Modeling Power Grids

-- not so easy

Per Arne Rikvold[1]

with

Ibrahim Abou Hamad,[1*]

Brett Israels,[1#] and Svetlana V. Poroseva[2]

[1] Department of Physics, Florida State University

[2] Department of Mechanical Engineering, University of New Mexico

Supported by NSF DMR

* Current address: BP R&D, Naperville, IL

Current address: Dept. of Chemistry, Univ. of Oregon.

Why study power grids?

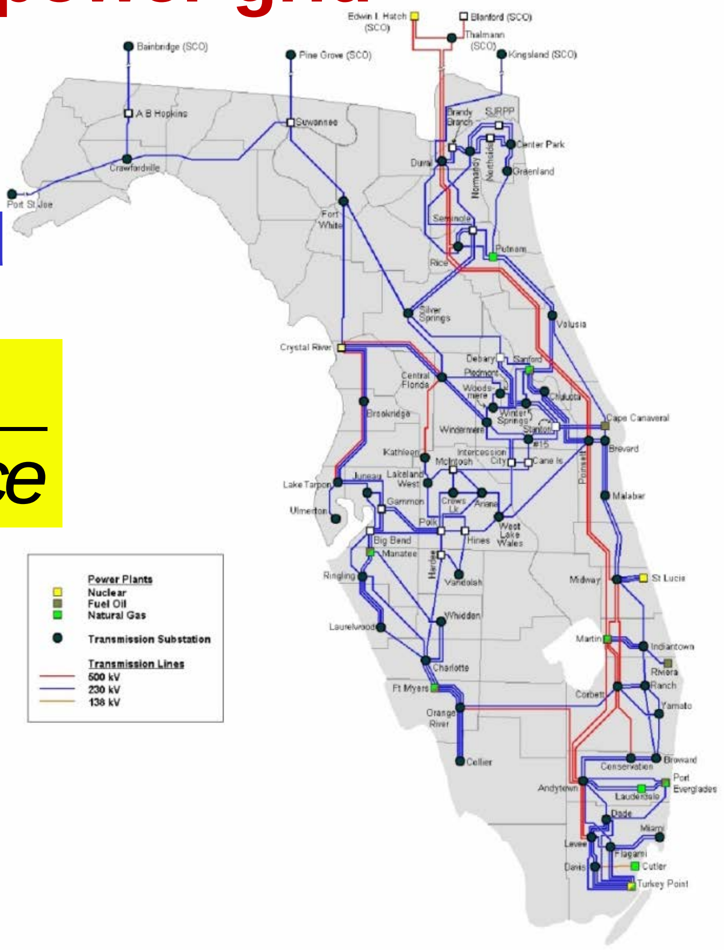
Motivation:

- Society relies heavily on power grid performance
- Modern & future power grids are large, complex, integrated
- Vulnerability to natural disasters, hostility, software failure
- Network theory is in rapid development

Goal

- Stop cascading failures by
  - partitioning a power grid into parts that are
    - ***weakly*** connected
    - nearly ***self-sufficient*** in power

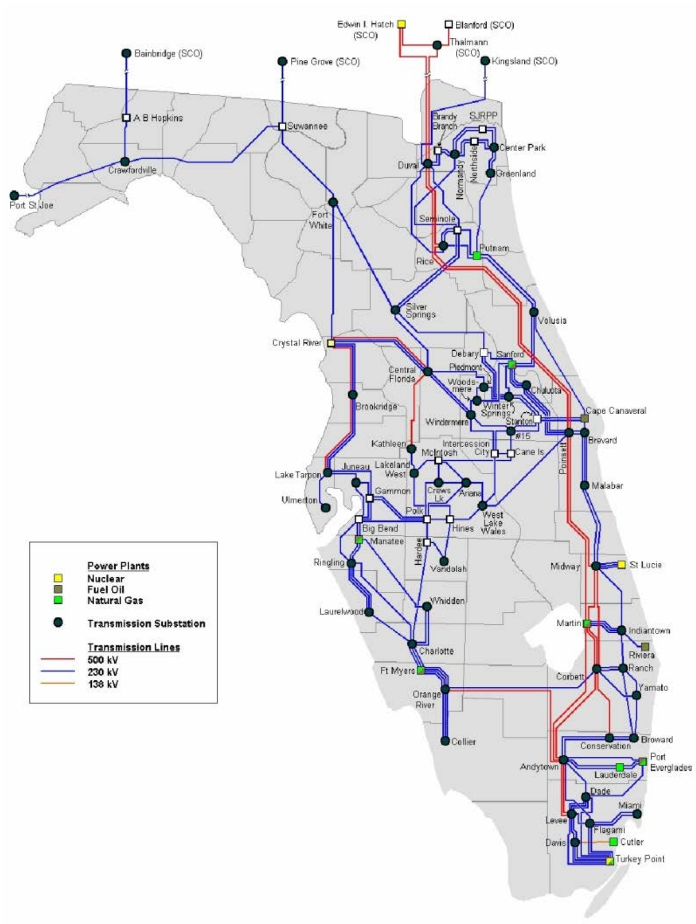
Example: Floridian high-voltage power grid



Weight Matrix for the Florida Grid

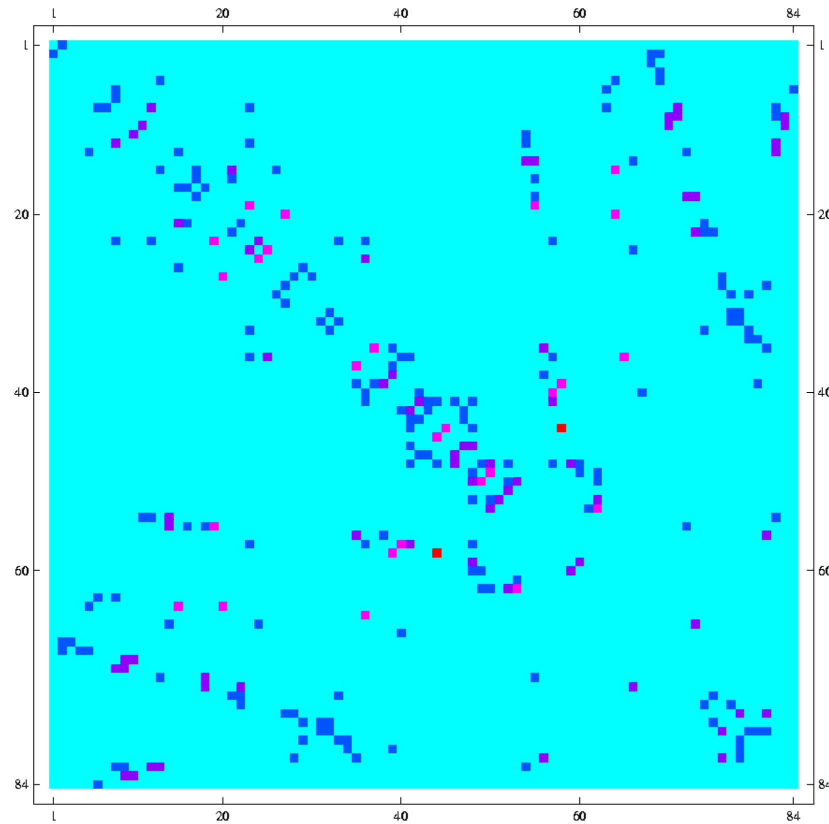
$$w_{ij} = \frac{\#\, of\ lines\ between\ vertices\ i\ \text{and}\ j}{normalized\ geographical\ distance}$$

Florida High-voltage Transmission Grid



Florida electric power grid map

Network of 84 vertices including 31 generators



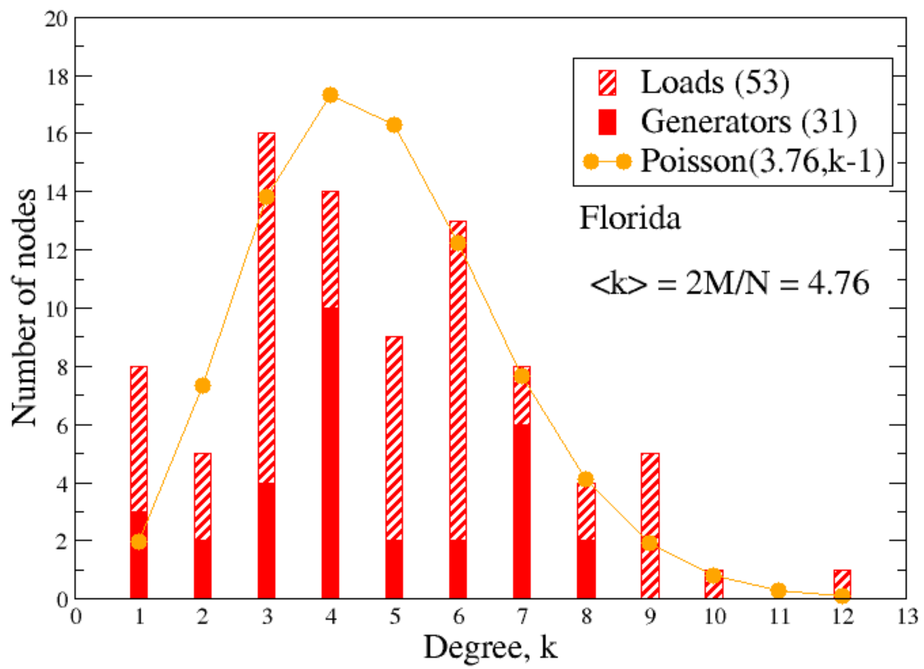
Weight matrix W

The weight of a connection is proportional to the number of parallel connecting lines between two vertices. Connections are represented by dots, with strengths from **high=red** to **low=blue**.

