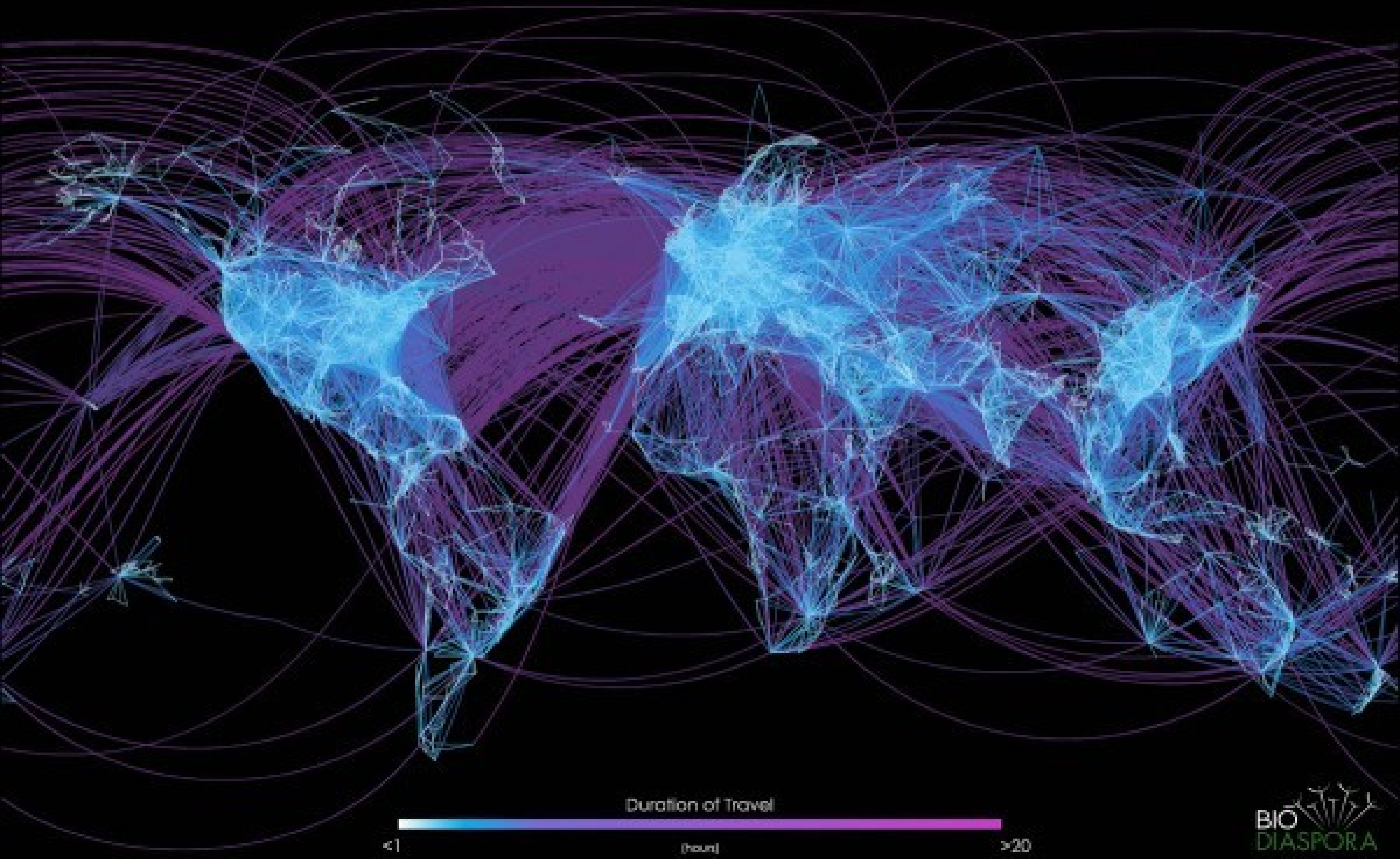


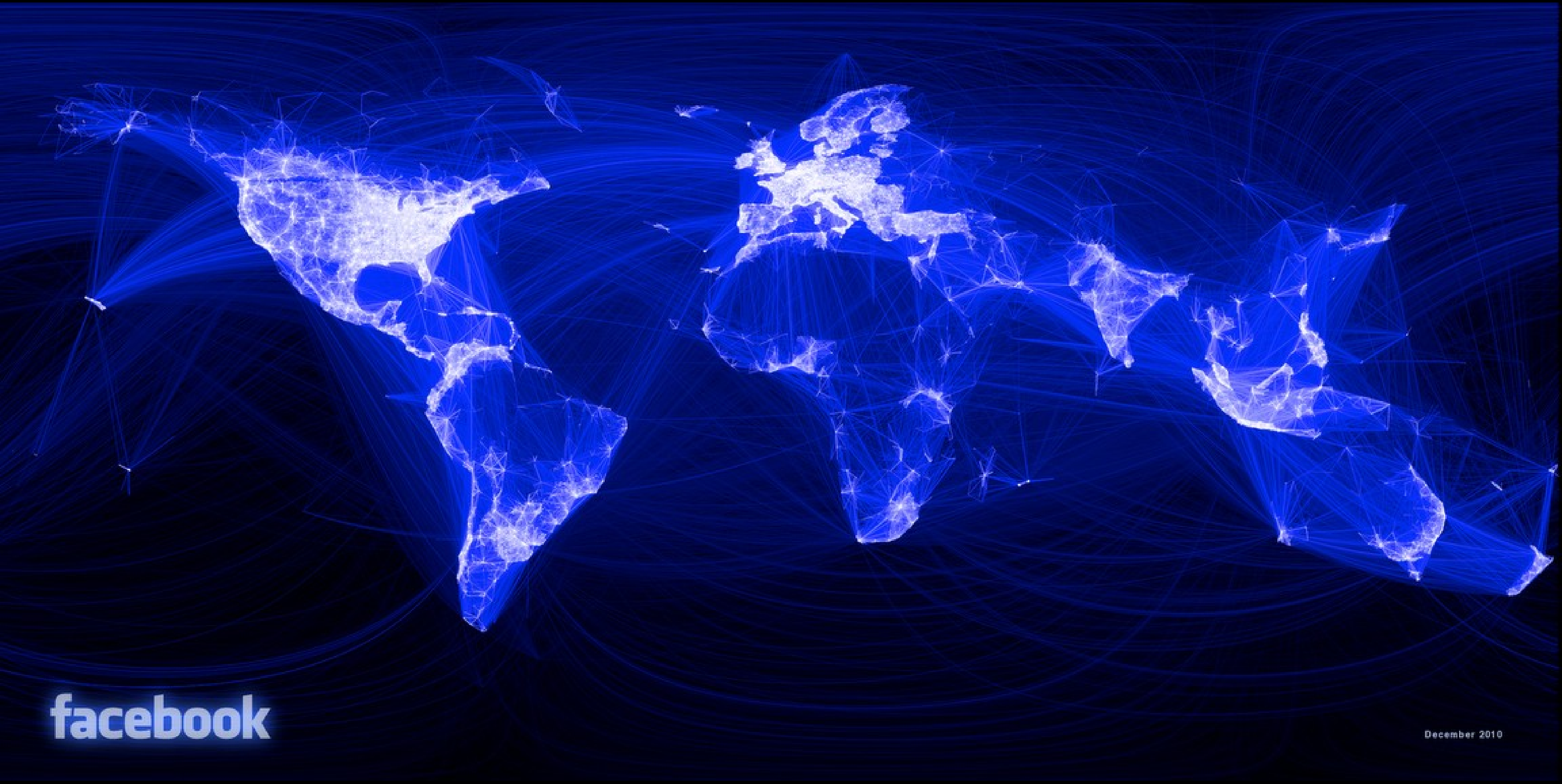


All layout representations are of **the same network**!

