Peer-to-Peer Networks
09 Random Graphs for Peer-to-Peer-Networks

Christian Schindelhauer
Technical Faculty
Computer-Networks and Telematics
University of Freiburg

CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG

Push & Pull ≠ Push & Pull

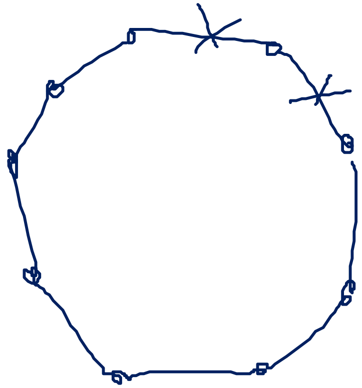
information spreading

Random Networks

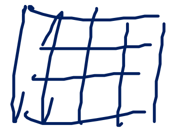
UNI FREIBURG

CoNe Freiburg

Peer-to-Peer Networking Facts



- Hostile environment
  - Legal situation
  - Egoistic users
  - Networking
    - ISP filter Peer-to-Peer Networking traffic
    - User arrive and leave
    - Several kinds of attacks
    - Local system administrators fight peer-to-peer networks
- Implication
  - Use stable robust network structure as a backbone
  - Napster: star
  - CAN: lattice
  - Chord, Pastry, ~~Tapestry~~: ring + pointers for lookup
  - Gnutella, FastTrack: chaotic “social” network