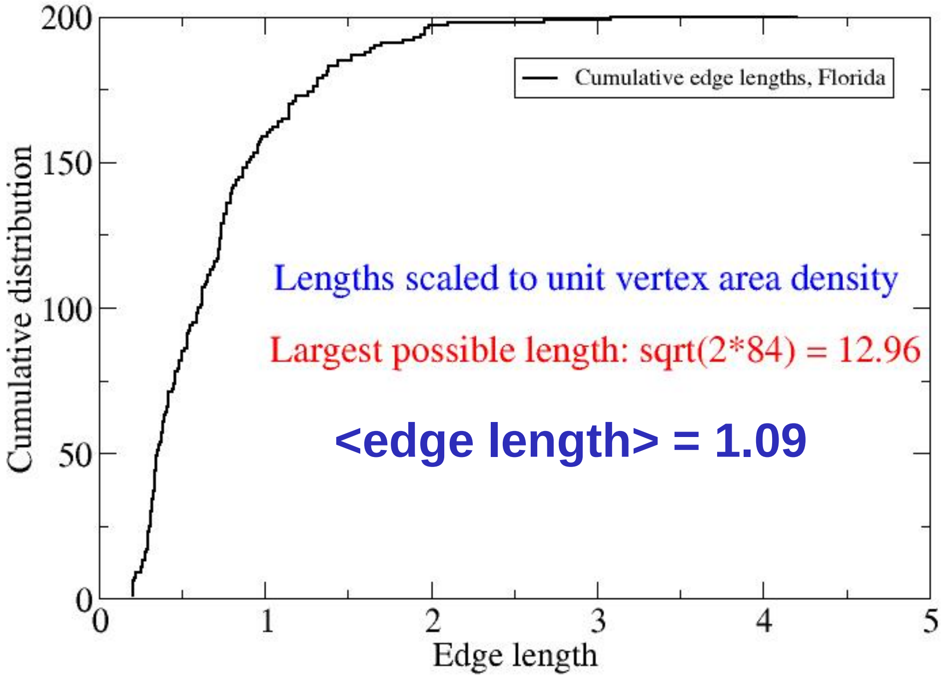
Building a model power grid that we can play with

- *Geographically embedded* network
  - Scale such that area density of vertices is unity (N vertices in square of side $N^{1/2}$)
- Proportion of power plants (Florida: 31/84)

Degree distribution

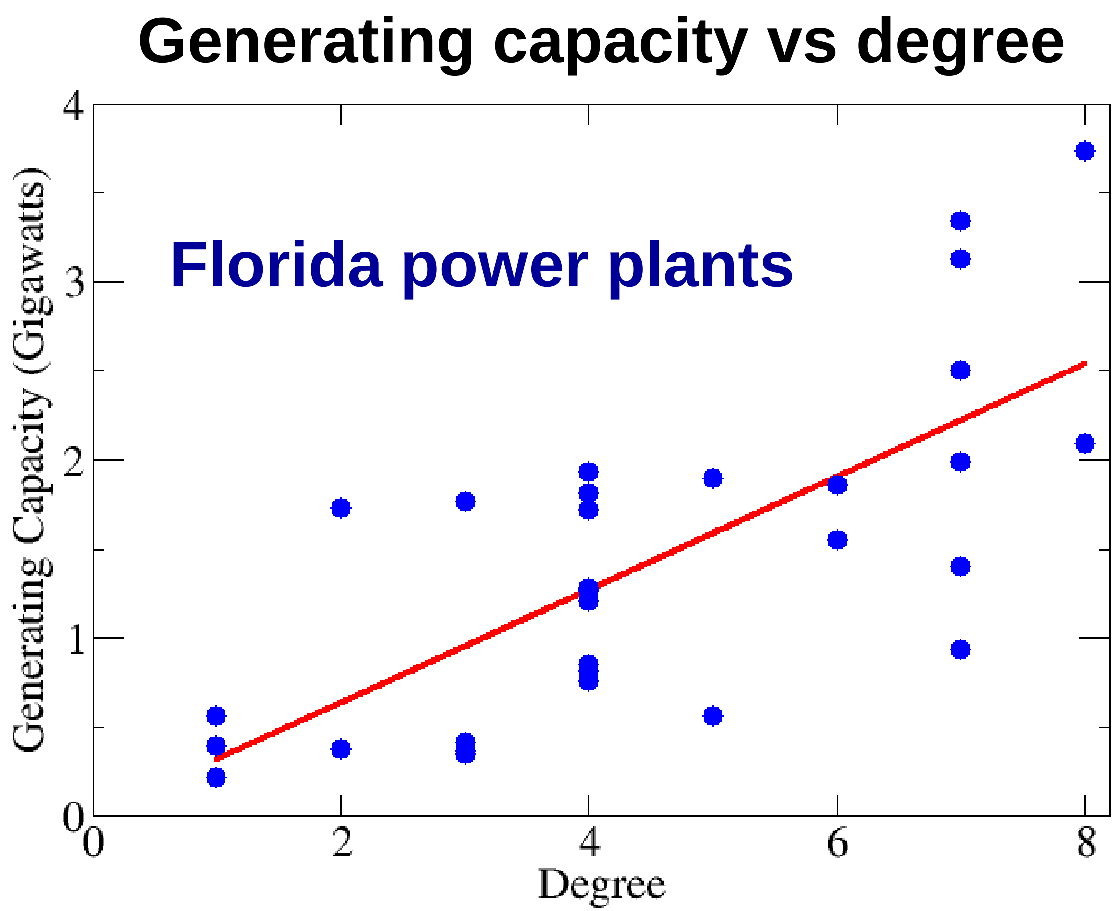


Edge-length distribution



Building a model power grid, 2

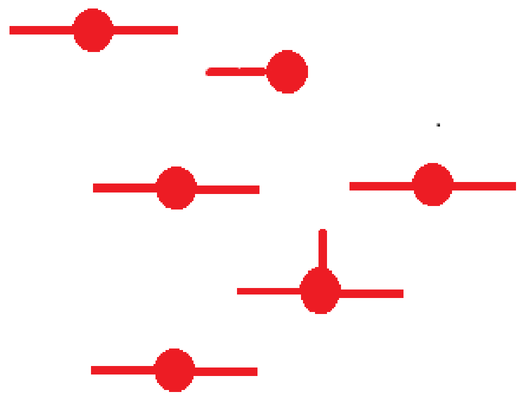
- Generating capacities
- Power demand of loads
  - Use degree as proxy



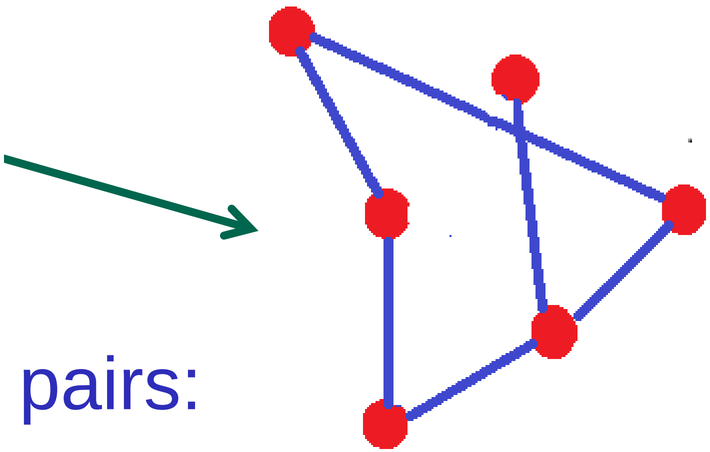
Building "Random Florida"

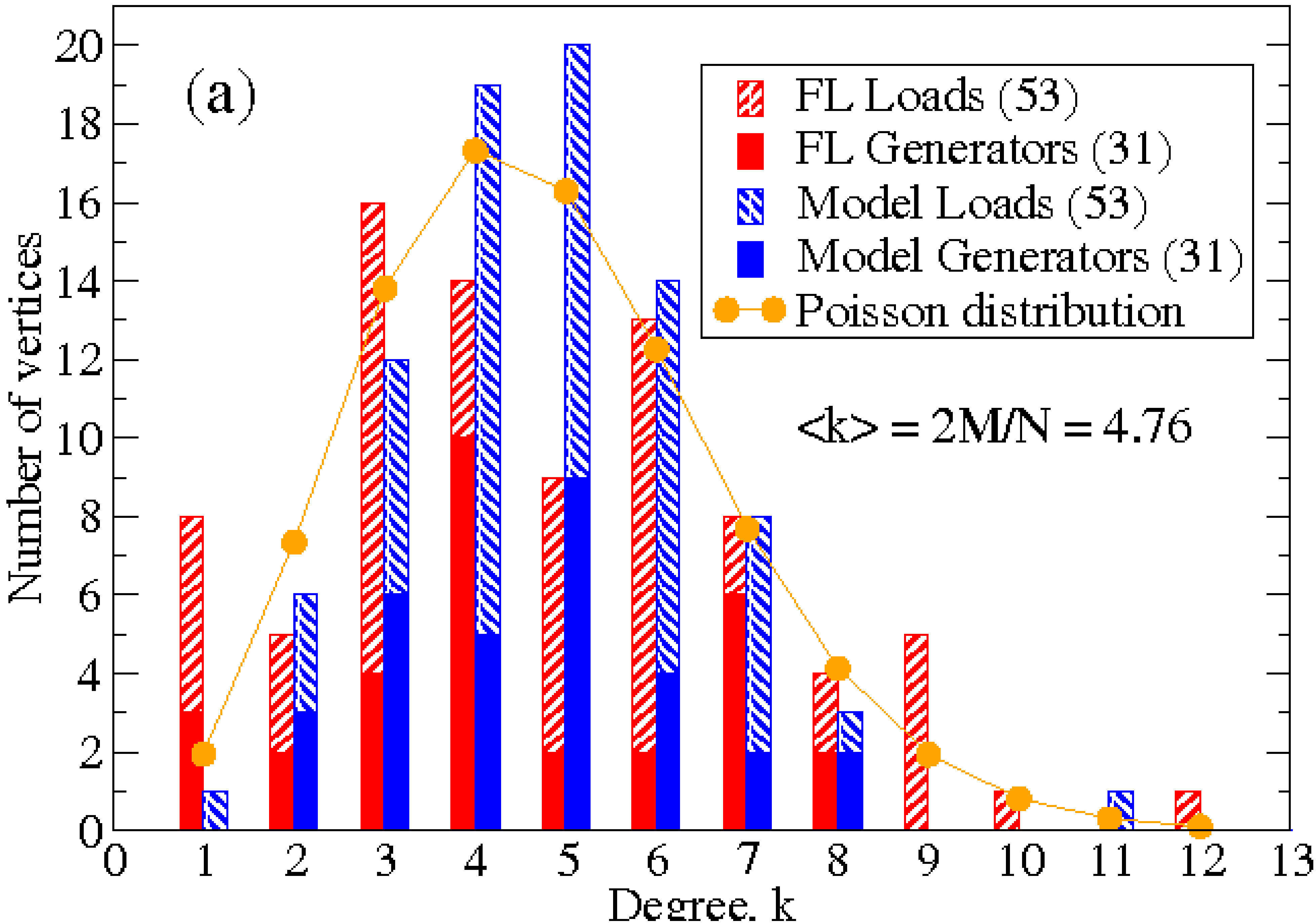
1. Place N=84 vertices randomly in square of side $N^{1/2}$. (One point per unit area.)
2. Choose 31 last vertices as generators.
3. Create degree distribution with $\langle k \rangle = 2M/N = 4.76$ using *stub method*:

- Connect $\langle k \rangle N$ stubs (half-edges) randomly to the N vertices:



- Connect the stubs randomly in pairs:

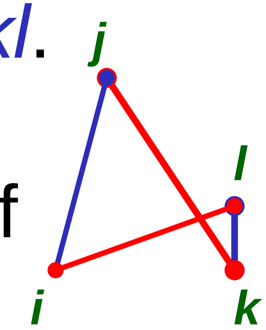




(a)
FL Loads (53)
FL Generators (31)
Model Loads (53)
Model Generators (31)
Poisson distribution
<k> = 2M/N = 4.76
Number of vertices
Degree, k
0
2
4
6
8
10
12
14
16
18
20
0
1
2
3
4
5
6
7
8
9
10
11
12
13