Geo Layout

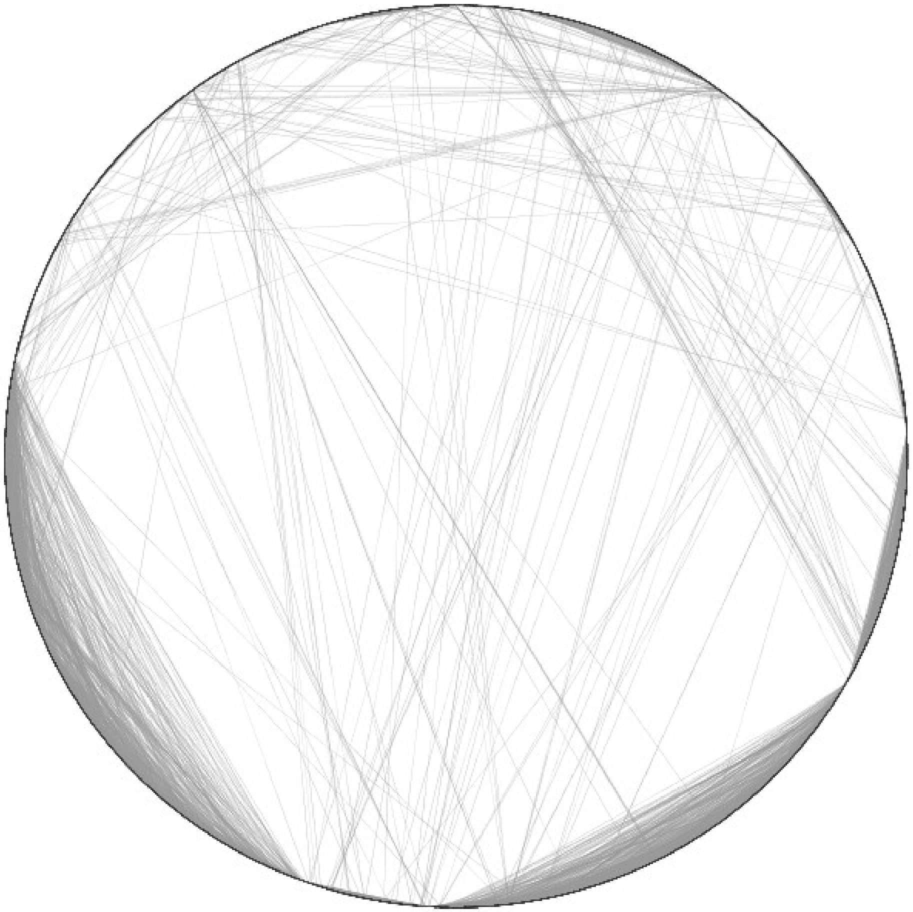



Duration of Travel
<1
(hours)
>20
BIO
DIASPORA



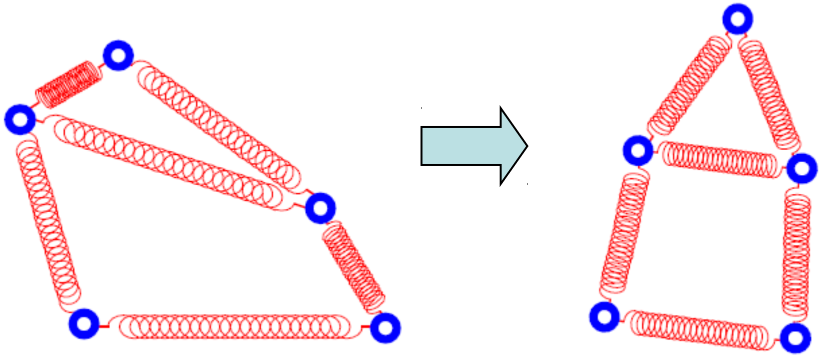
facebook
December 2010

Circular Layout



Force-directed methods

- A graph is treated as a system of entities with attraction and repulsion forces acting between them
- The algorithm seeks a configuration with locally minimal energy, i.e. , a position for every entity such that the sum of the forces on each entity is zero



Force-Directed Graph Layout

```
set up initial node velocities to (0,0)
set up initial node positions randomly
loop
    total_kinetic_energy := 0 // running sum of total kinetic energy over all particles
    for each node
        net-force := (0, 0) // running sum of total force on this particular node

        for each other node
            net-force := net-force + repulsion( node, other node )
        next node

        for each spring connected to this node
            net-force := net-force + attraction( node, spring )
        next spring

        // without damping, it moves forever
        node.velocity := (node.velocity + timestep * net-force) * damping
        node.position := node.position + timestep * node.velocity
        total_kinetic_energy := total_kinetic_energy + node.mass * $(node.velocity)^2$
    next node
until total_kinetic_energy is less than some small number
```

What happens with disconnected components? Solution: Center of gravity
What happens if damping/timestep are too big? Flickering / bad convergence