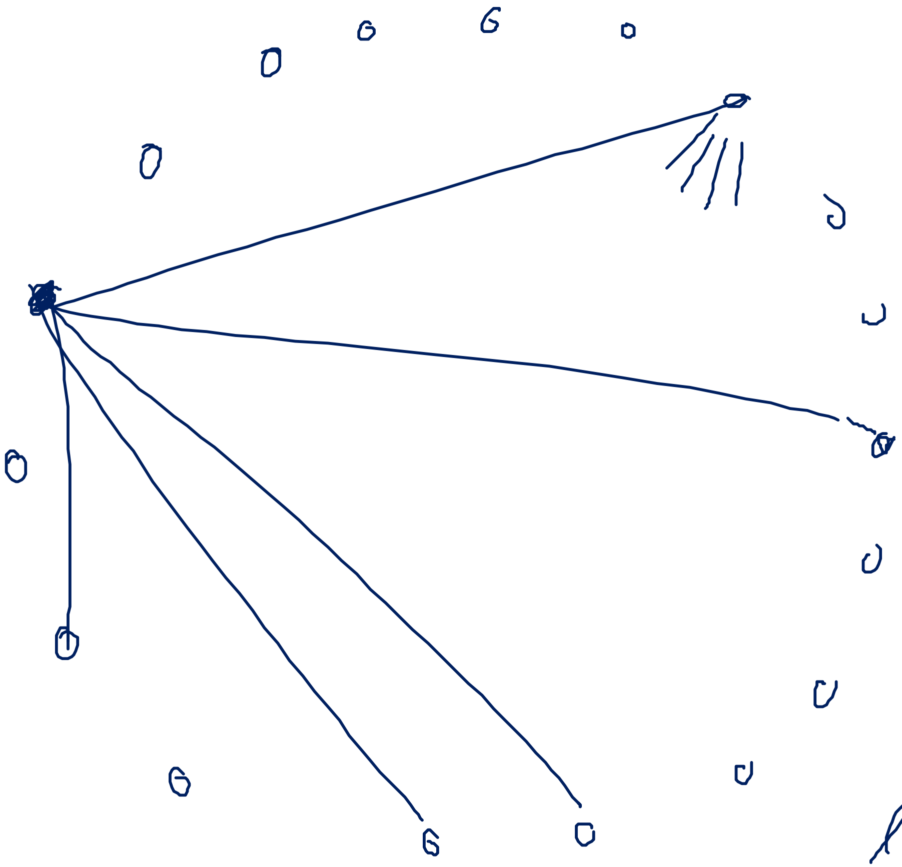


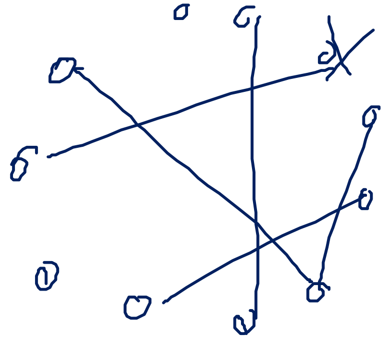
- Idea: Use a Random d-regular Network

UNI FREIBURG

CoNe Freiburg

Erdös - Renyi





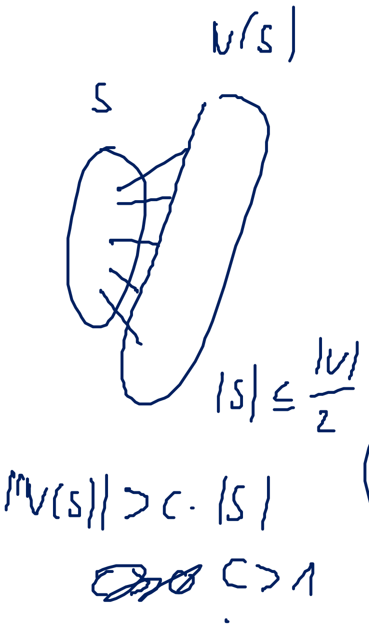
$\Omega(n \cdot \log n)$ edges to get

λ - connectivity

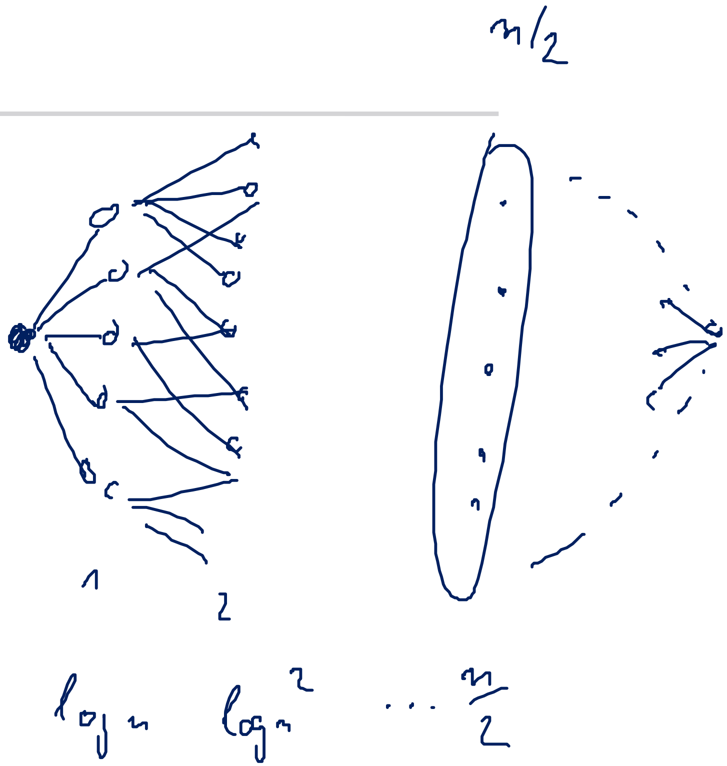
log n -

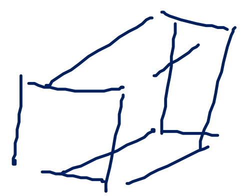
UNI FREIBURG

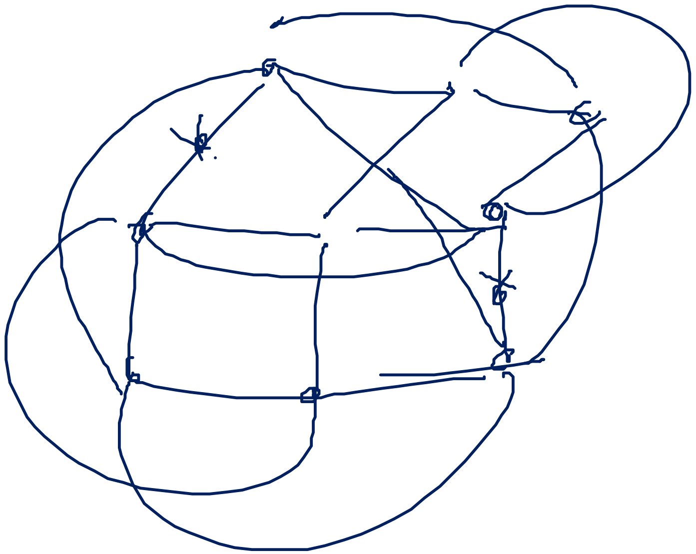
Why Random Networks ?

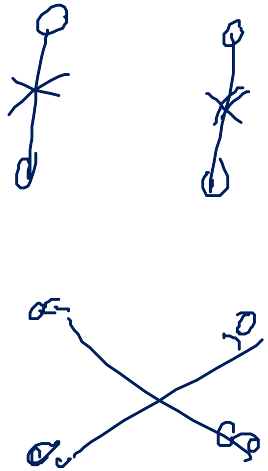


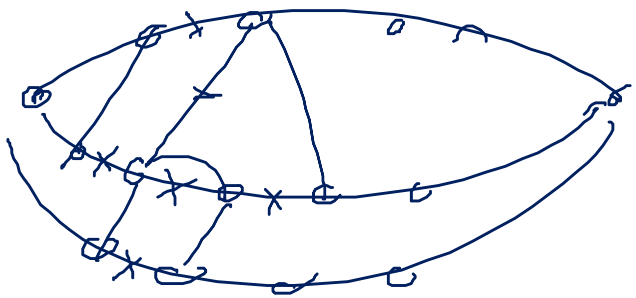
- Random Graphs ...
  - Robustness
  - Simplicity
  - Connectivity
  - Diameter
  - Graph expander
  - Security
- Random Graphs in Peer-to-Peer networks:
  - Gnutella
  - JXTApose











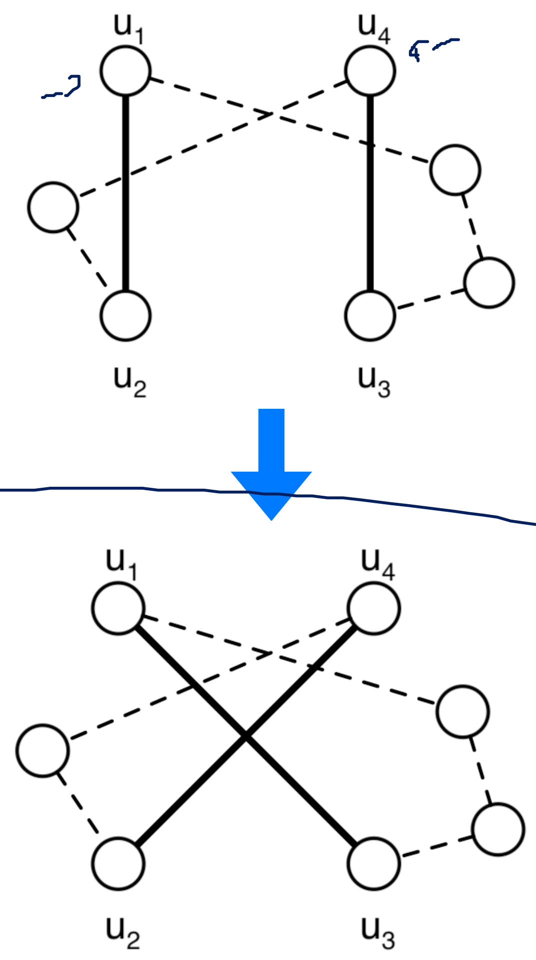
Dynamic Random Networks ...

- Peer-to-Peer networks are highly dynamic ...
  - maintenance operations are needed to preserve properties of random graphs
  - which operation can maintain (repair) a random digraph?

Desired properties:	
Soundness	Operation remains in domain (preserves connectivity and out-degree)
Generality	every graph of the domain is reachable does not converge to specific small graph set
Feasibility	can be implemented in a P2P-network
Convergence Rate	probability distribution converges quickly

Simple Switching

- Simple Switching
  - choose two random edges
    - $\{u_1,u_2\} \in E$, $\{u_3,u_4\} \in E$
  - such that $\{u_1,u_3\}$, $\{u_2,u_4\} \notin E$
    - add edges $\{u_1,u_3\}$, $\{u_2,u_4\}$ to E
    - remove $\{u_1,u_2\}$ and $\{u_3,u_4\}$ from E
- McKay, Wormald, 1990
  - Simple Switching converges to uniform probability distribution of random network
  - Convergence speed:
    - $O(nd^3)$ for $d \in O(n^{1/3})$
- Simple Switching cannot be used in Peer-to-Peer networks
  - Simple Switching disconnects the graph with positive probability
  - No network operation can re-connect disconnected graphs