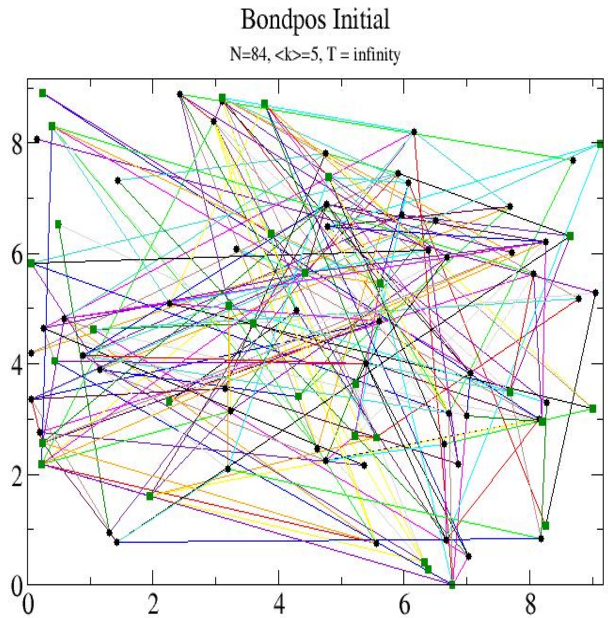
4. Assign "edge energy" *E*(*ij*) equal to length of edge *ij*, and "cool" the system of edges to favor shorter ones:

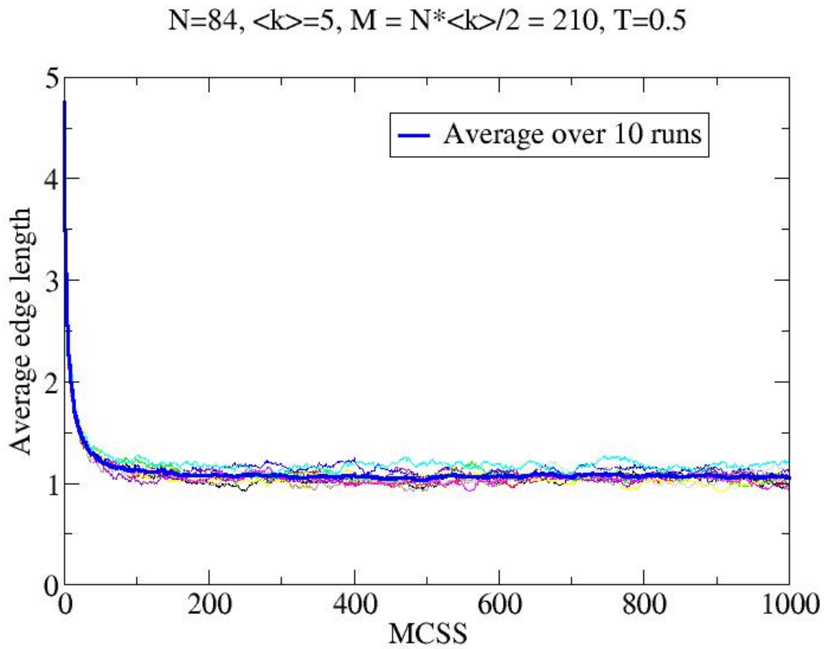
- Randomly choose two different edges, *ij* and *kl*.
- Calculate *E*(*ij*,*kl*) = *E*(*ij*) + *E*(*kl*).
- Interchange *j* and *l* and calculate the energy of the new edge pair, *E*(*il*,*kj*).
- Accept new edge pair with *Metropolis probability*:

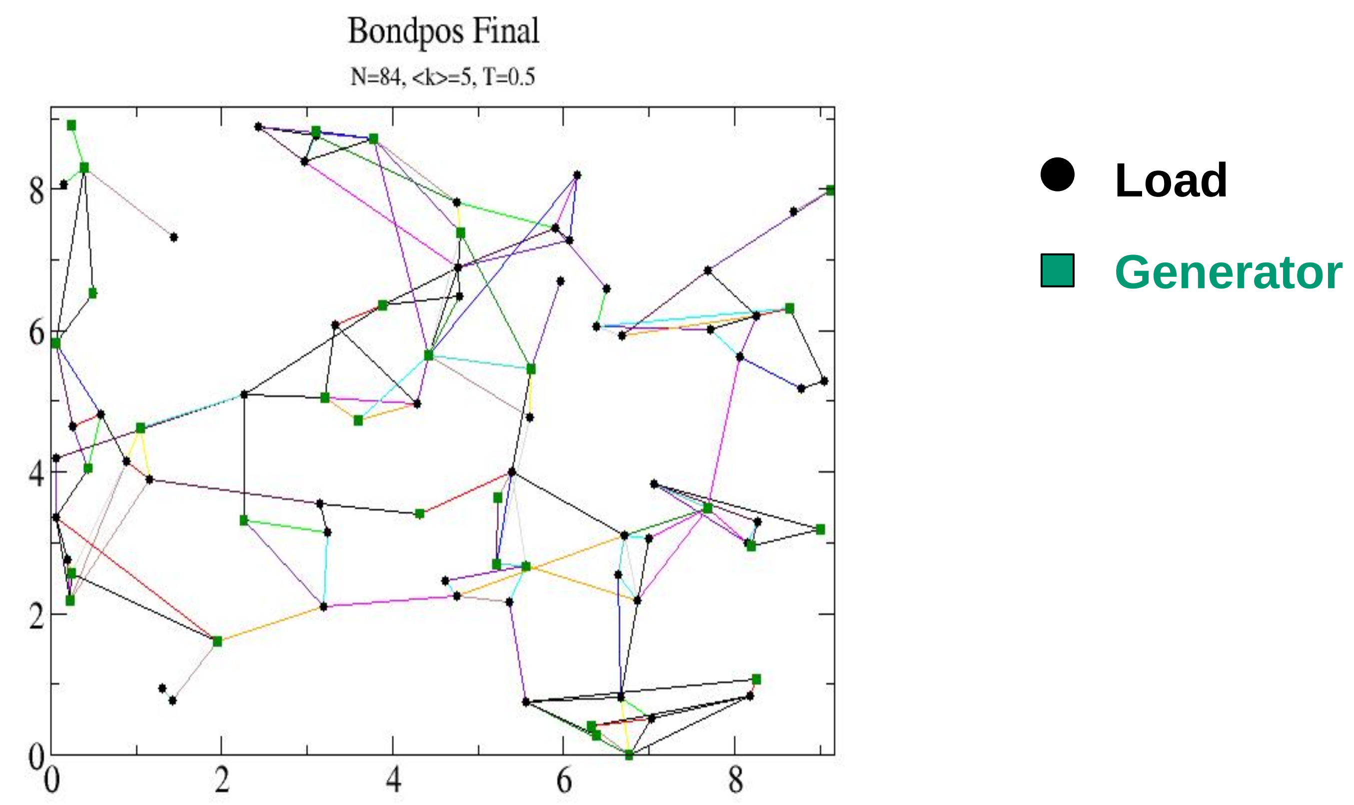


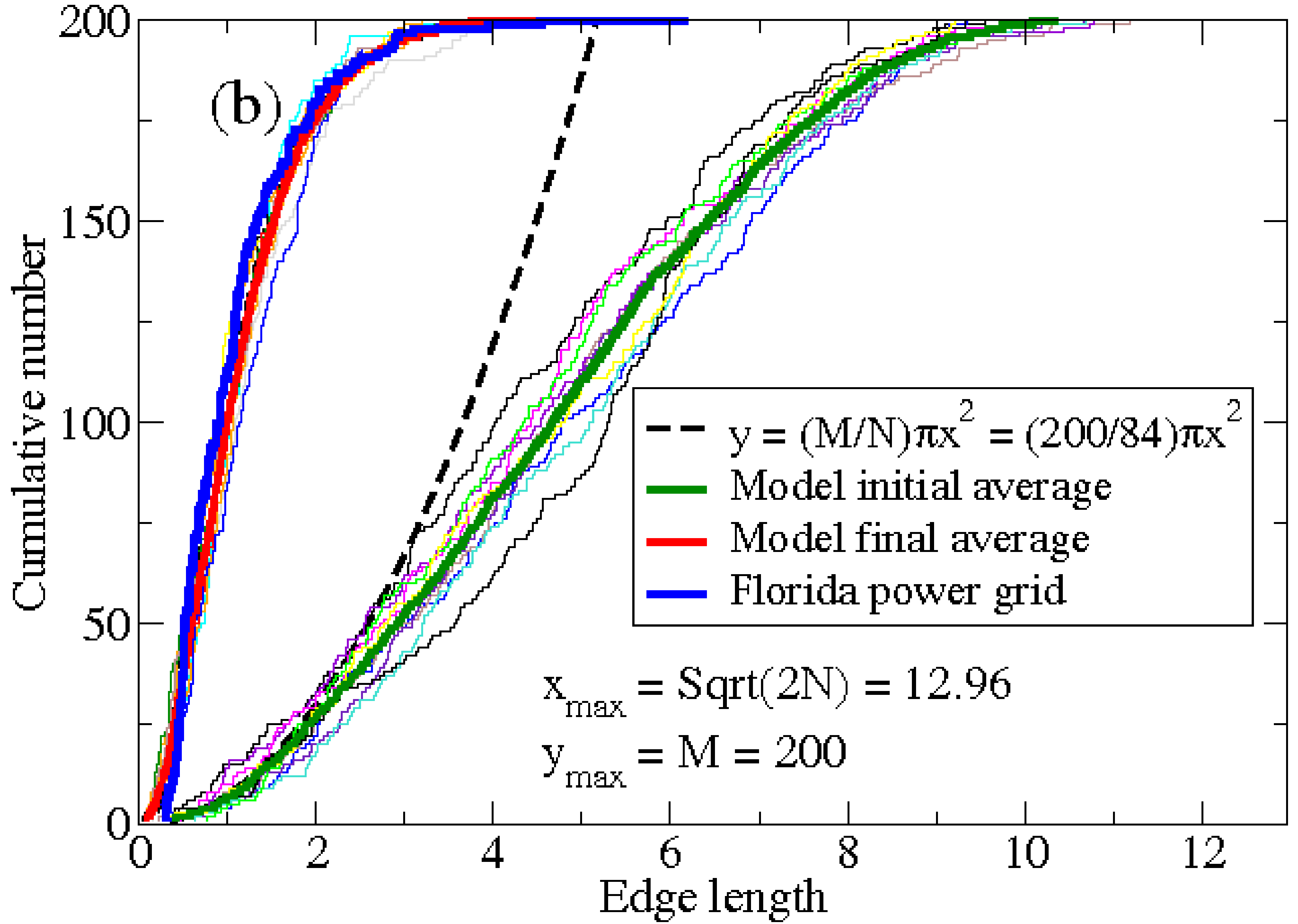
$$P(ij,kl \to il,kj) = \begin{cases} 1 & \Delta E \leq 0 \\ \exp(-\Delta E/T) & \Delta E > 0 \end{cases}$$

