Experiences and Challenges in Large Scale Graph Drawing

Yifan Hu

AT&T Labs Research
Florham Park, NJ 07005, USA
yifanhu@research.att.com

Bertinoro Workshop on Graph Drawing, March 9-14, 2008

Outline of the talk

- Algorithms for large (undirected) graphs
- Applying to some large graphs from the real world
- Challenges and an open question

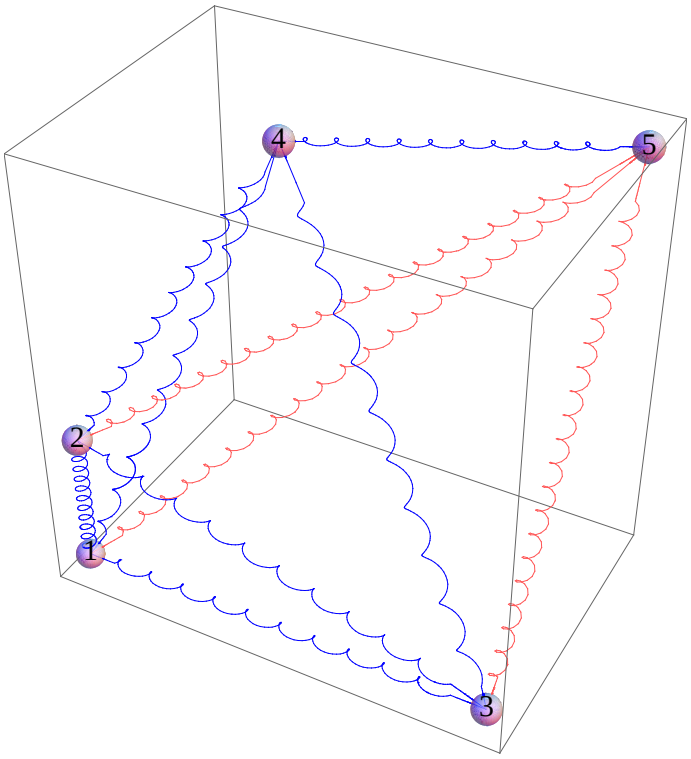
Spring embedding

T. Kamada and S. Kawai (1989).

A spring between every pair of vertices

Ideal spring length = graph distance

{1 -> 2, 2 -> 3, 3 -> 1, 1 -> 4, 2 -> 4, 3 -> 4, 4 -> 5}



Spring embedding

$$\text{stress}(x) = \sum_{i \neq j, i, j \in V} w_{ij} \left(\| x_i - x_j \| - d(i, j) \right)^2$$

d_{ij} is the ideal distance, taken to be the graph distance,
between i
and j

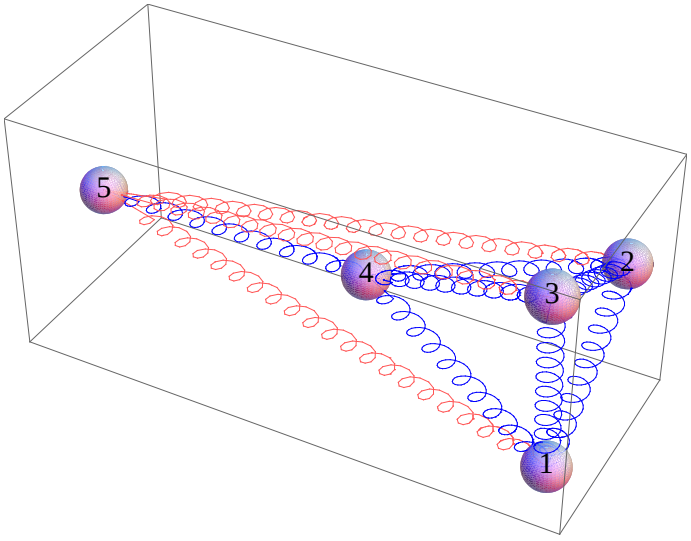
Force on a vertex i is proportional to the difference to the ideal distance

$$F(i) = \sum_{j \neq i} w_{ij} \left(\| x_i - x_j \| - d_{ij} \right) \vec{ij}$$

Computationally/memory expensive. Complexity $O\left(|V|^2 \right)$. But may be reduced...

Stress majorization (Garsner, Koren & North, 2005) (details later)

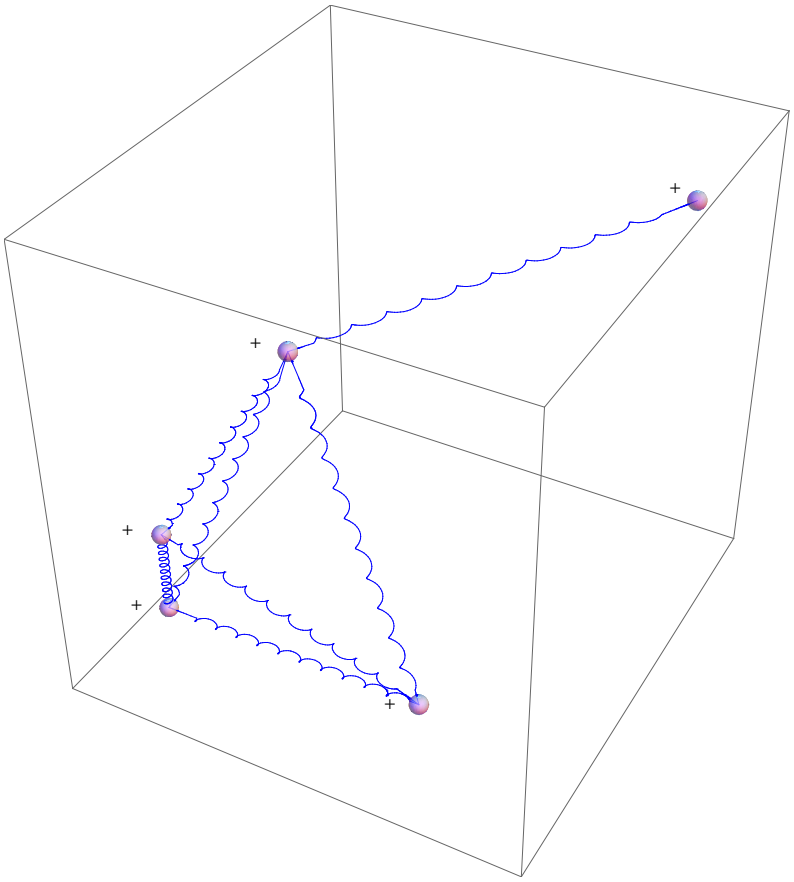
{1 -> 2, 2 -> 3, 3 -> 1, 1 -> 4, 2 -> 4, 3 -> 4, 4 -> 5}



Forced directed algorithm (spring-electrical embedding)

T. M. J. Fruchterman and E. M. Reigold (1991)

{1 -> 2, 2 -> 3, 3 -> 1, 1 -> 4, 2 -> 4, 3 -> 4, 4 -> 5}



Forced directed algorithm (spring-electrical embedding)

The attractive spring force: proportional to the square of the distance

$$F_a(i,\ j)\ =\ \frac{||x_i - x_j||^2}{K}\ \vec{\text{ij}},\ \ i \leftrightarrow\ j$$