Necessities of Graph Transformation

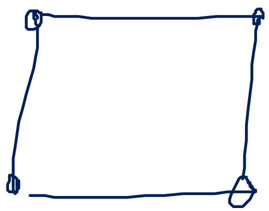
	Simple-Switching
Graphs	Undirected Graphs
Soundness	✓
Generality	✓
Feasibility	✔
Convergence	✔

d-regular graph:
every node has d neighbors
= degree d

- Problem: Simple Switching does not preserve connectivity
- Soundness
  - Graph transformation remains in domain
  - Map connected d-regular graphs to connected d-regular graphs
- Generality
  - Works for the complete domain and can lead to any possible graph
- Feasibility
  - Can be implemented in P2P network
- Convergence Rate
  - The probability distribution converges quickly

Directed Random Graphs

- Peter Mahlmann, Christian Schindelhauer
  - Distributed Random Digraph Transformations for Peer-to-Peer Networks, 18th ACM Symposium on Parallelism in Algorithms and Architectures, Cambridge, MA, USA. July 30 - August 2, 2006



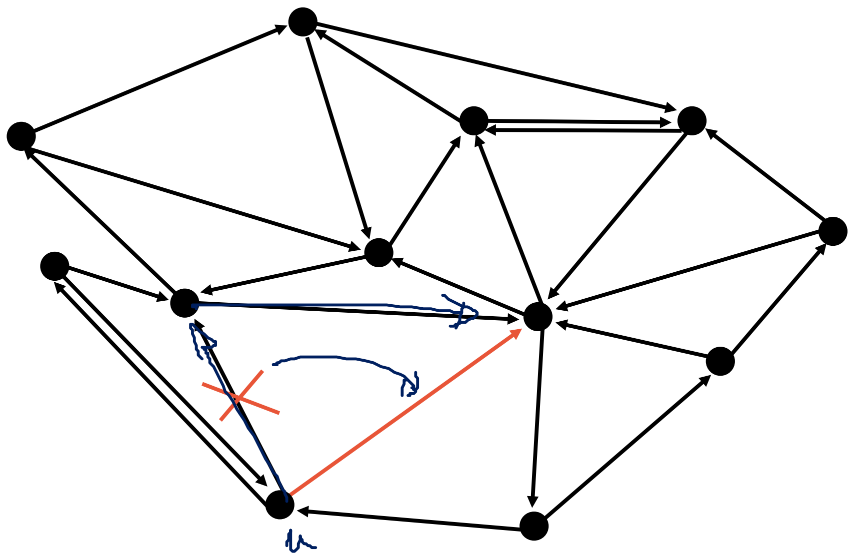
Directed Graphs

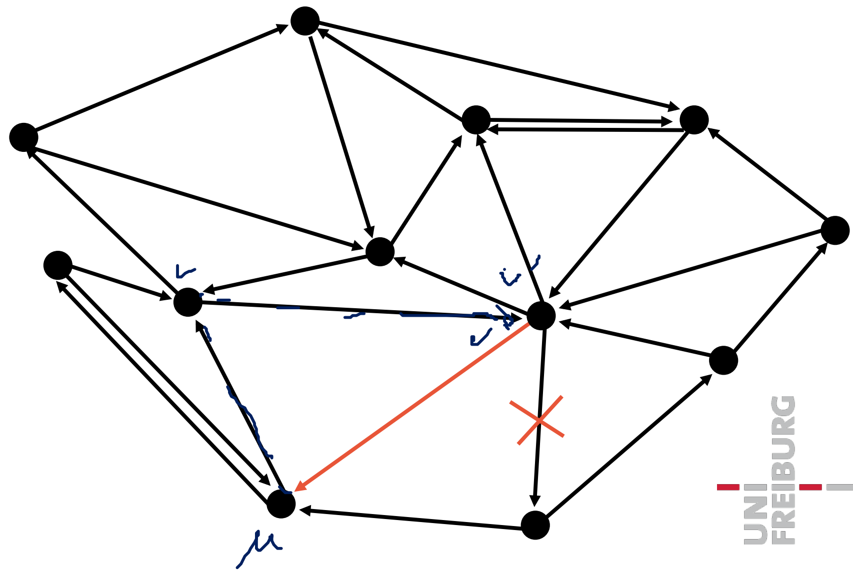
Push Operation:

1. Choose random node *u*
2. Set *v* to *u*
3. While a random event with p= 1/*h* appears
   a) Choose random edge starting at *v* and ending at *v*‘
   b) Set *v* to *v*‘
3. Insert edge (*u*,*v*)
4. Remove random edge starting at *v*

Pull Operation:

1. Choose random node *u*
2. Set *v* to *u*
3. While a random event with p= 1/*h* appears
   a) Choose random edge starting at *v* and ending at *v*‘
   b) Set *v* to *v*‘
3. Insert edge (*v*,*u*)
4. Remove random edge starting at *v*‘