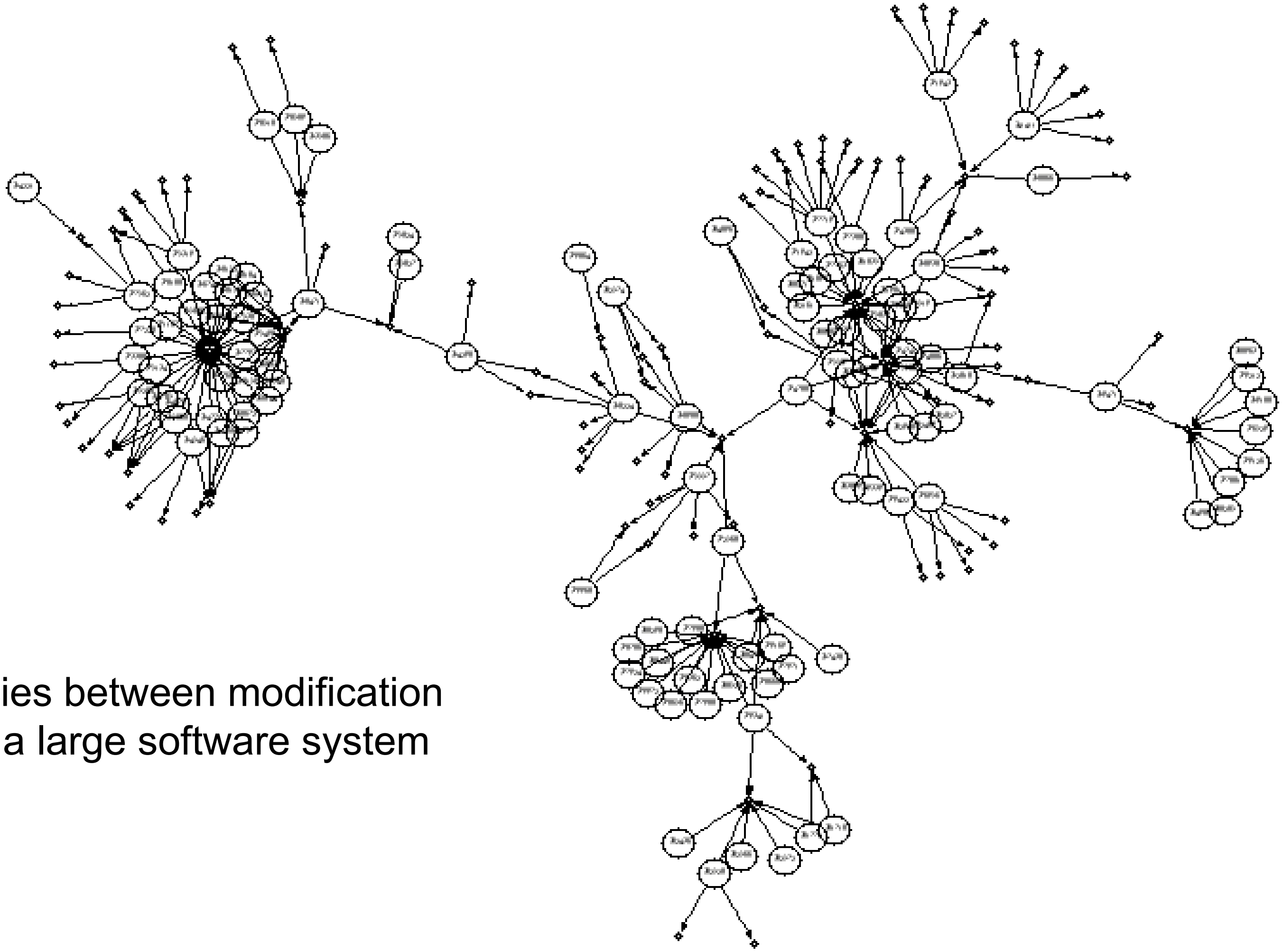
Graph Layouts

- Examples:
  - **Fruchterman**, Thomas M. J.; **Reingold**, Edward M. (1991), "Graph Drawing by Force-Directed Placement". Software – Practice & Experience (Wiley) 21 (11): 1129–1164
  - **Yifan Hu**, (2006), “Efficient, High-Quality Force-Directed Graph Drawing”, The Mathematica Journal, vol. 10, issue 1
  - S. Martin, W. M. Brown, R. Klavans, and K. Boyack (2011), "**OpenOrd**: An Open-Source Toolbox for Large Graph Layout," SPIE Conference on Visualization and Data Analysis (VDA)