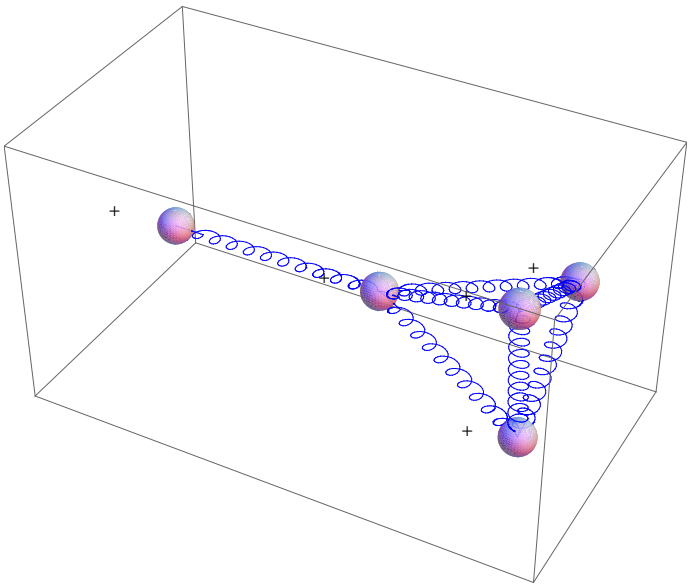
The repulsive electrical force: proportional to the inverse of the distance

$$F_r(i,\ j)\ =\ -\frac{K^2}{||x_i - x_j||}\ \vec{\text{ij}}\ ,\ i \neq j$$

The energy to minimize is

$$E\,(x) = \sum_{i \leftrightarrow j}\ \frac{2}{3K}\,||\,x_i - x_j\,||^3 - \sum_{i \neq j} K^2 \ln\Big(||\,x_i - x_j\,||\Big)$$

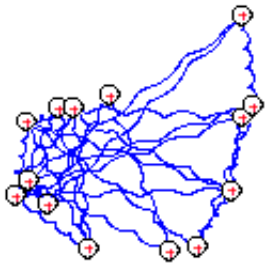
Complexity can be reduced to $O(|\,V\,|\ \log|\,V\,|)$ by approximate long range repulsive force



Force directed algorithm: iterative process

The complicated form of the energy for spring-electrical embedding means that right now the best algorithm is iterative: steepest decent

- **The iterative forced directed algorithm**



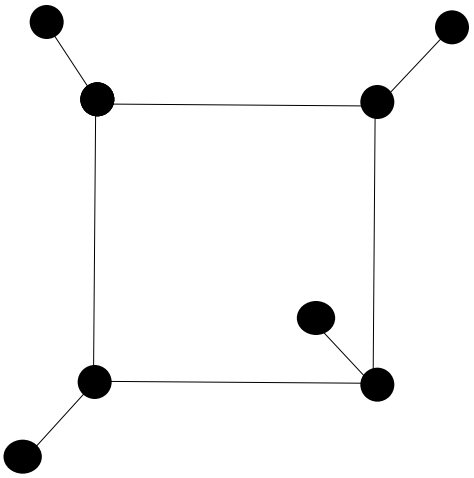
Force directed algorithm for very large graphs

Simple iterative approach takes a lot of iterations to converge on large networks

Also can have many local minima.

Slow convergence + many local minima → standard iterative algorithm unsuitable for large graphs

Multilevel approach an effective way to overcome both issues



Stuck in an energy trap: local minima

◄ | ►

Multilevel approach

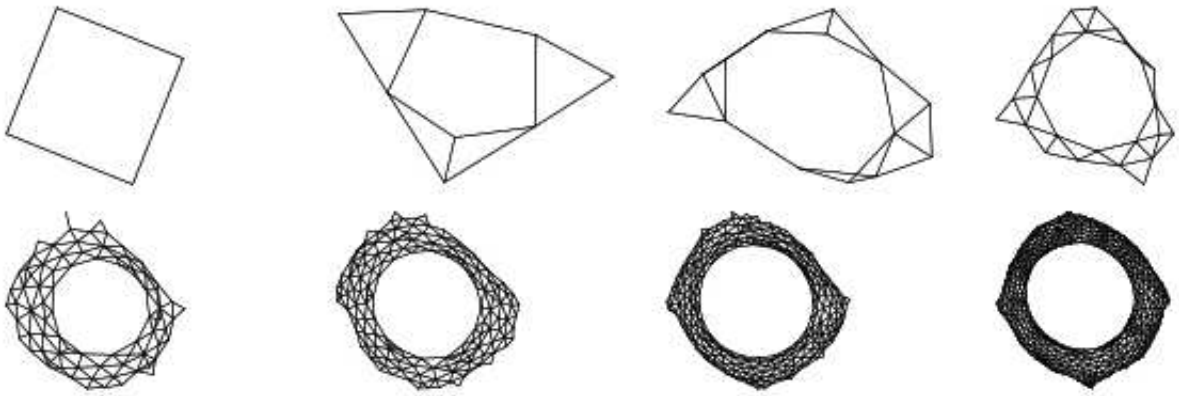
Multilevel/multiscale for graph visualization (Harel & Koren, 1998, Walshaw, 2003, etc)

Replaced a large graph by a series of smaller and smaller "coarser" graphs

Each coarser graph encapsulates the essential info of the original graph

But smaller graphs are much easier to draw (globally) optimally

Layout a coarser graph → interpolate the layout to a finer graph → refine the layout → repeat

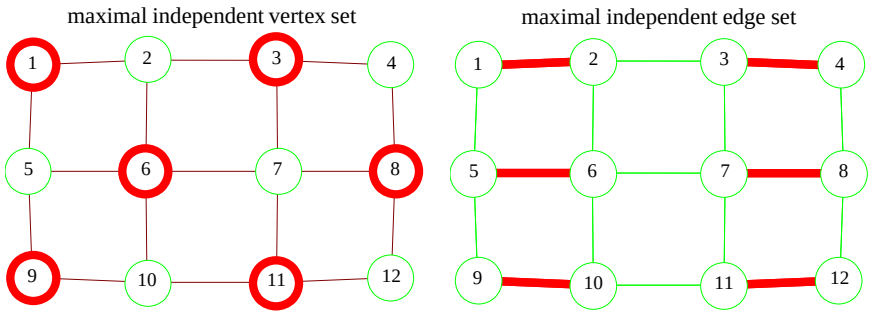


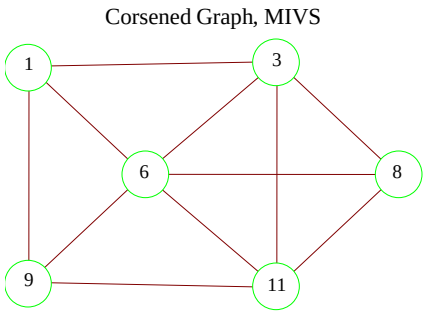
Multilevel key ingredient: graph coarsening

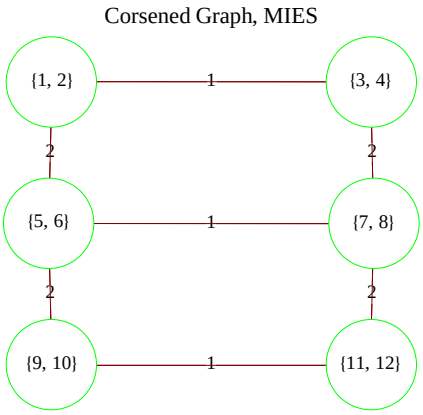
Two most popular coarsening schemes:

Maximal independent vertex set, then link the vertices in the set.