Simulation of Push-Operations

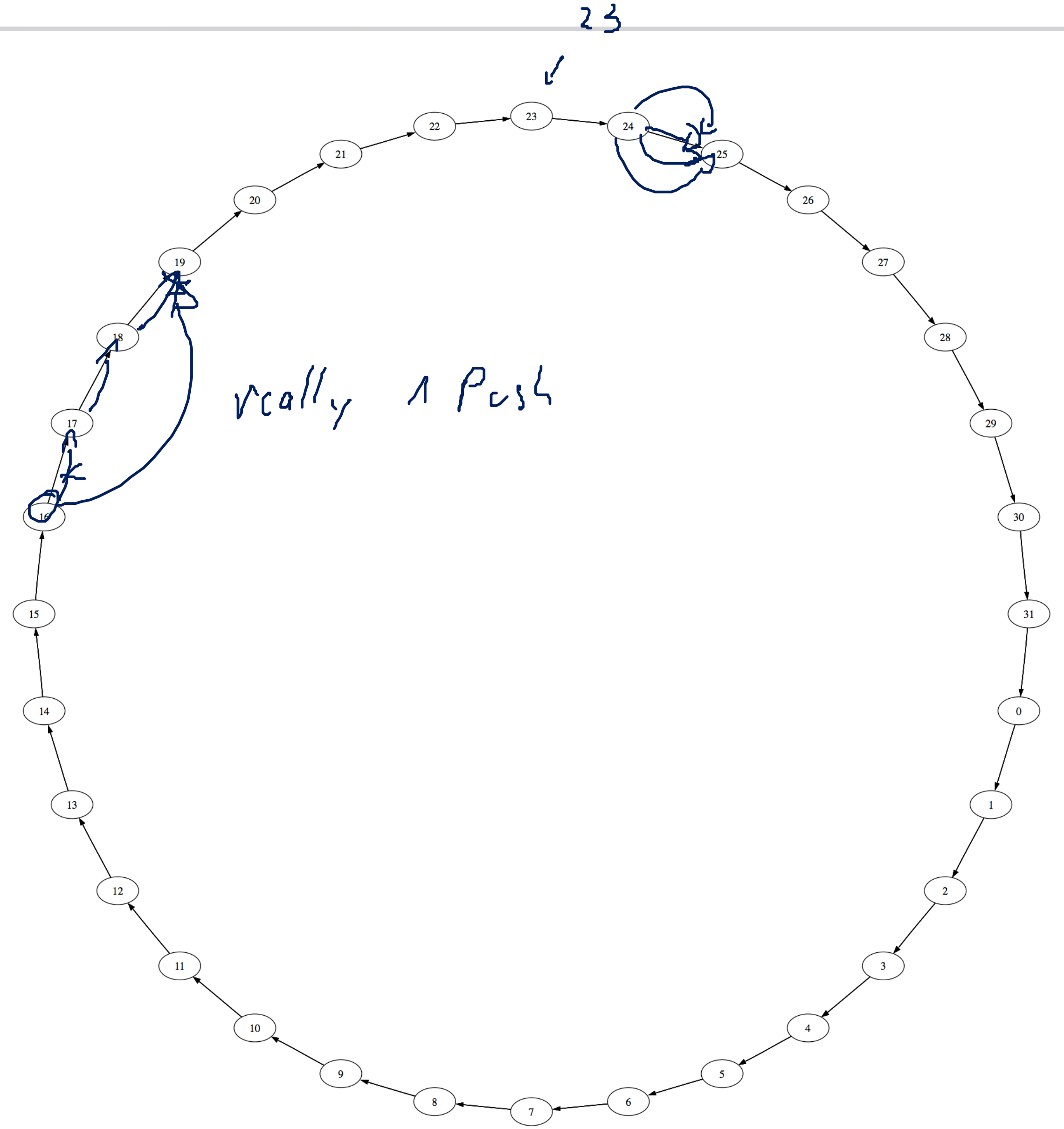
Start situation

Parameter:

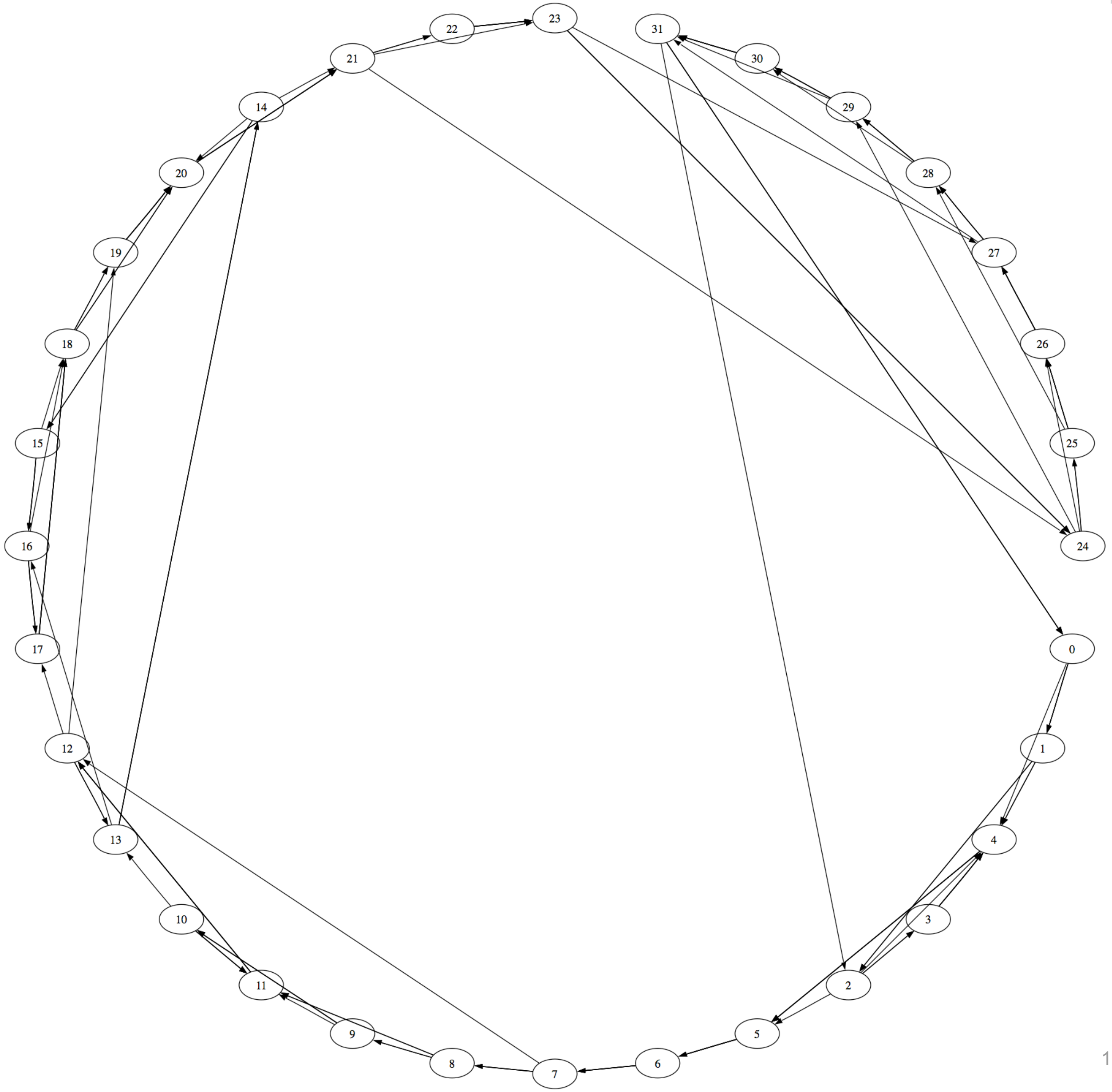
n = 32 nodes

out-degree d = 4

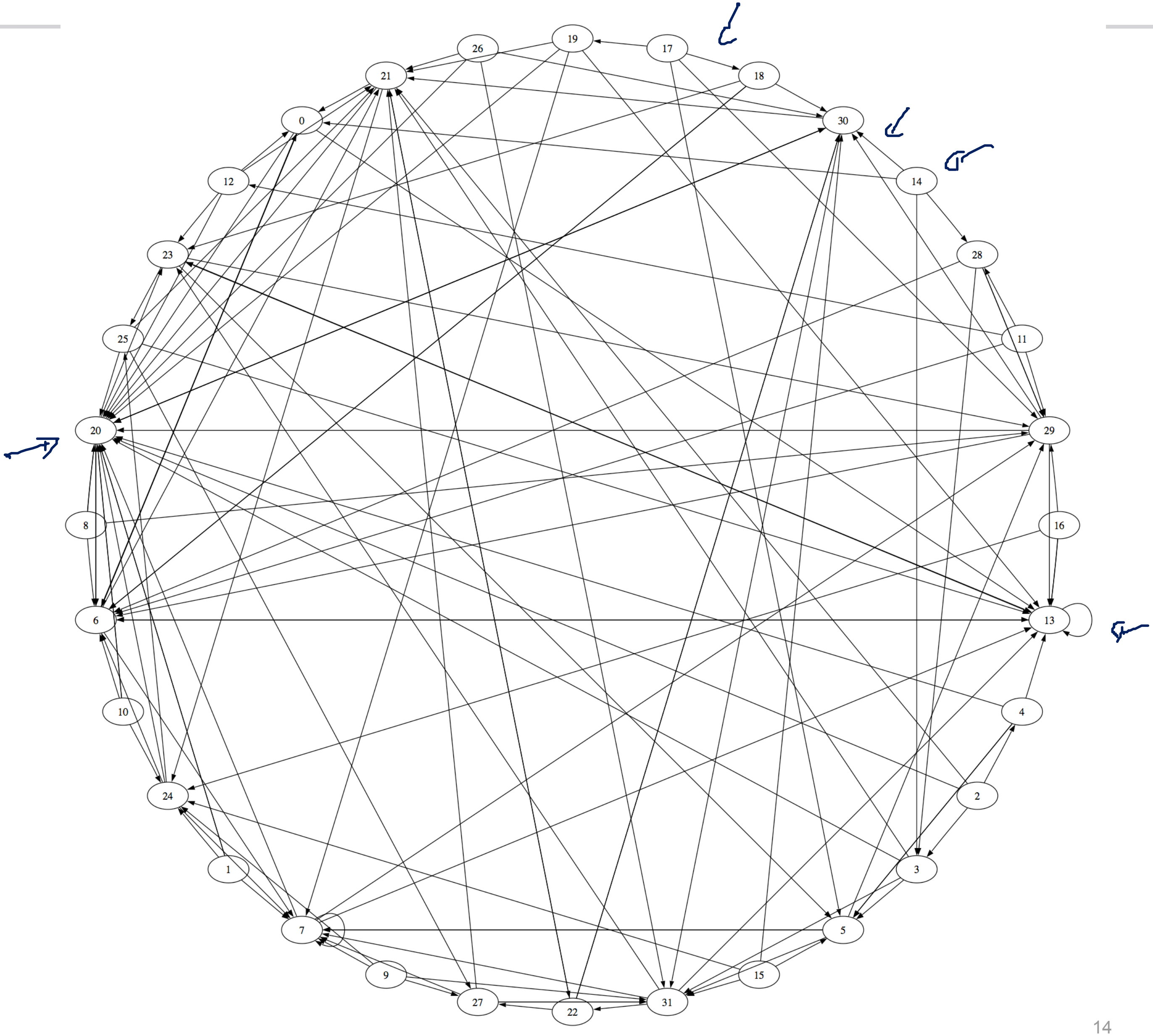
Hop-distance h = 3