OpenOrd

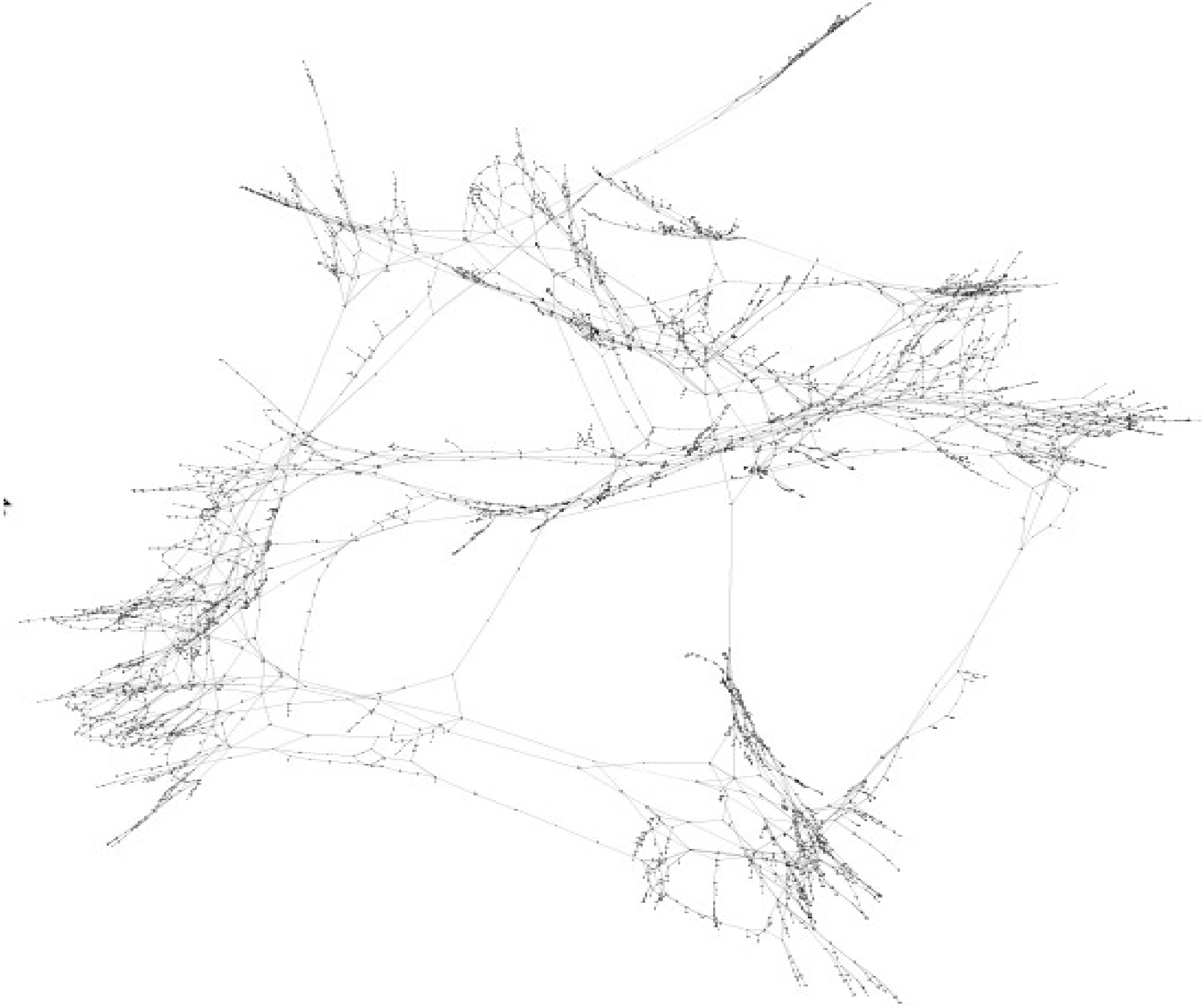
- Available as a plug-in for Gephi
- Based on a previously implemented closed-source algorithm known as VxOrd
- Uses a multi-stage approach (liquid, expansion, cool-down, crunch, and simmer)
- Good performance in large networks (100k – 1M nodes)

Force-directed layout



Dependencies between modification requests in a large software system

Aesthetics: distance, dispersion



OpenOrd