- Repeat until Energy becomes stationary.
- Select *representative "equilibrium" configuration* as your "Random Florida"


Bondpos Initial
N=84, <k>=5, T = infinity
0
2
4
6
8


N=84, <k>=5, M = N*<k>/2 = 210, T=0.5
Average over 10 runs
Average edge length
MCSS
0
1
2
3
4
5
200
400
600
800
1000


Bondpos Final
N=84, <k>=5, T=0.5
0
2
4
6
8
Load
Generator



(b)
Cumulative number
Edge length
y = (M/N)πx^2 = (200/84)πx^2
Model initial average
Model final average
Florida power grid
x_max = Sqrt(2N) = 12.96
y_max = M = 200
0
50
100
150
200
0
2
4
6
8
10
12

Partitioning of “Random FL”

Quality measure *E*

Modularity:

$$Q = \frac{1}{w}\sum_{ij}\left(w_{ij} - \frac{w_i w_j}{w}\right)\delta\left(C(i), C(j)\right)$$

- ***w*** is the total weight
- $\boldsymbol{w_i}$ is the weighted number of edges connecting to node ***i*** or the *weight* of node ***i***.
- $\boldsymbol{\delta(C(i),C(j))}$ equals 1 if ***i*** and ***j*** are in the same cluster, 0 otherwise.
- One wants to *maximize Q* while *minimizing* In/Out currents. Thus the Quality measure ***E*** to maximize is:

$$E = \frac{Q}{Q_{\text{init}}} - \sqrt{\frac{VAR(|\tilde{I}\rangle)}{VAR(|\tilde{I}\rangle_{\text{init}})}}$$

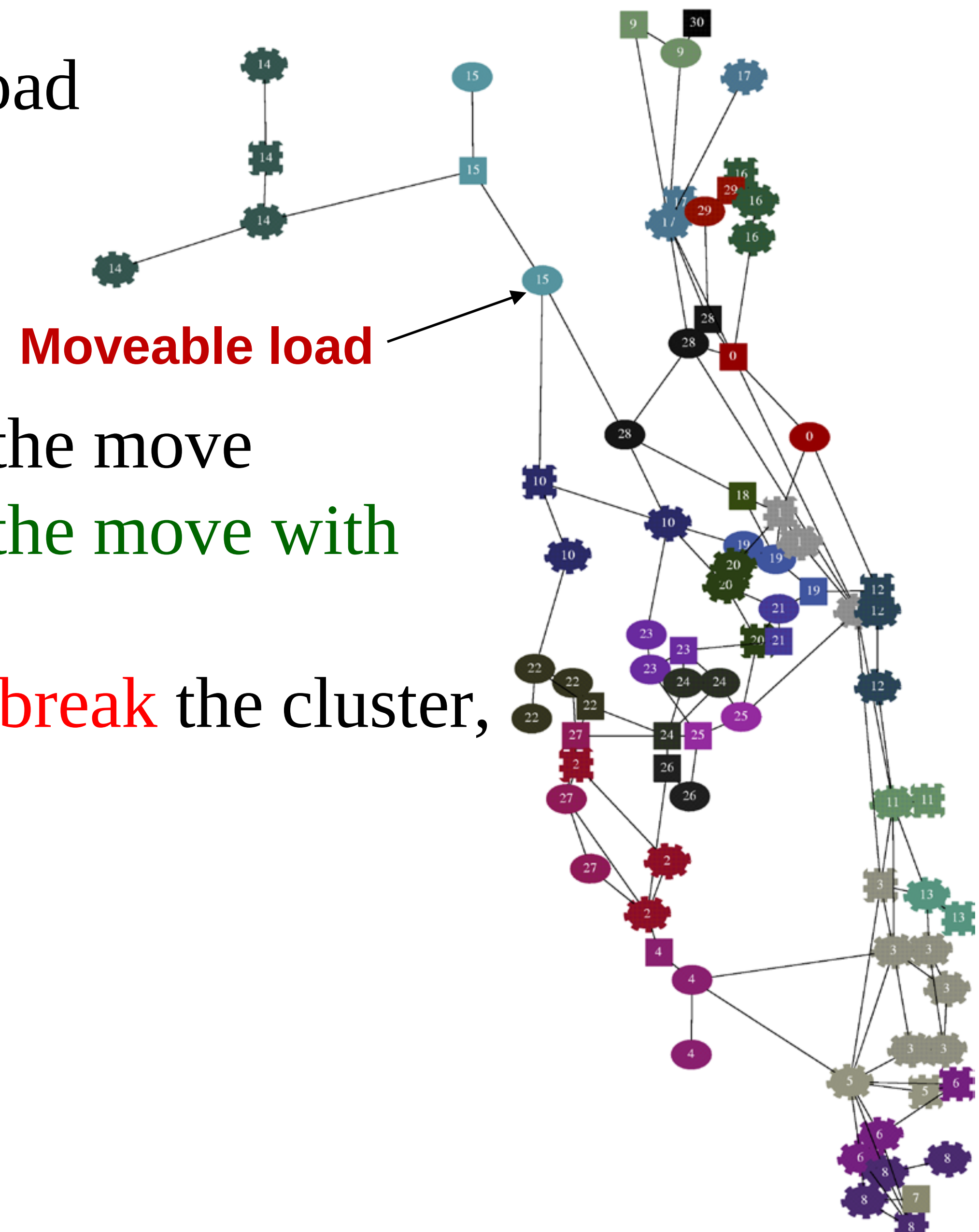
Partitioning Algorithm

- Bottom-up procedure resembling Real-Space Renormaliztion Group (RSRG) in stat-mech.

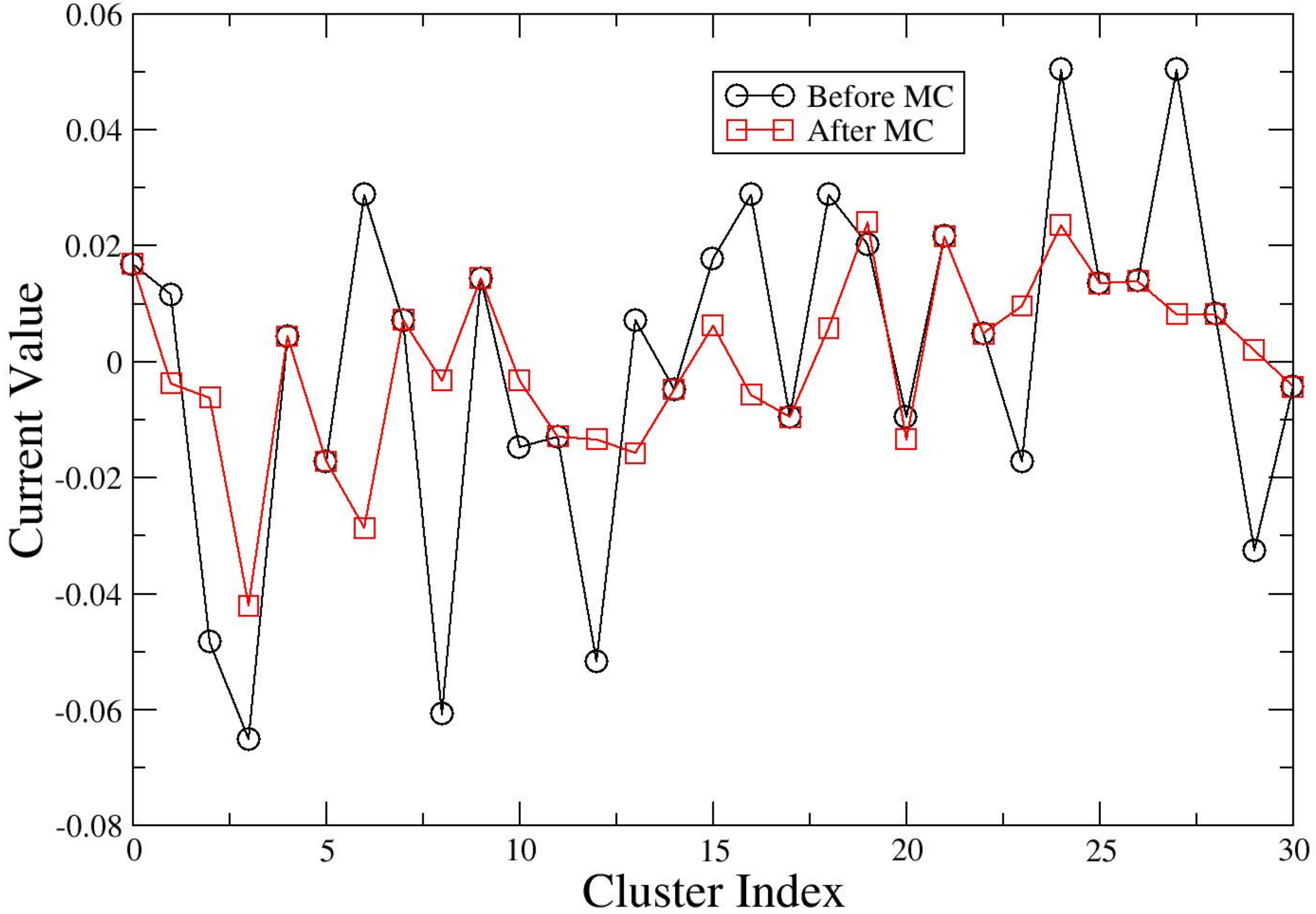
1. Scan over all *load* vertices *i* and connect each to its nearest *generator j* (i.e., $\forall$ *loads i* connect to *generator j* such that R_{ij} = Min.)
2. Run Monte Carlo Simulated Annealing, trying to move each original load to neighboring cluster to maximize ***E***.
3. Build new network with each old cluster as a new vertex.
4. Separate new vertices into super-generators ($I > 0$) and super-loads ($I < 0$).
5. Return to 1. (But note, in MC, the *original* (small) vertices are moved, *not* the "supervertices."