Edge contraction of *maximal independent edge set*.







◄ | ►

◄ | ►

An implementation

A multilevel implementation that uses quadtree data structure to approximate long range force: sfdp

The University of Florida Sparse Matrix Collection

The largest collection of sparse matrices (graphs). Total 1877, of which 1502 are square matrices.

The largest publicly available set of sparse matrices that arise in real applications.

Domains: structural engineering, computational fluid dynamics, model reduction, electromagnetics, semiconductor devices, thermodynamics, materials, acoustics, computer graphics/vision, robotics/kinematics, and other discretizations;

Others that does not have a geometry: optimization, circuit simulation, networks and graphs (including web connectivity matrices), economic and financial modeling, theoretical and quantum chemistry, chemical process simulation, mathematics and statistics, and power networks.

Total 1877 matrices, of which 1502 are square matrices.

Sparse matrices can be considered as adjacency matrix of graphs

Graph visualization is a way to discover and visualize structures in complex relations.

Get a glimpse of the kind of things people are studying by visualizing the sparse matrices

Challenges: network/power-law kind of graphs

With multilevel approach, combined with suitable approximation scheme via quadtree/kd-tree, force directed algorithm can be very efficient and effective for a lot of graphs

Network/power-law kind of graphs pose some special challenges!

Challenges for network/power-law like graphs

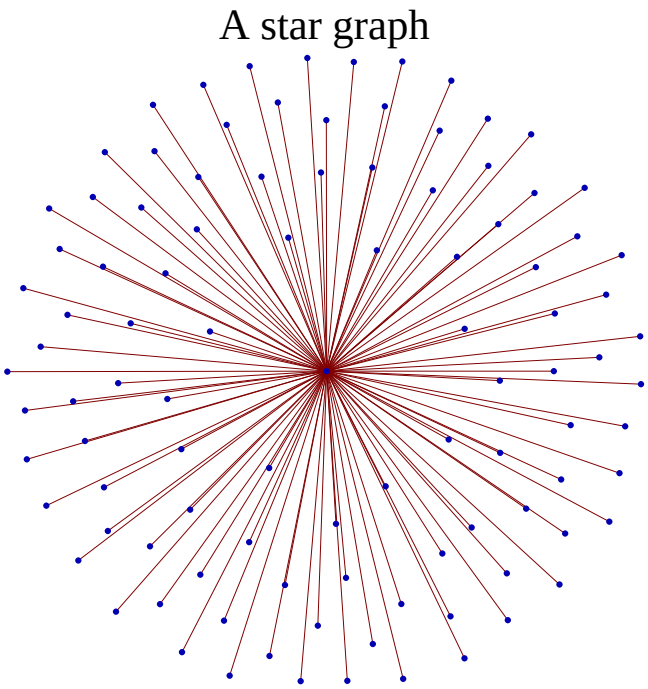
Typical example (gupta1): coarsening gets ever slower

```
level -- 0, n = 31802, nz = 2132408 nz/n = 67.052638
 level -- 1, n = 20861, nz = 2076634 nz/n = 99.546235
  level -- 2, n = 12034, nz = 1983352 nz/n = 164.812365
     nc = 11088, nf = 12034, coarsening too slow, stops
```

From level 2 (12034 vertices), the next level 3 will have 11088 vertices, the difference is so small that multilevel process has to be aborted.

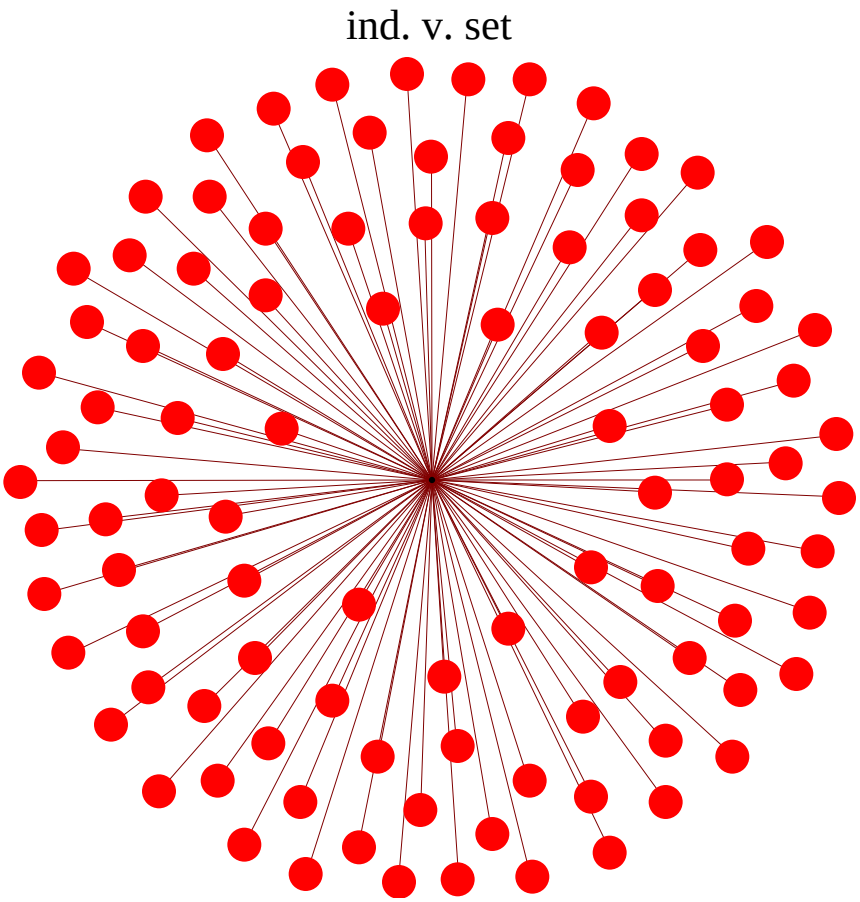
Challenges for network/power-law like graphs

Such networks often contain structures not amiable to graph coarsening.