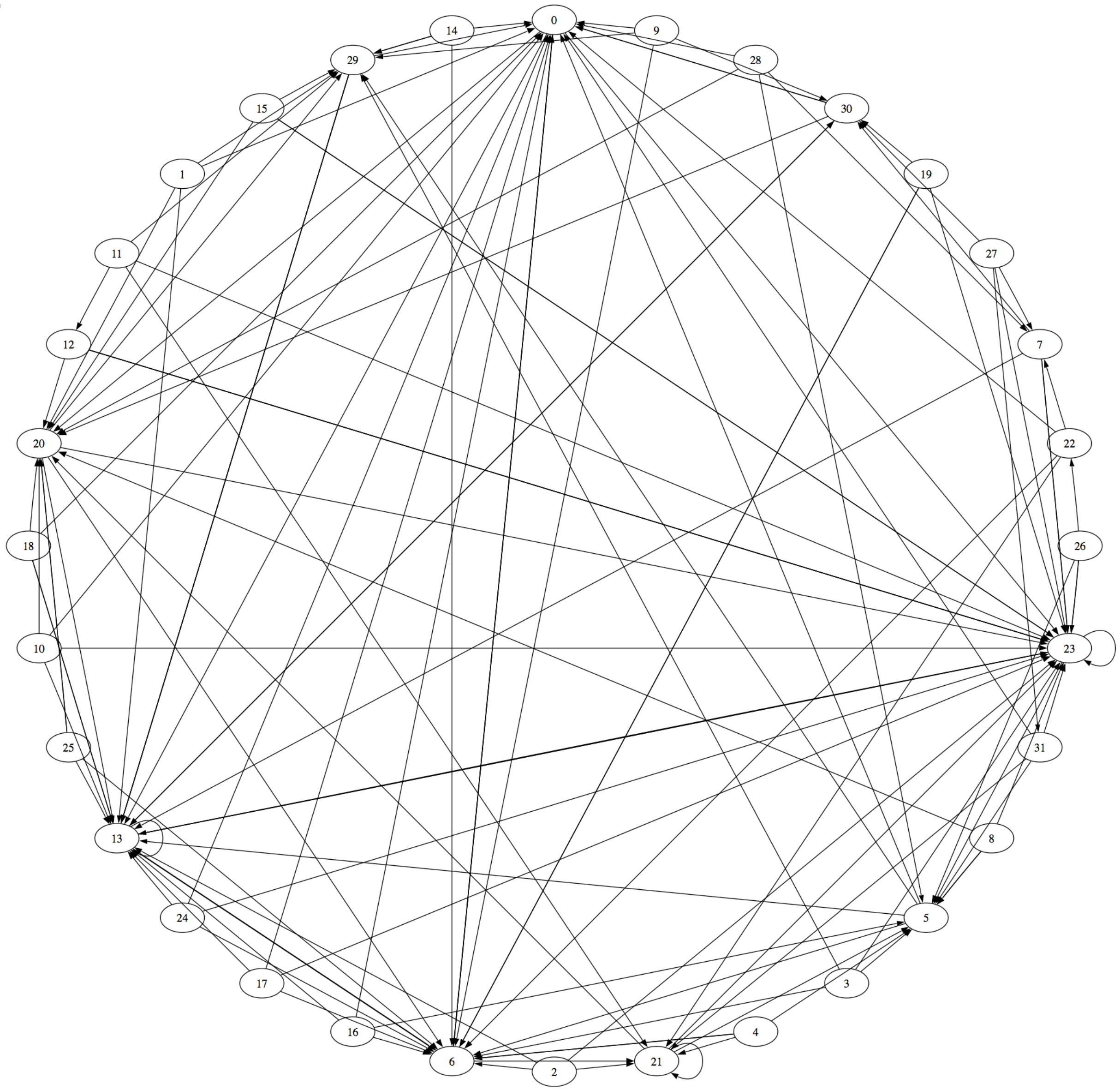
1 Iteration Push ... = m x Push



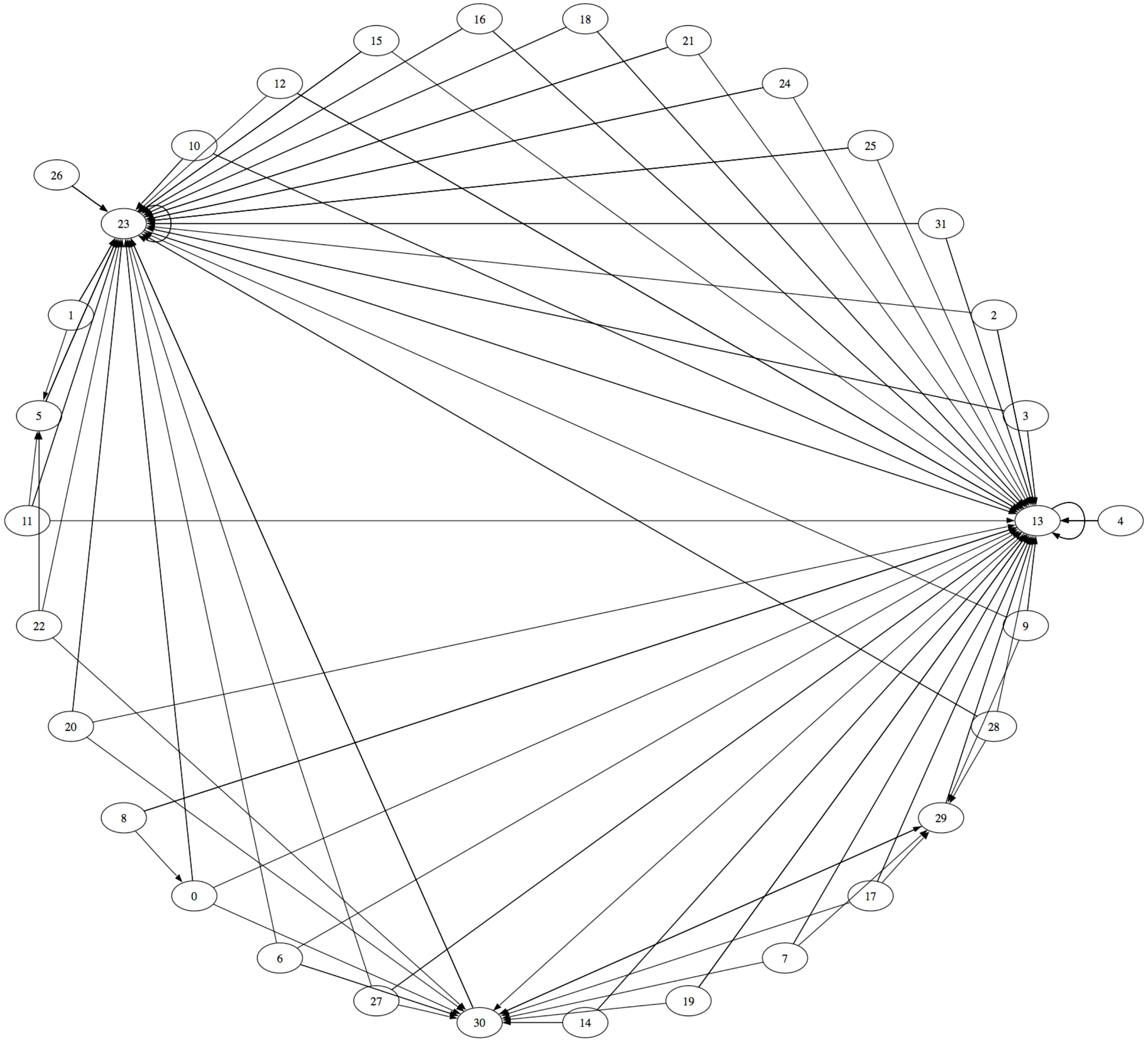
10 Iterations Push ...