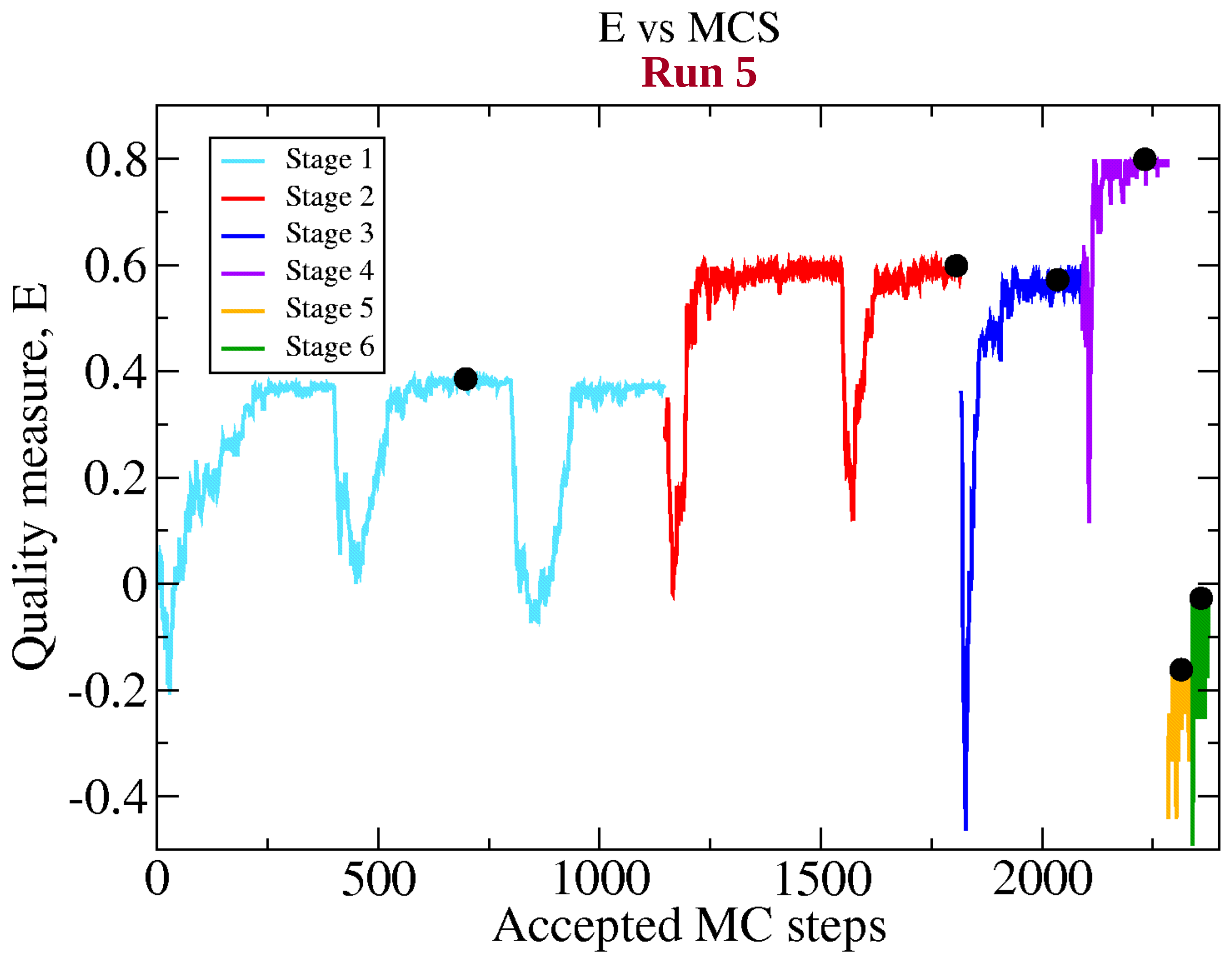
Monte Carlo Simulated Annealing

- Randomly select a peripheral load
- Randomly select a neighboring cluster to move it to
- Calculate ΔE
- If $\Delta E > 0$ *provisionally* accept the move
- If $\Delta E < 0$ *provisionally* accept the move with probability $e^{(\Delta E/T)}$
- Check that acceptance will not break the cluster, and *permanently accept* if OK.
  - Otherwise, *reject*.
- Decrease T and repeat
- When $T \sim 0$ reset to initial T
- Save configuration with Max E



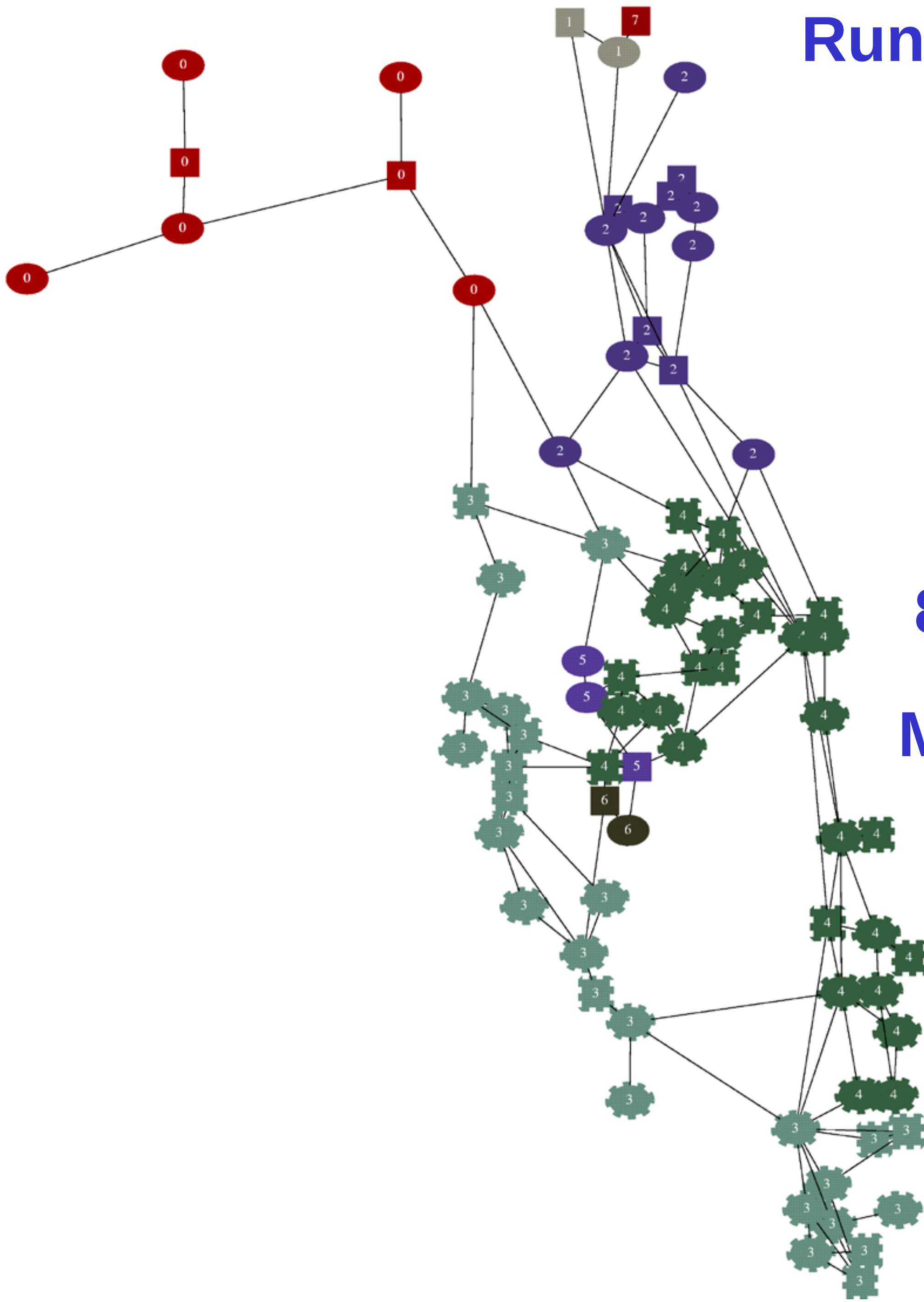
In/Out Current Vector Before and After MC





E vs MCS
Run 5
Stage 1
Stage 2
Stage 3
Stage 4
Stage 5
Stage 6
0.8
0.6
0.4
0.2
0
-0.2
-0.4
Quality measure, E
0
500
1000
1500
2000
Accepted MC steps