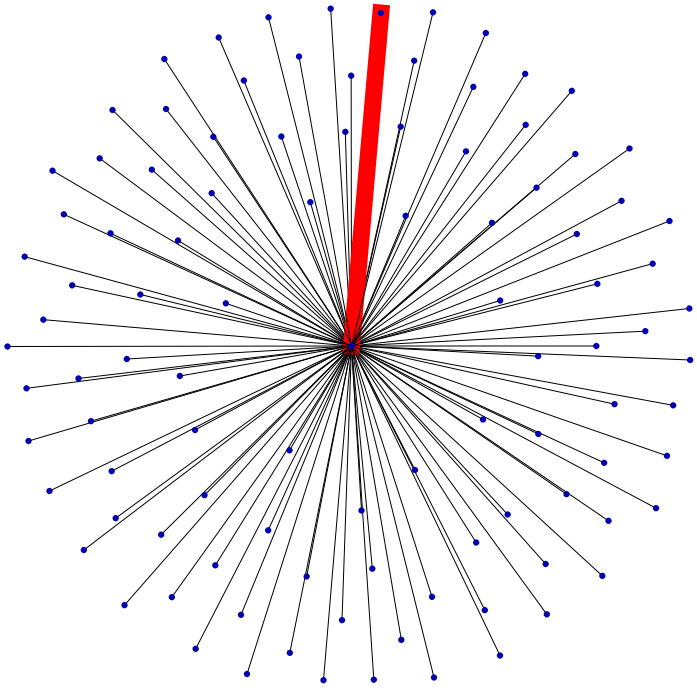
Challenges for network like graphs

Maximal independent vertex set: there is either one independent vertex, or $n - 1$.

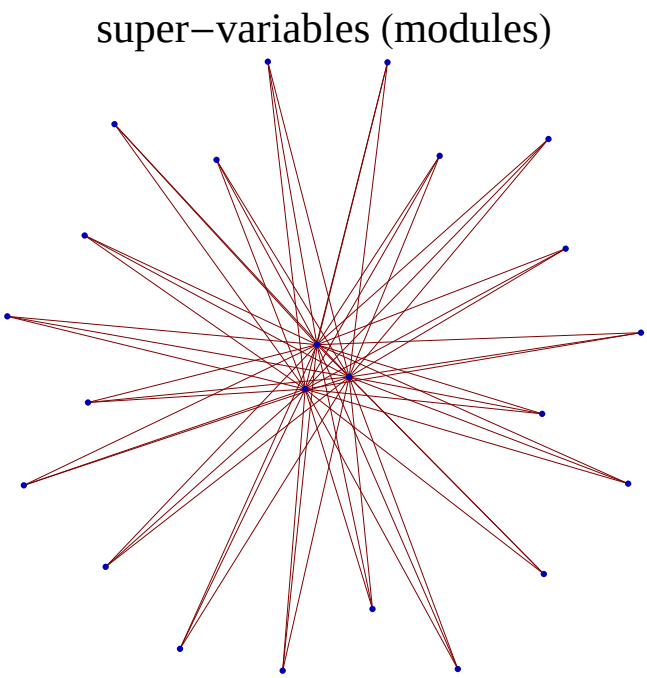


Maximal independent edge set: there is only one independent edge.

max ind. edge set



Challenges for network like graphs



◄ | ►

The multilevel coarsening scheme are improved to handle such issues.

Allow us to layout all 1502 square matrices, results:

http://www.research.att.com/~yifanhu/GALLERY/GRAPHS/index.html

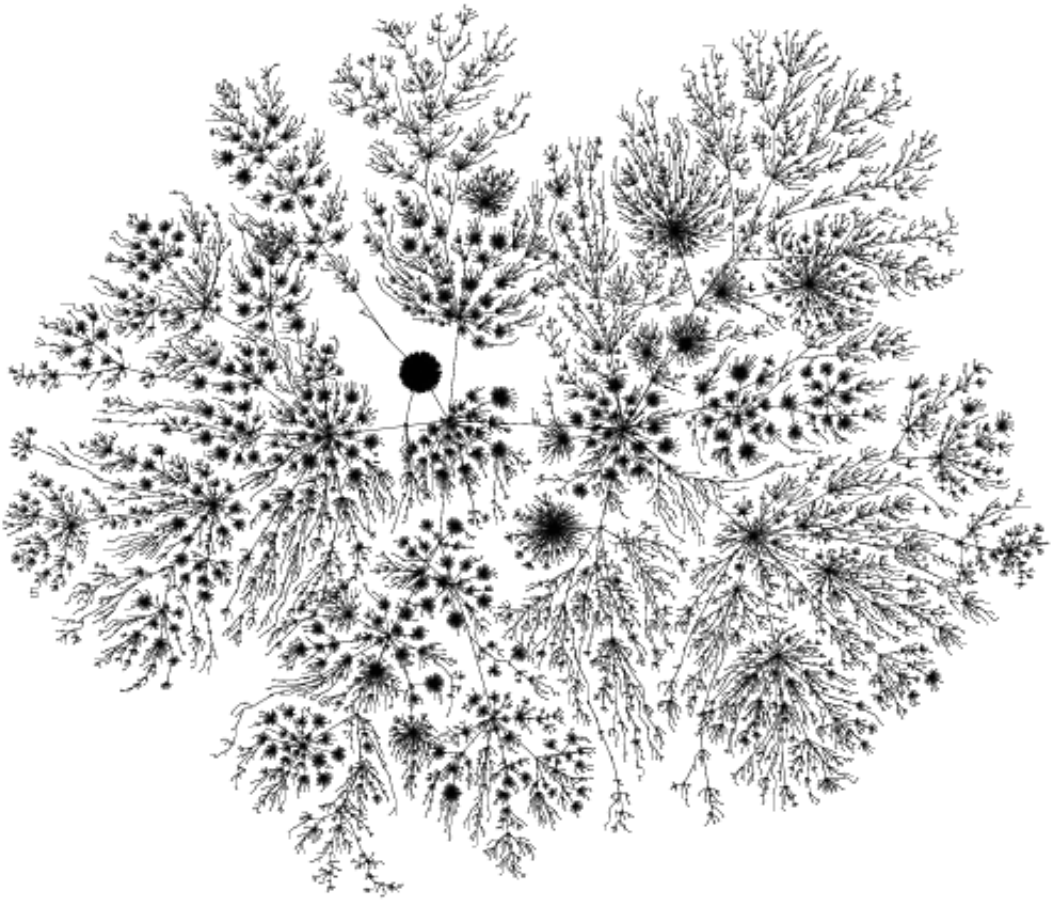
Other Challenges

But the challenges remained:

An open question: how to suitably visualize network/power law like graphs?