20 Iterations von Push ...

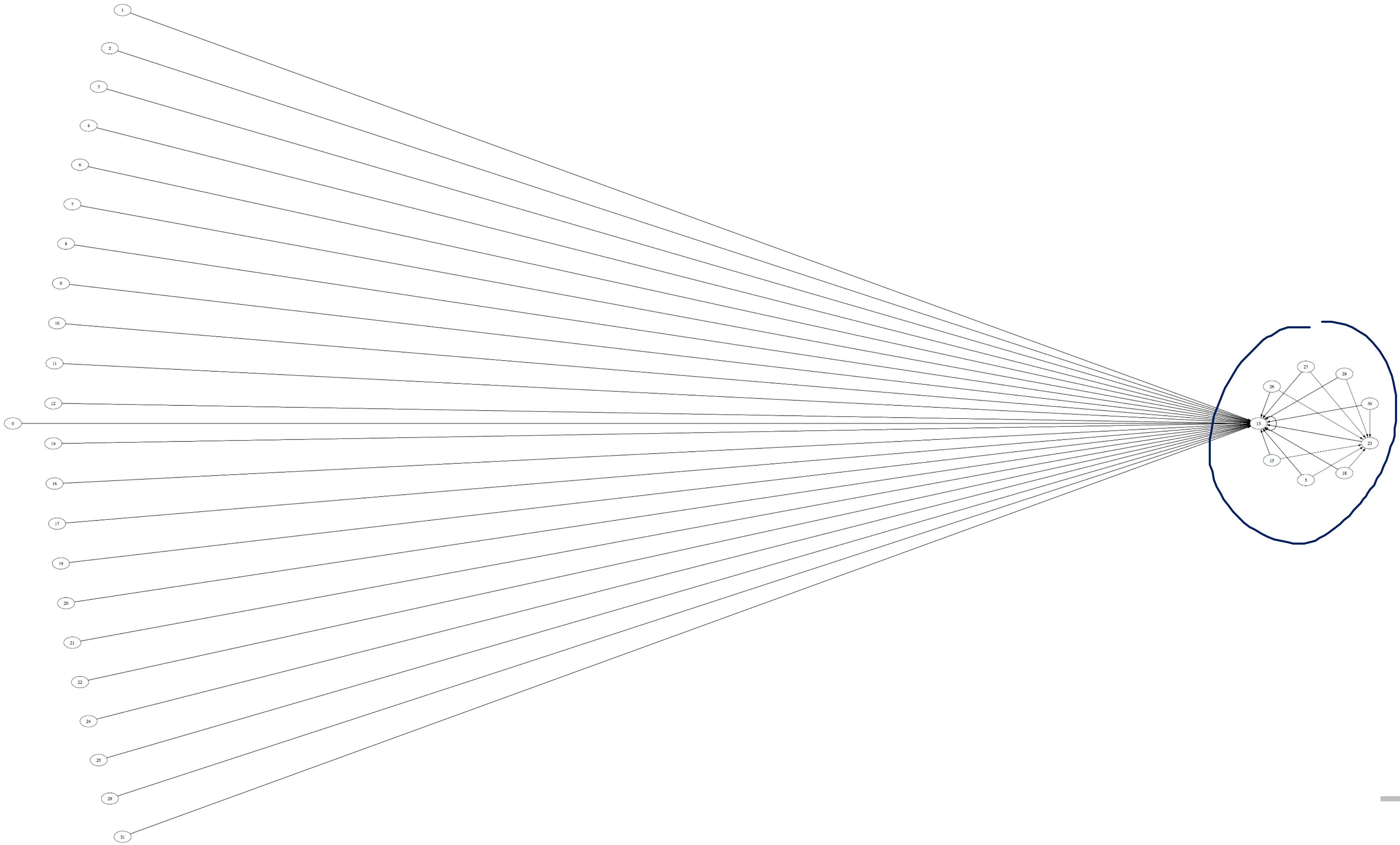


30 Iterations Push ...

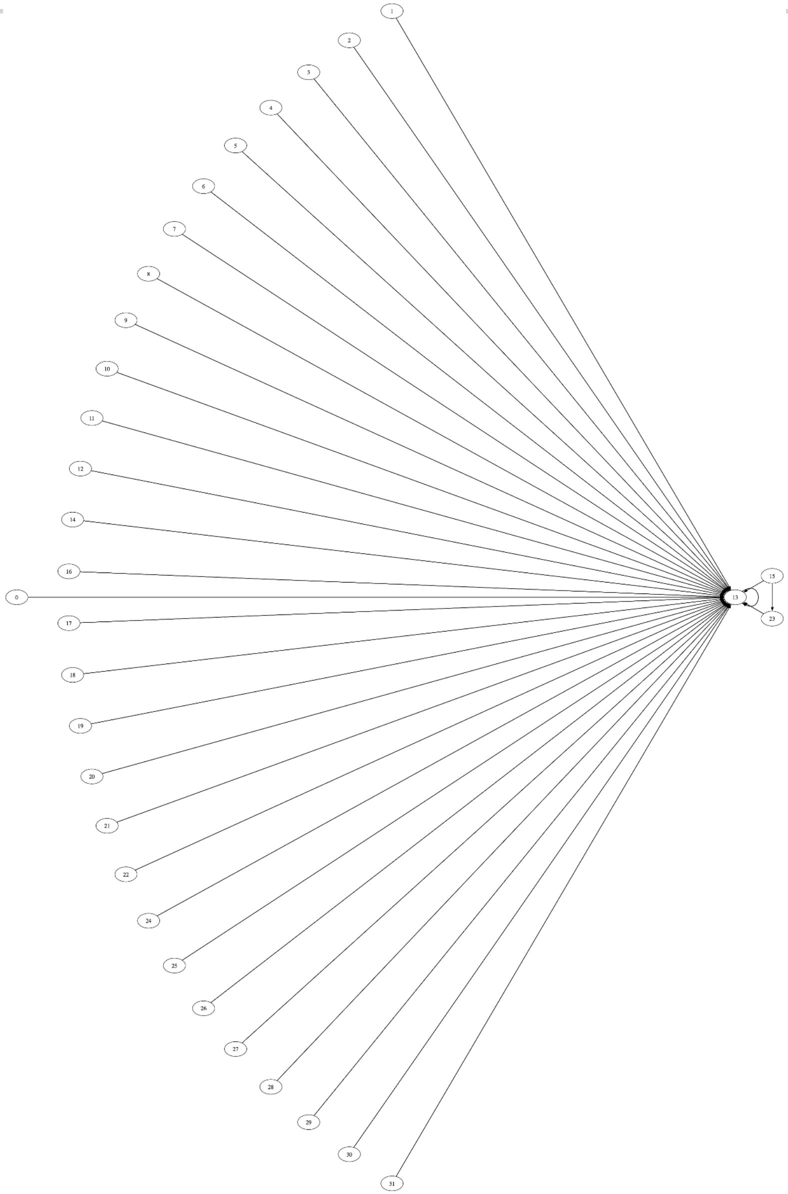


CoNe Freiburg

40 Iterations Push ...

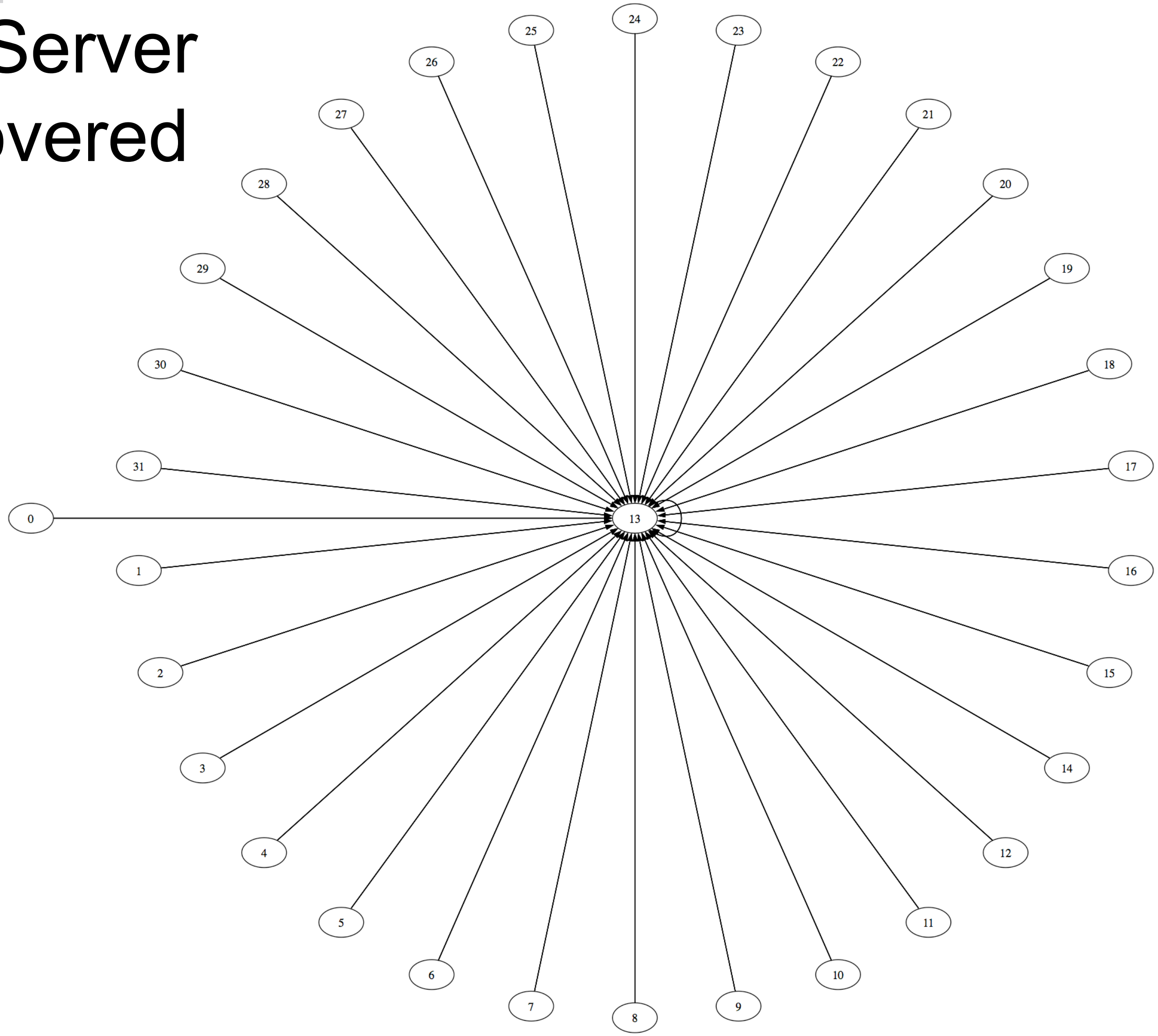