Run 5

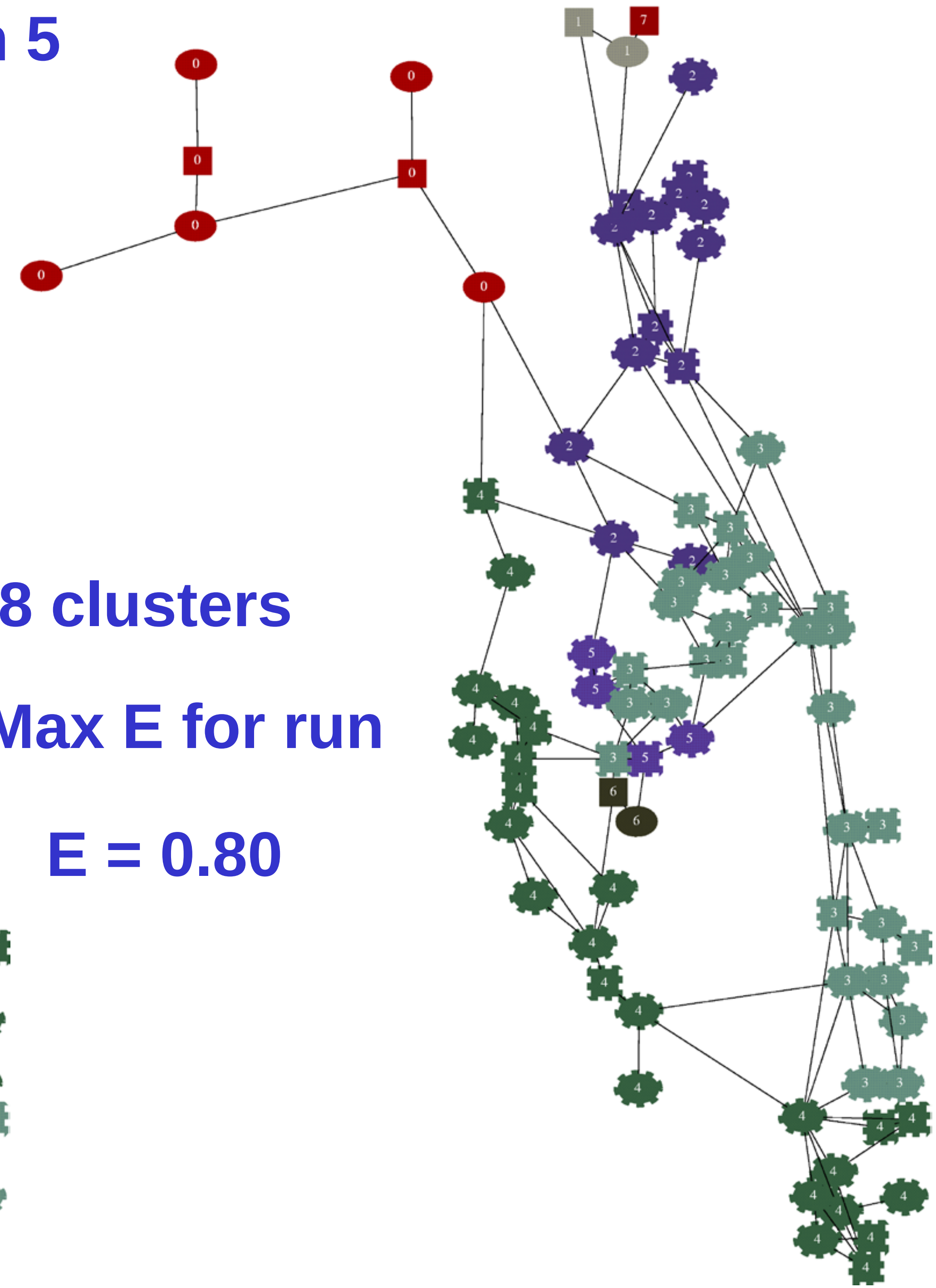


4 Before MC

8 clusters

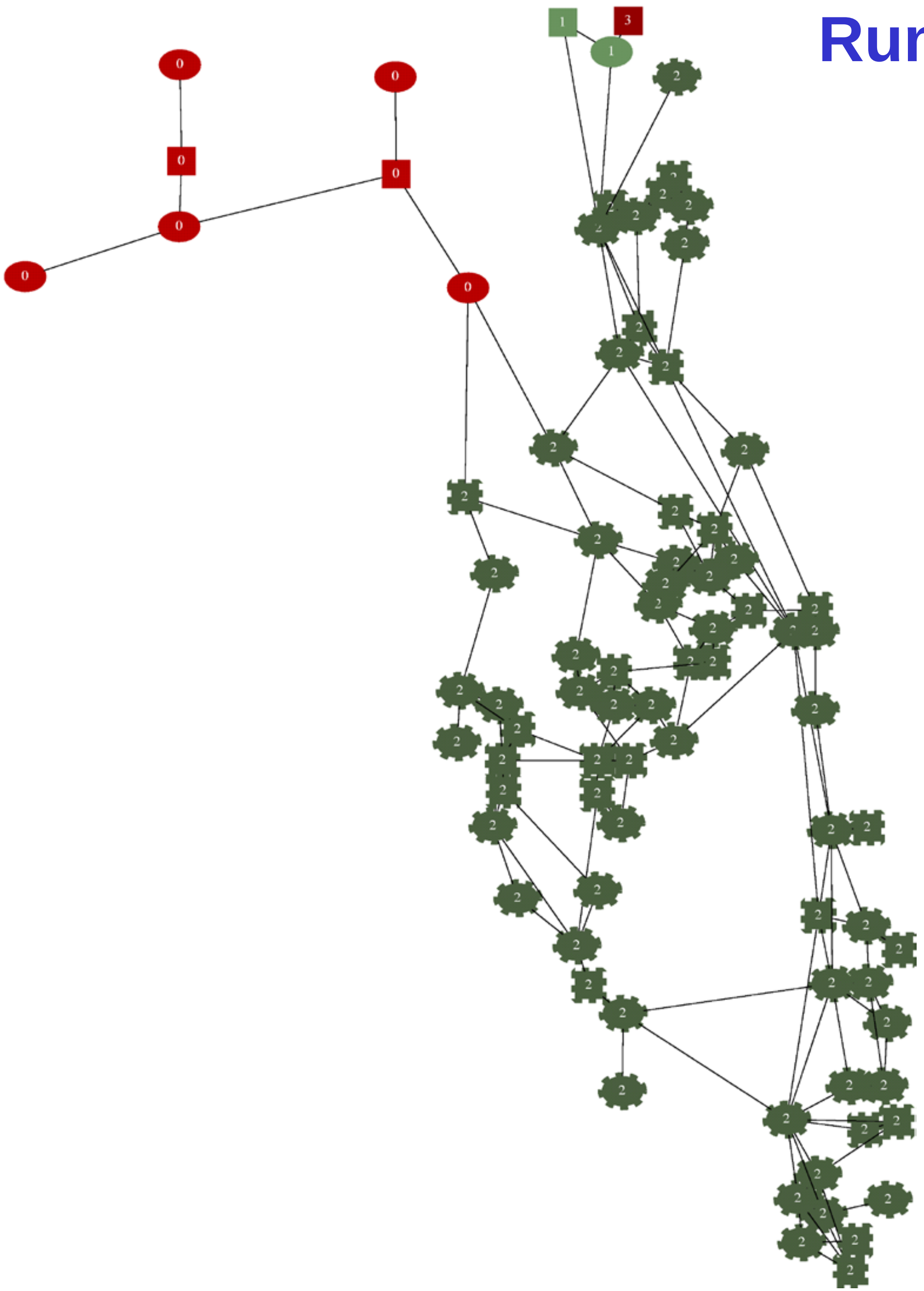
Max E for run

E = 0.80



4 After MC

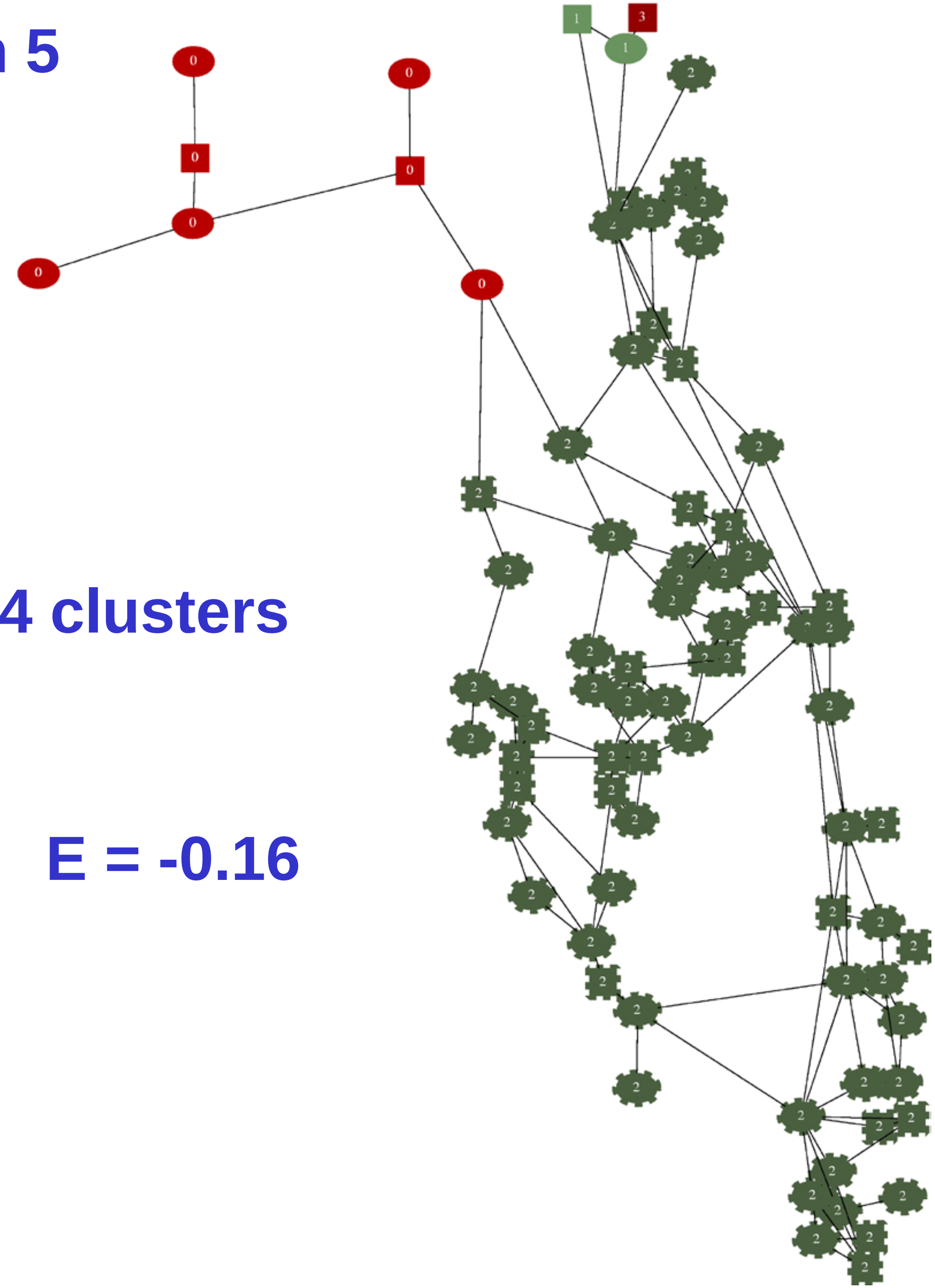
Run 5



5 Before MC

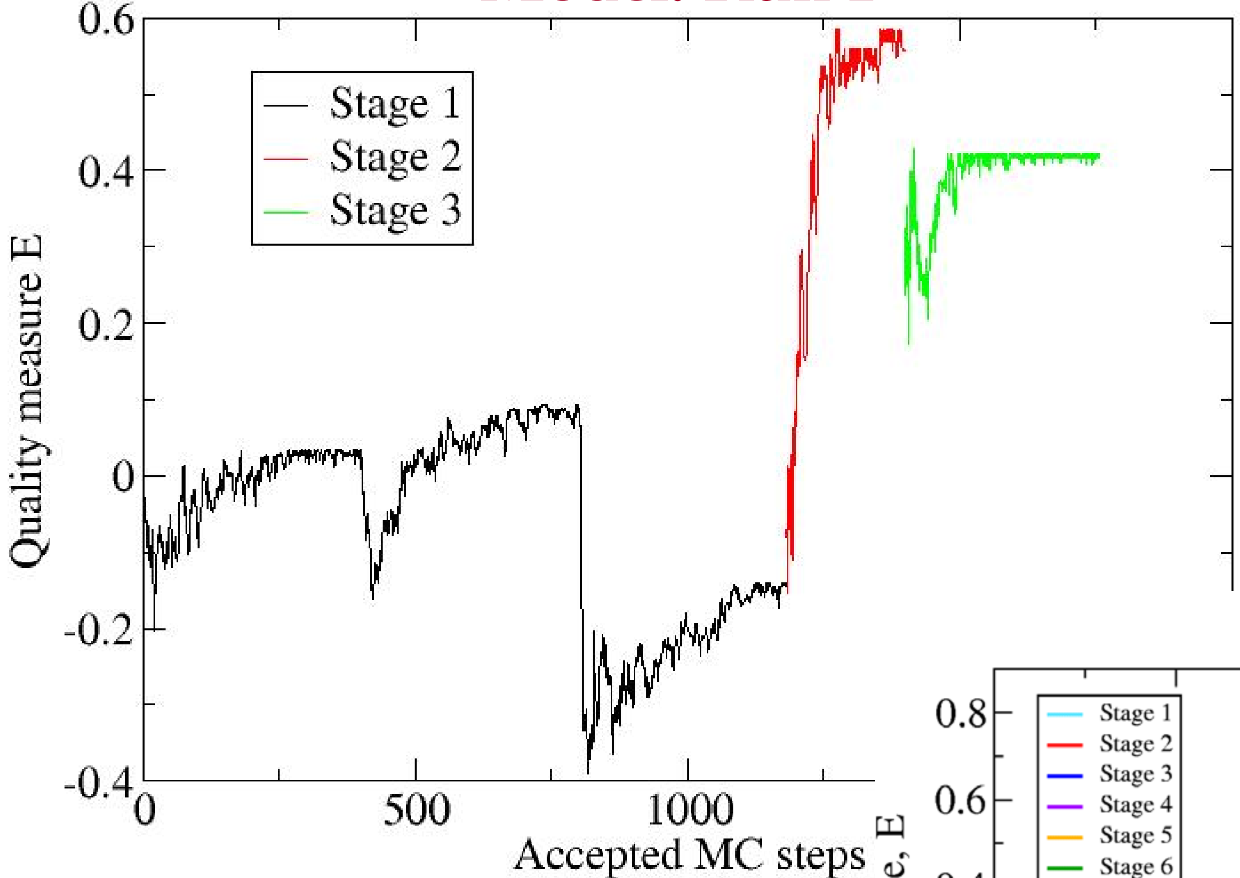
4 clusters

E = -0.16

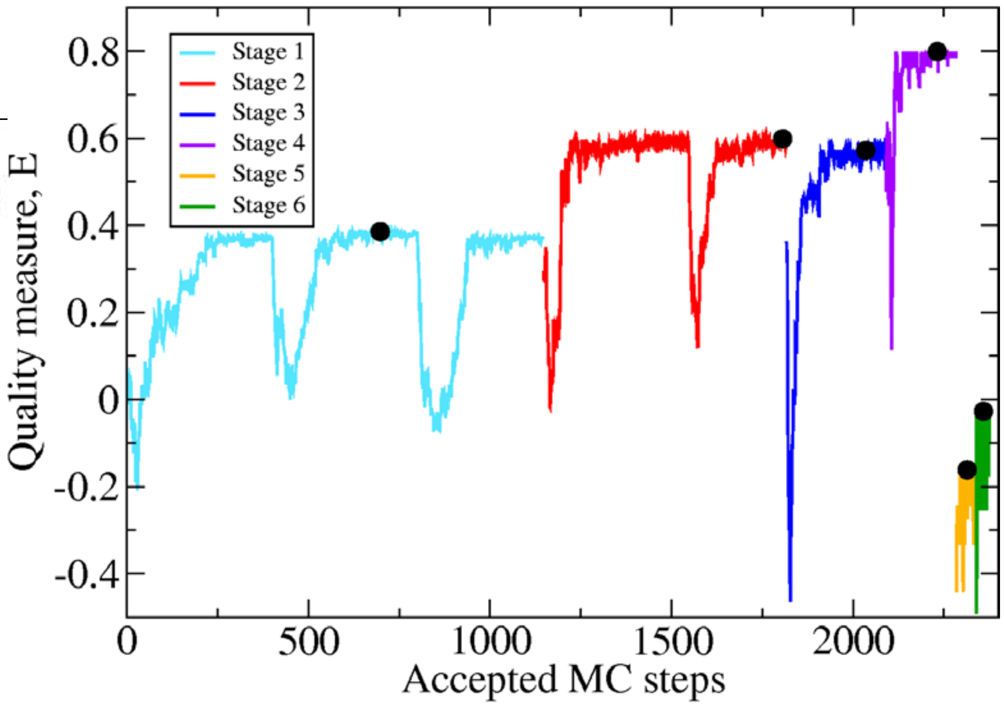


5 After MC

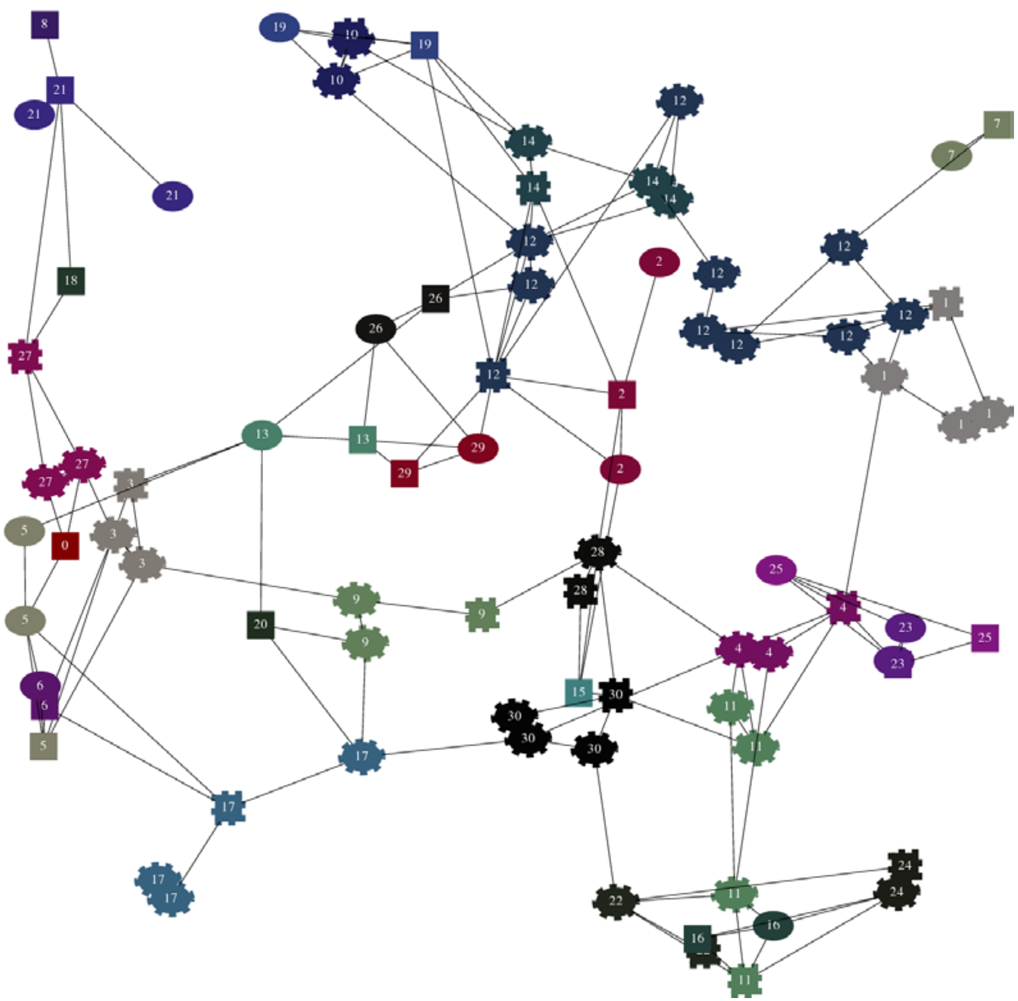
Model. Run F



Florida. Run 5



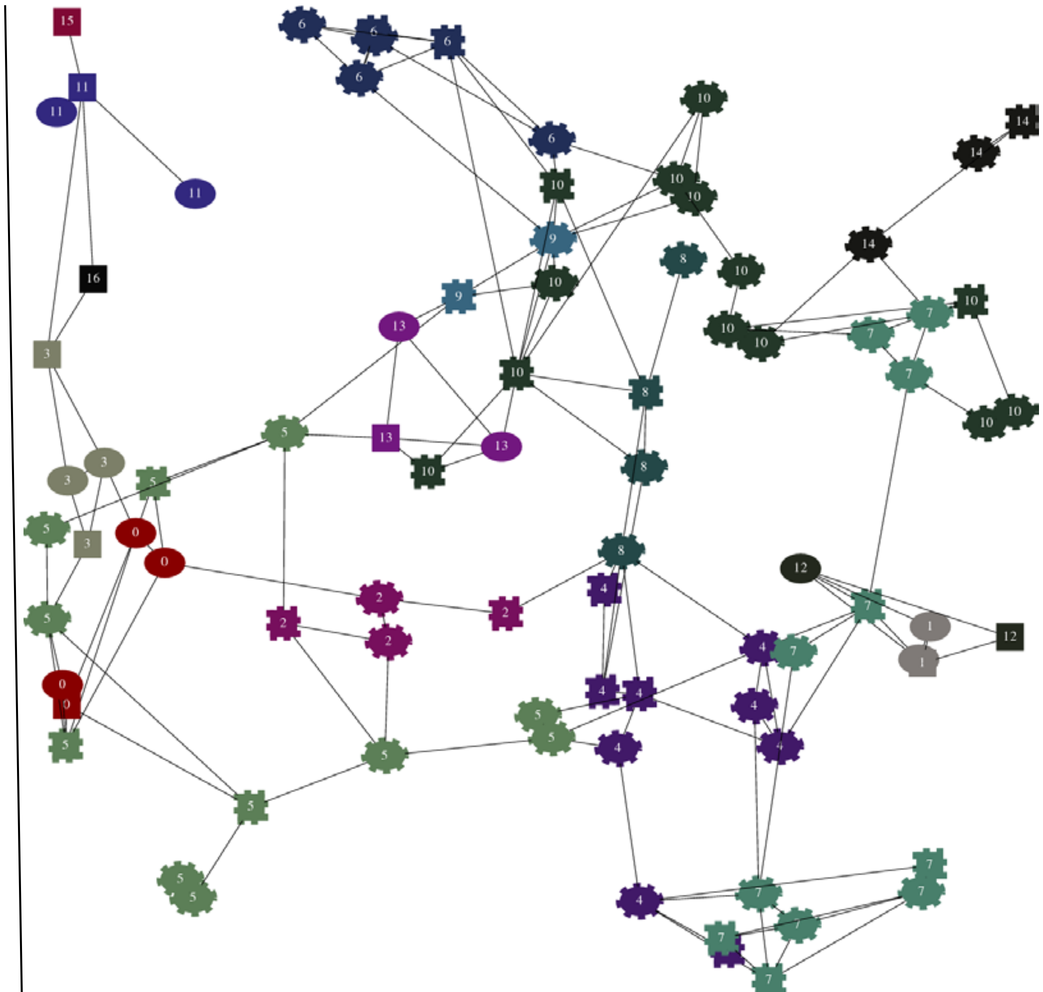
Run F



31 clusters

E = 0.09

1 After MC

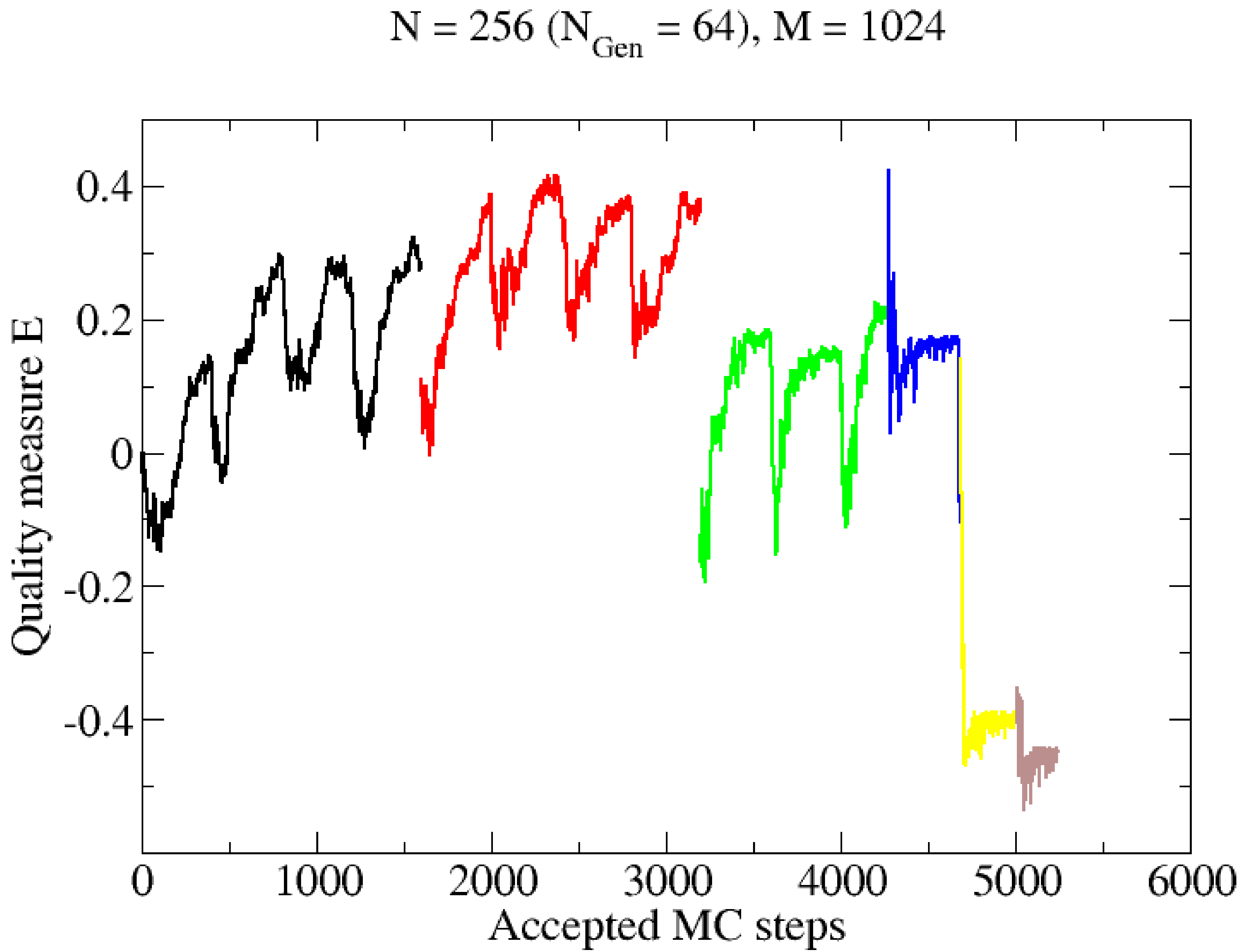


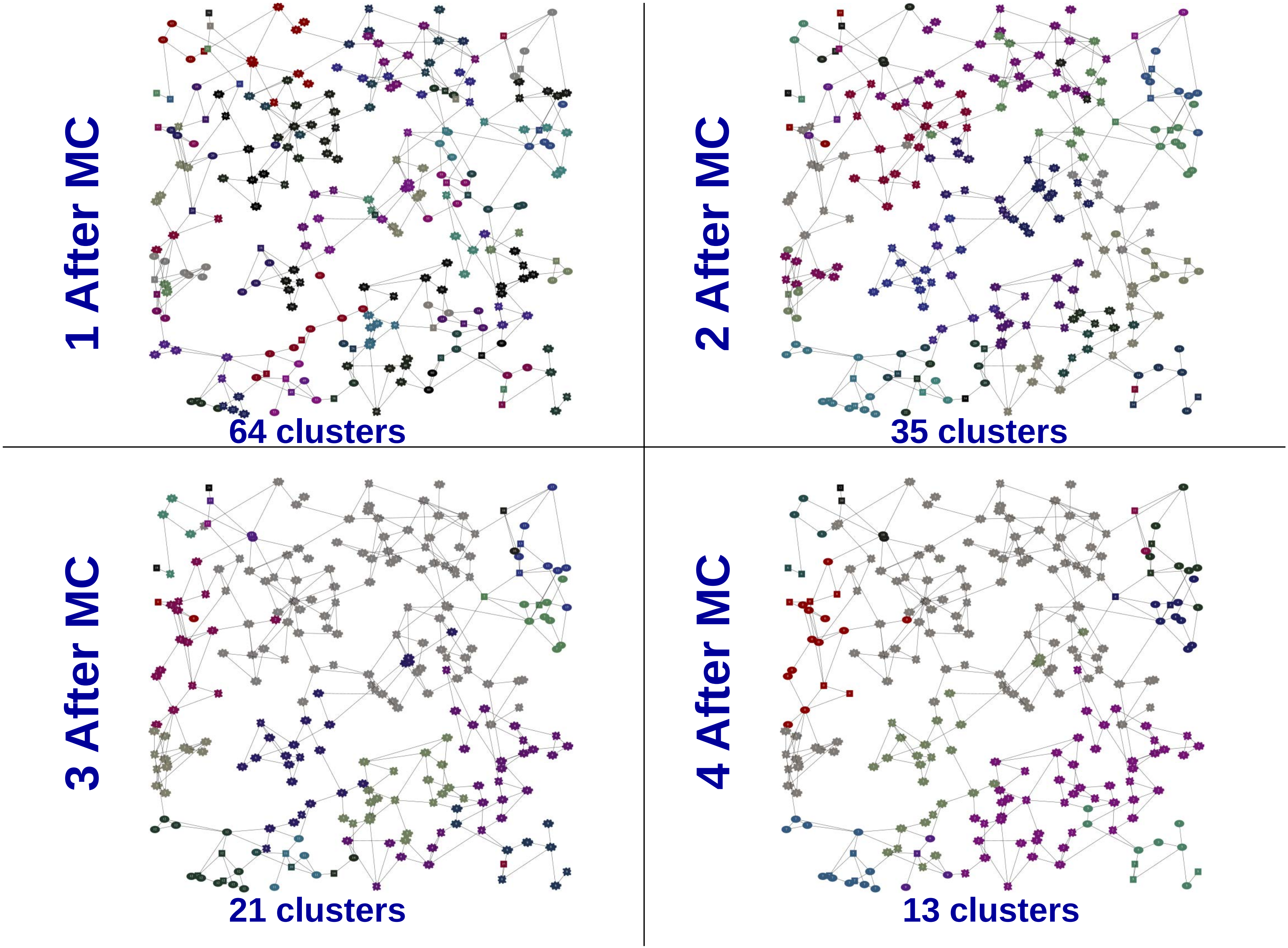
17 clusters

E = 0.59

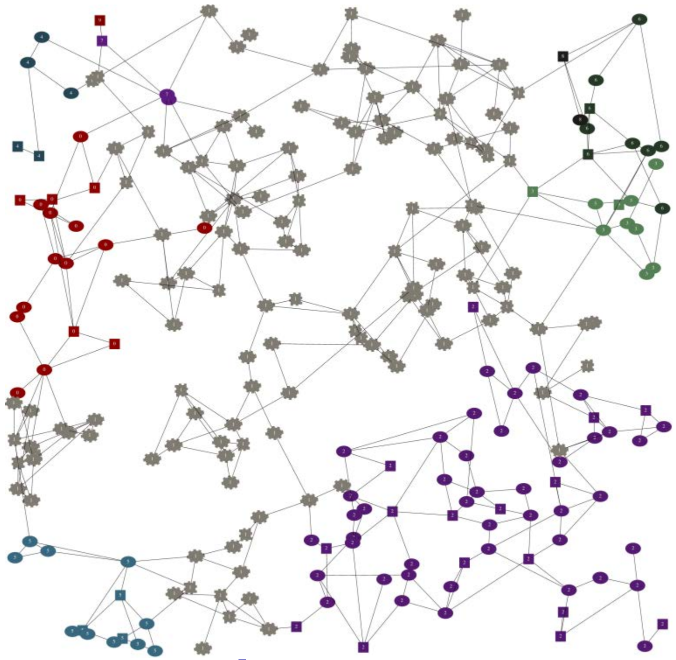
2 After MC

Scaling it up