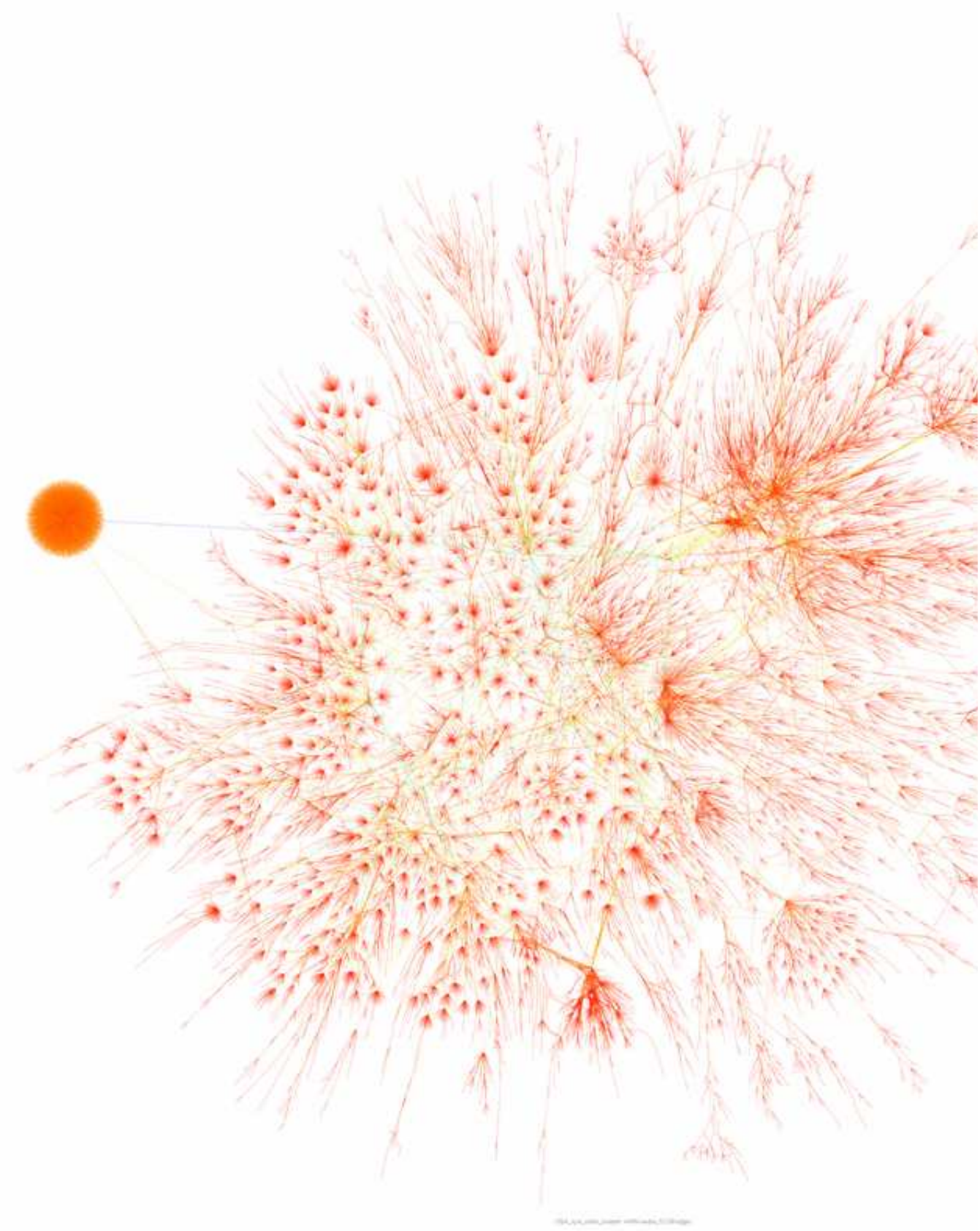
We can plot a spanning tree/planar subgraph, but it is that sufficient?

Spanning tree of USA graph. ($|V| = 44\,954$).



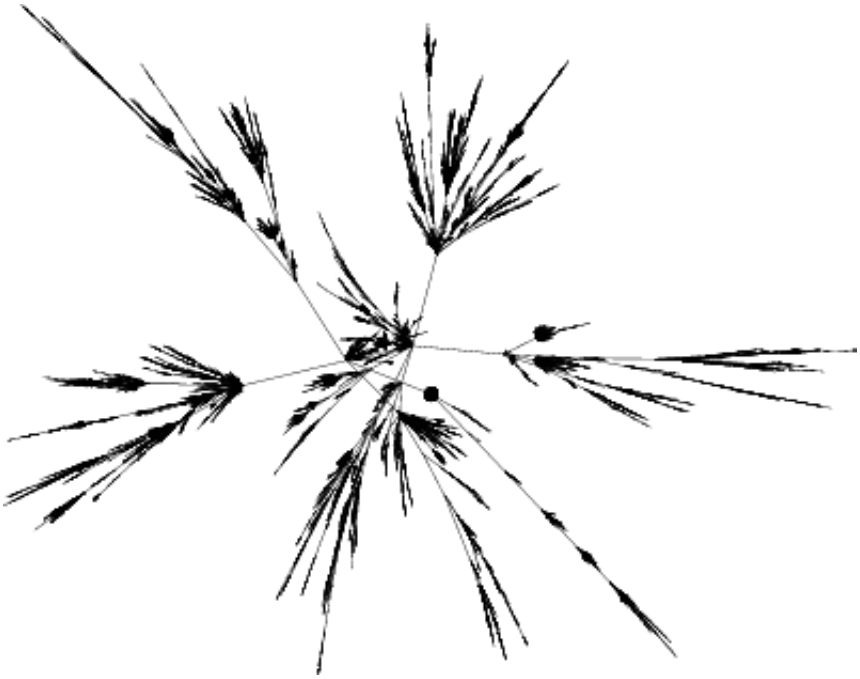
A power-law like graph

The original graph: not too bad



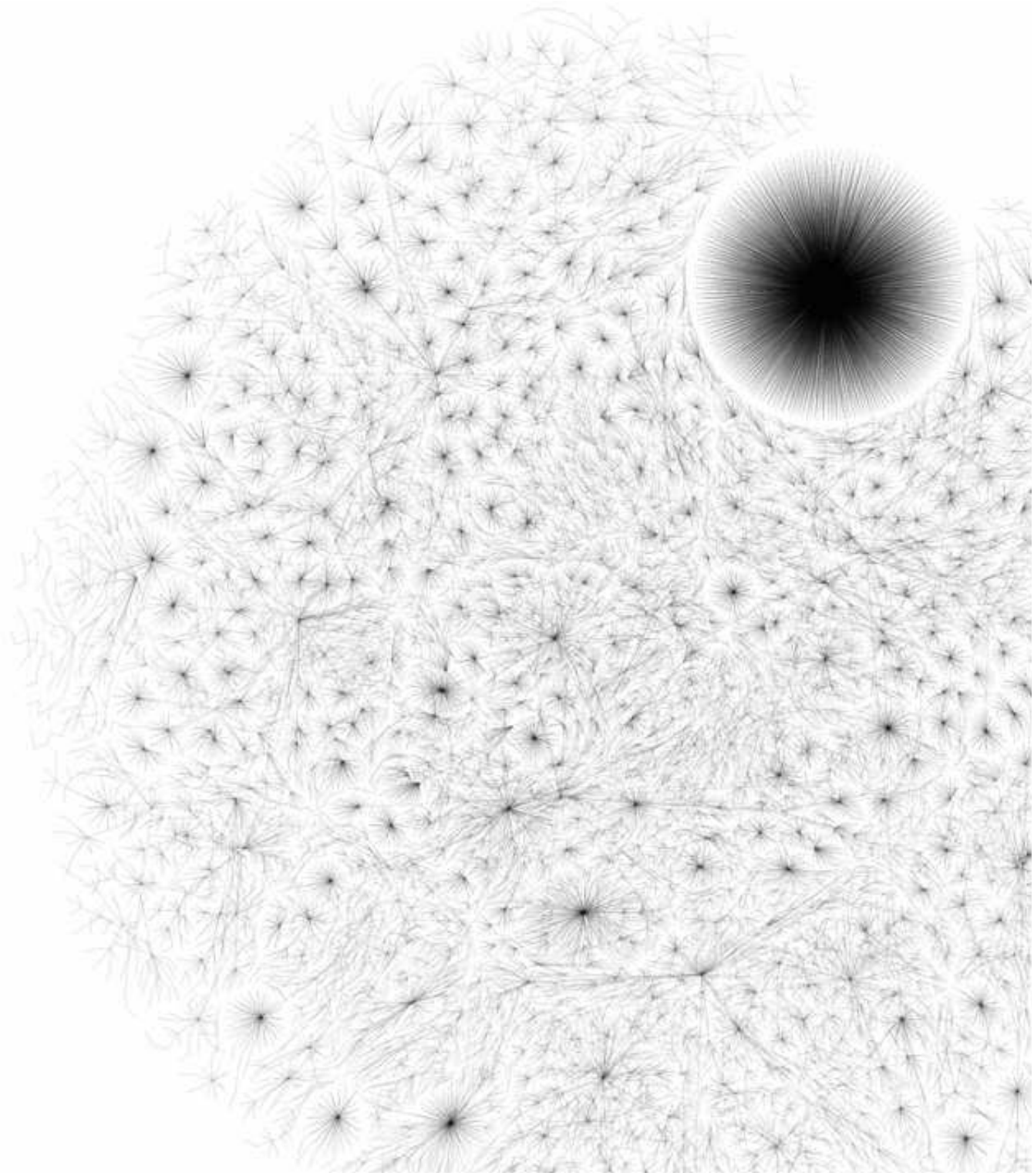
Using other code on the tree

FM^3 (Hachul & Junger 2005)