N = 256 (N_Gen = 64), M = 1024
Quality measure E
Accepted MC steps
0.4
0.2
0
-0.2
-0.4
0
1000
2000
3000
4000
5000
6000

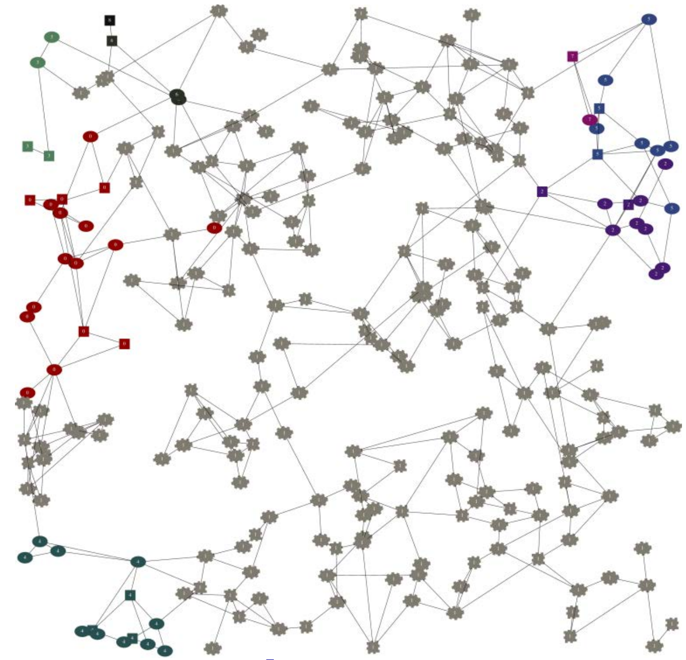

1 After MC
64 clusters
2 After MC
35 clusters
3 After MC
21 clusters
4 After MC
13 clusters

5 After MC



10 clusters

6 After MC



9 clusters

Remarks and Conclusions

- Developed simple *model power-grids* that enable us to experiment with partitioning algorithms on grids with different characteristics.
- Used network theory to partition the model power-grid network taking into account the generating power of each of the power plants.
- Used MC simulated annealing to optimize the resulting clusters for better internal connectivity and power self-sufficiency.
- The approach can be scaled to larger grids.

Thank you, Yousuff!

Publications

- I. Abou Hamad, B. Israels, P. A. Rikvold, and S. V. Poroseva. ``Spectral Matrix Methods for Partitioning Power Grids: Applications to the Italian and Floridian High-voltage Networks." *Physics Procedia* **4**, 125-129 (2010).

- I. Abou Hamad, P. A. Rikvold, and S. V. Poroseva. ``Floridian High-voltage Power-grid Network Partitioning and Cluster Optimization Using Simulated Annealing." *Physics Procedia* **15**, 2-6 (2011).

- P. A. Rikvold, I. Abou Hamad, B. Israels, and S. V. Poroseva. ``Modeling Power Grids." *Physics Procedia* **34**, 119-123 (2012).