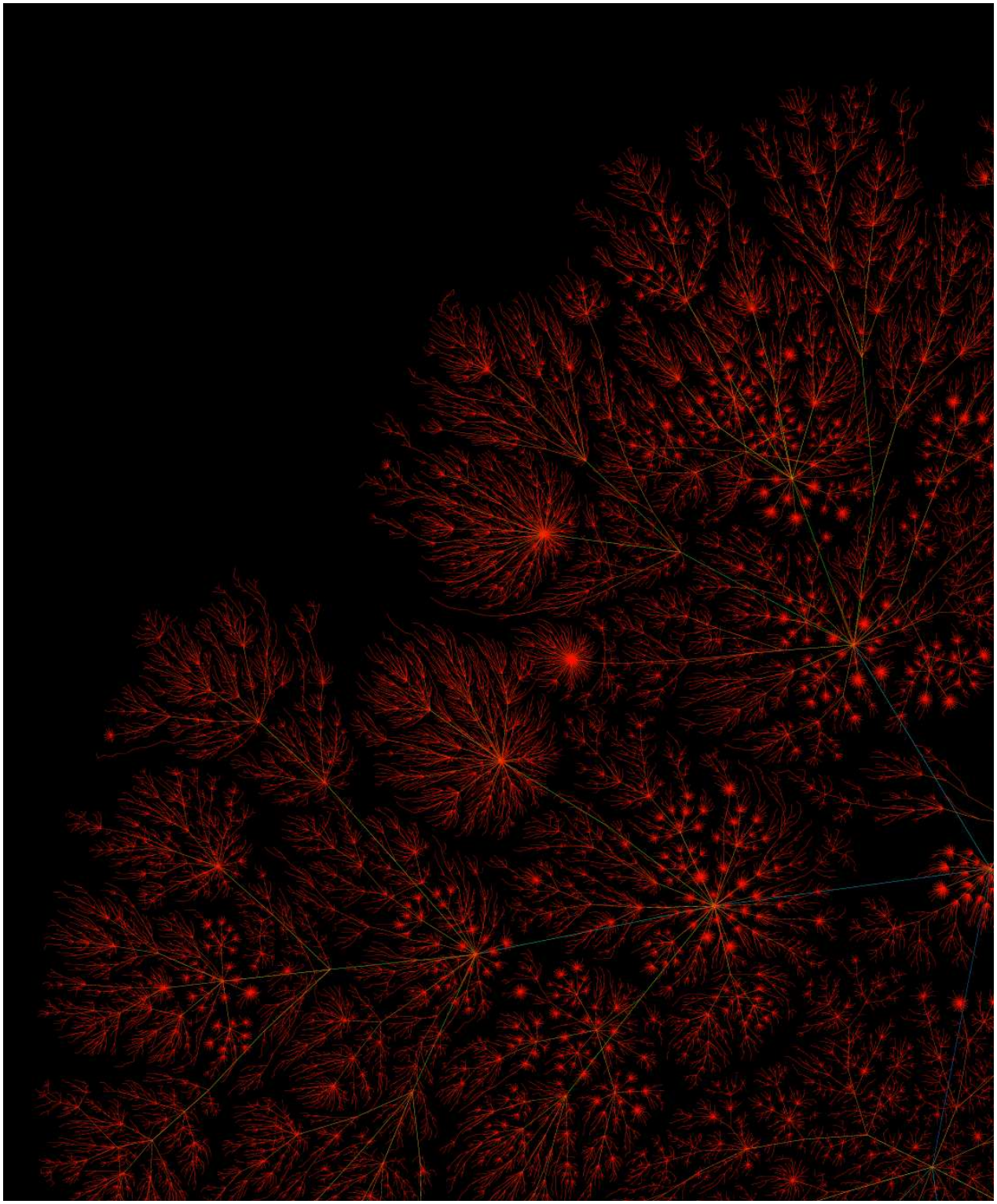
Using other code on the tree

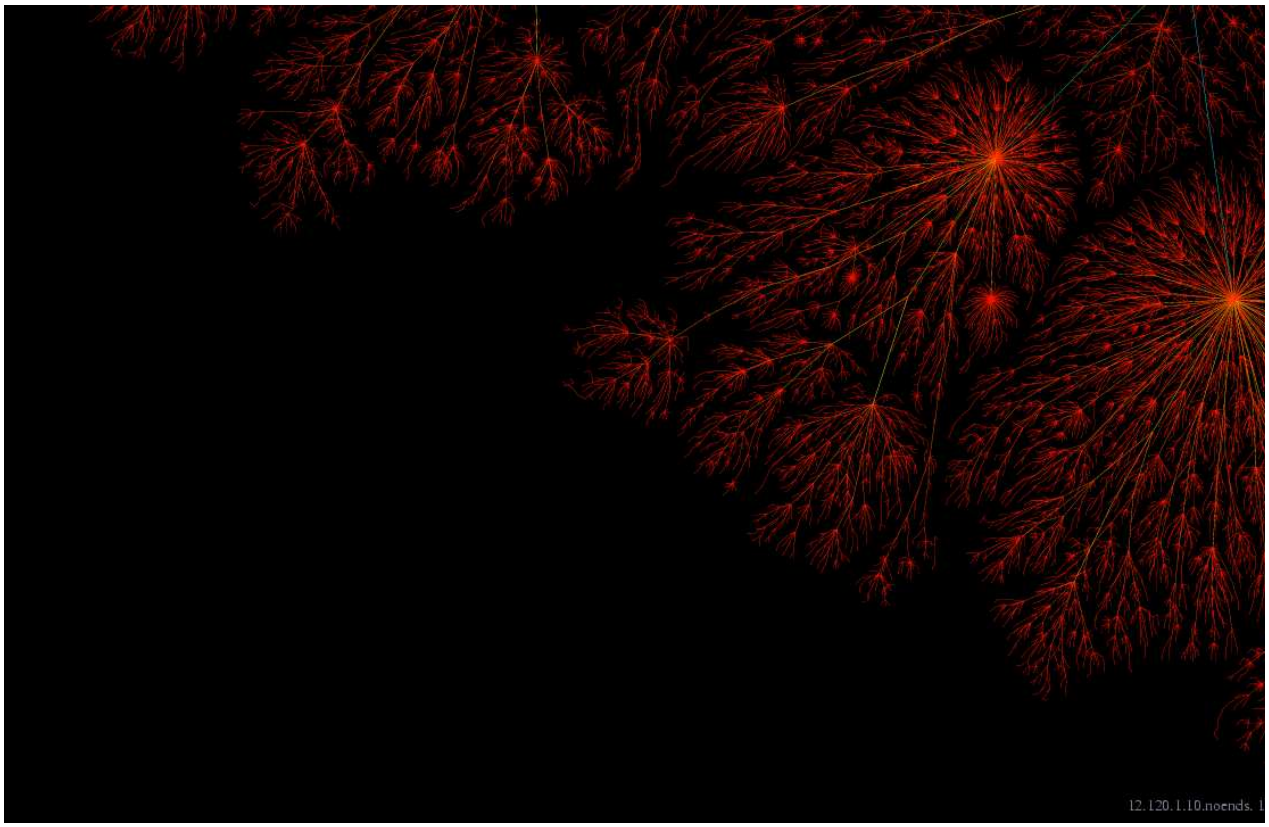
LGL



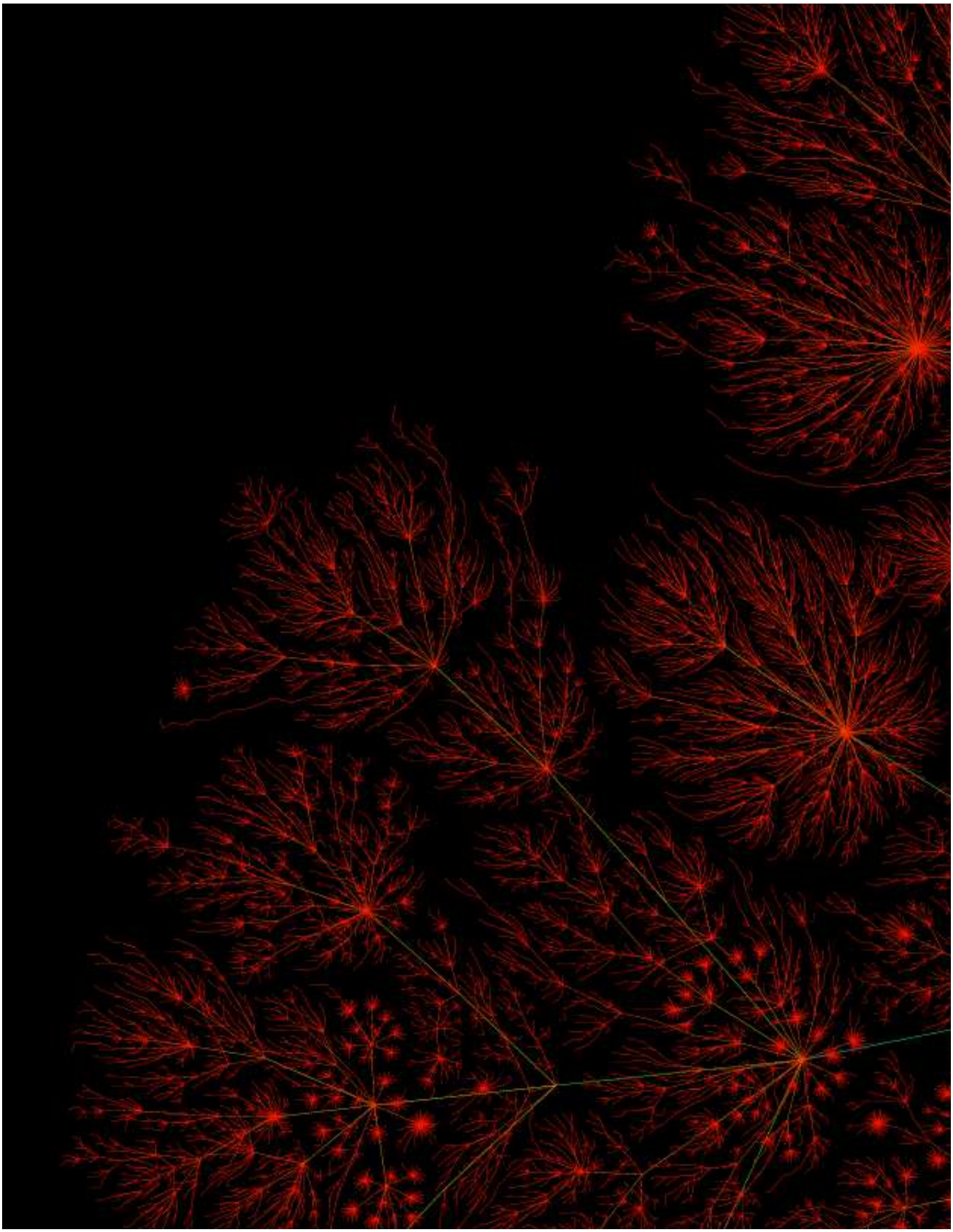
But if the graph is larger/more edges

Graph "12.120.1.10.noends". $|V|$ =136205. 511 seconds for layout, 58 seconds post processing.

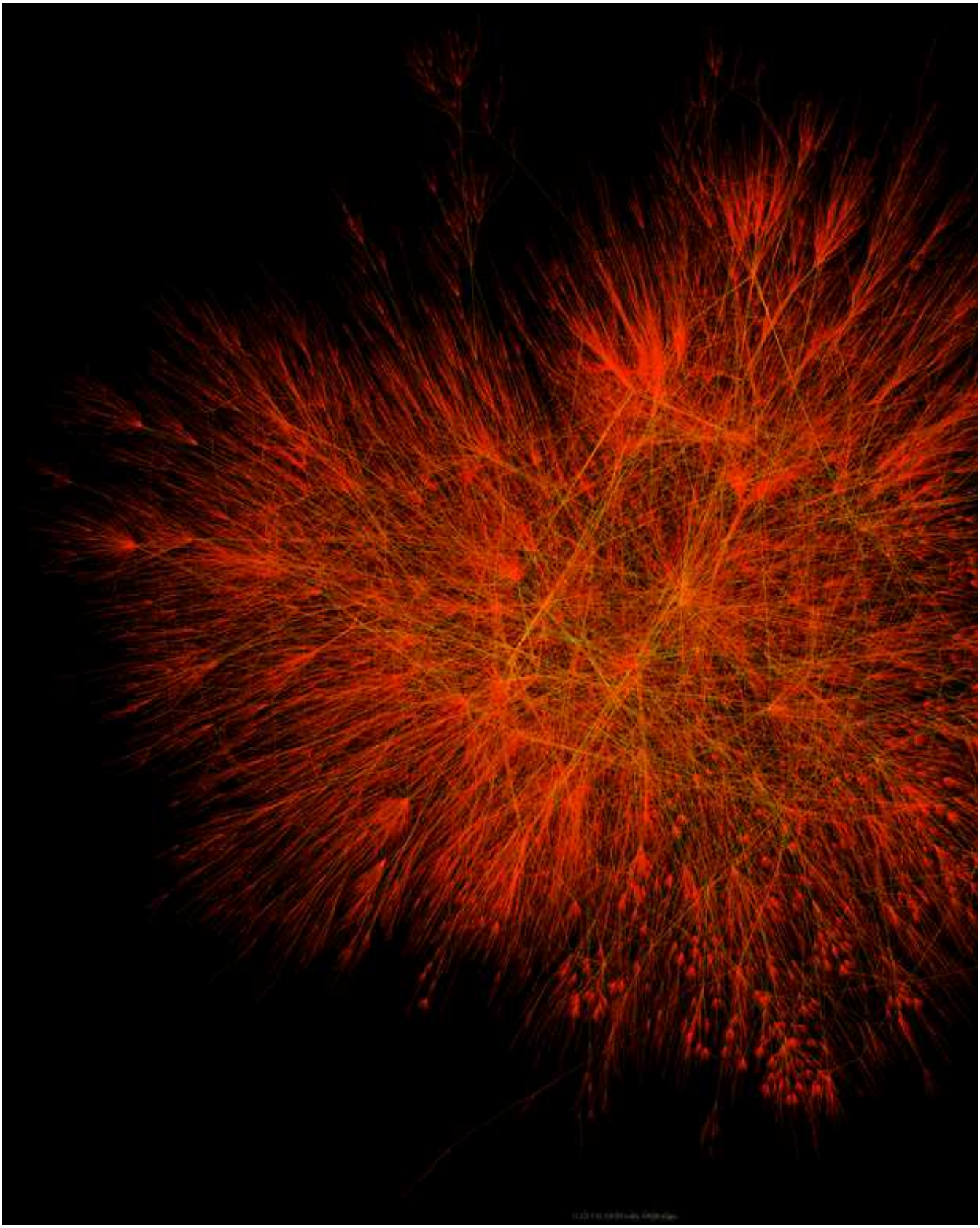




Details

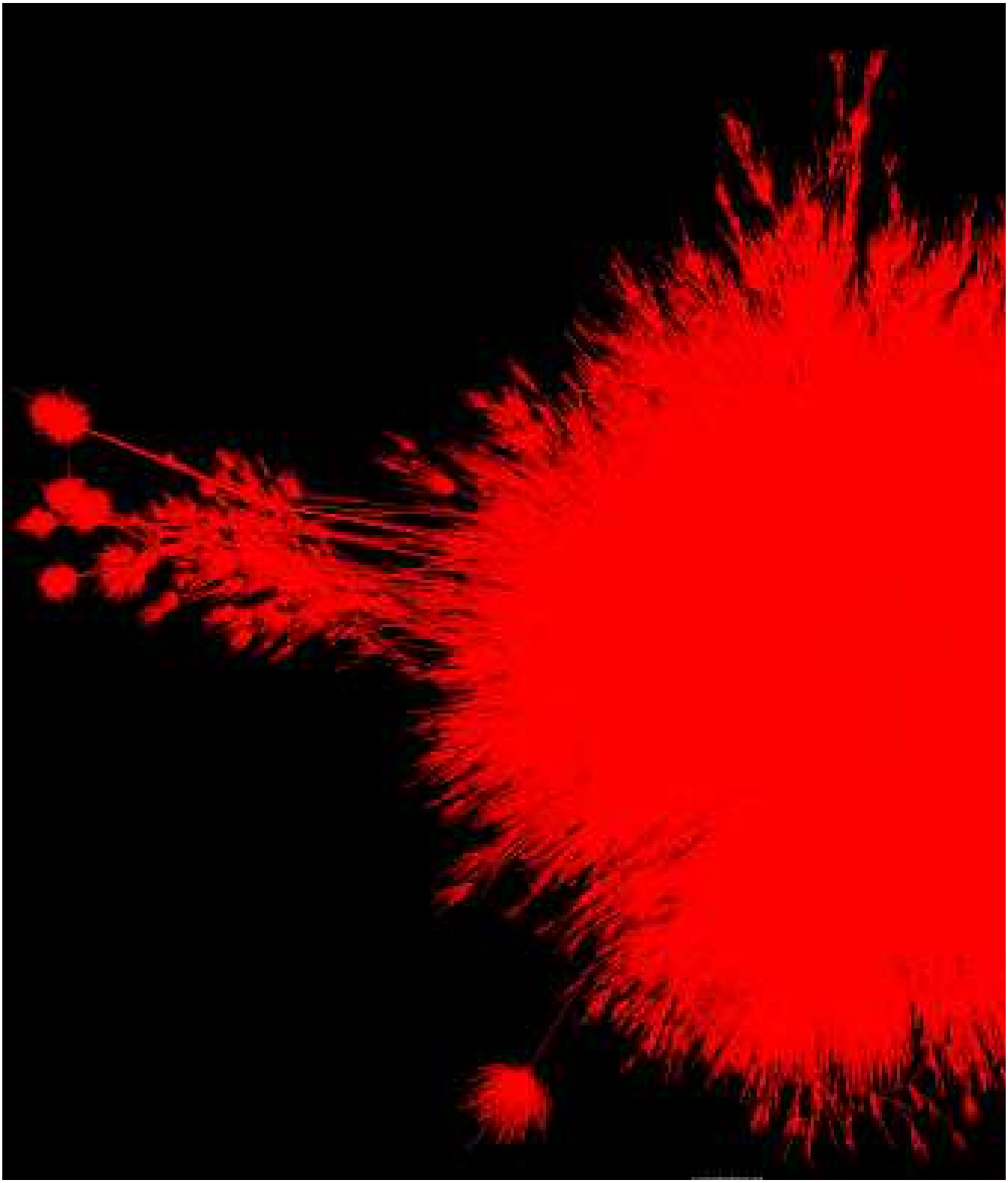


The original graph: getting close to a hair ball



Here is a real hair-ball!

Graph "20070723". $|V| = 281\,184$. An internet graph taken on July 23, 2007.



An even bigger hair-ball!

The largest graph in the UFL collection with 8863287 vertices and 44185251 edges, took 35 hours of cpu time.

http://www.research.att.com/~yifanhu/GALLERY/GRAPHS/GIF_SMALL/Gleich@wb-edu.html

Conclusions

We can handle pretty large graphs, particularly for "regular" graphs.

An open question: how best to draw hair-balls (network/power-law graphs)?

possible solutions
-- spanning tree
-- max planar subgraph
-- domain specific knowledge (e.g., shrink star graphs, other structures)
-- multilevel/topological fisheye view?
-- ??