50 Iterations Push ...



70 Iterations Push ...

Client-Server rediscovered



Simulation of Pull-Operation ...

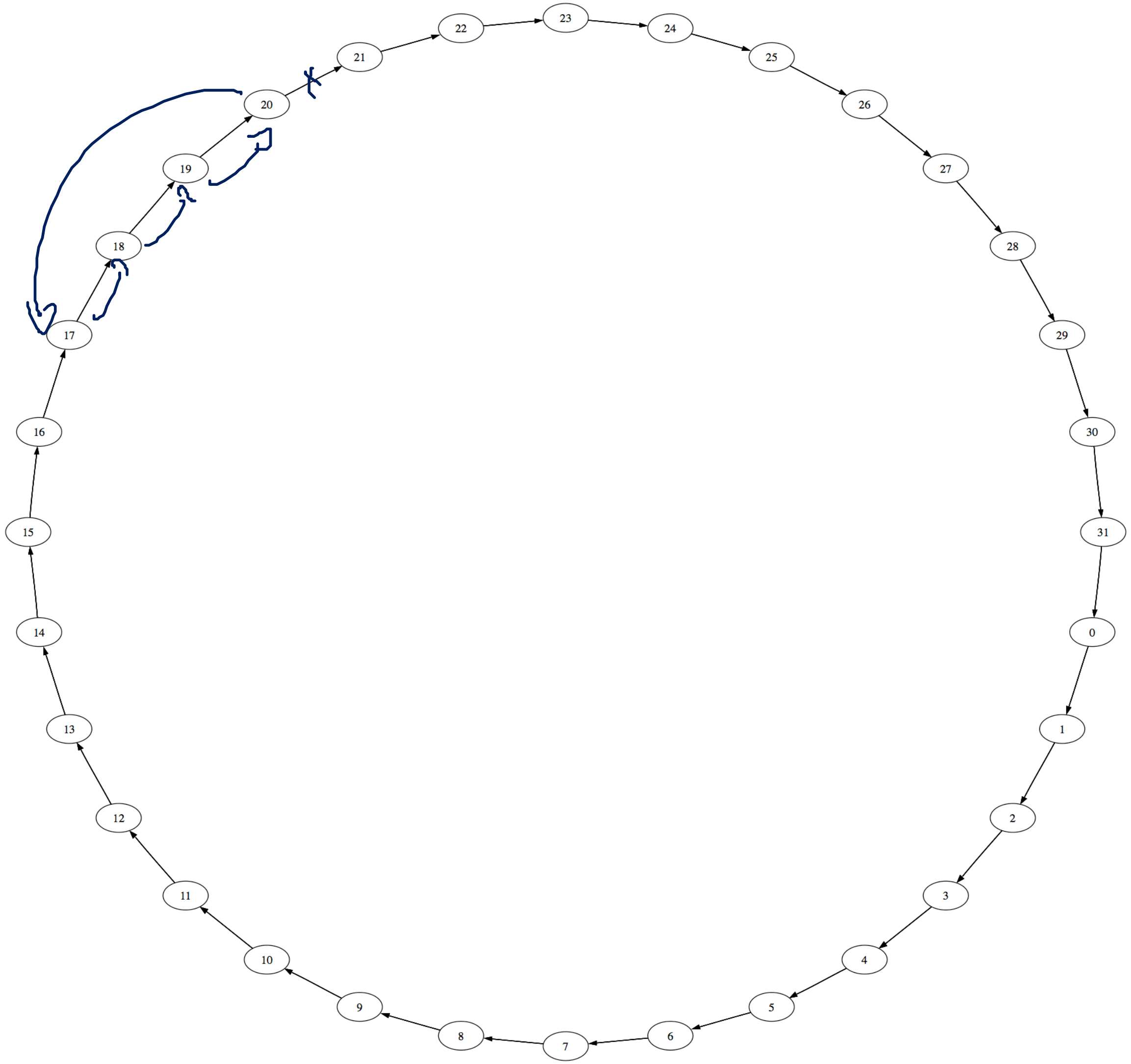
Start situation

Parameter:

n = 32 nodes

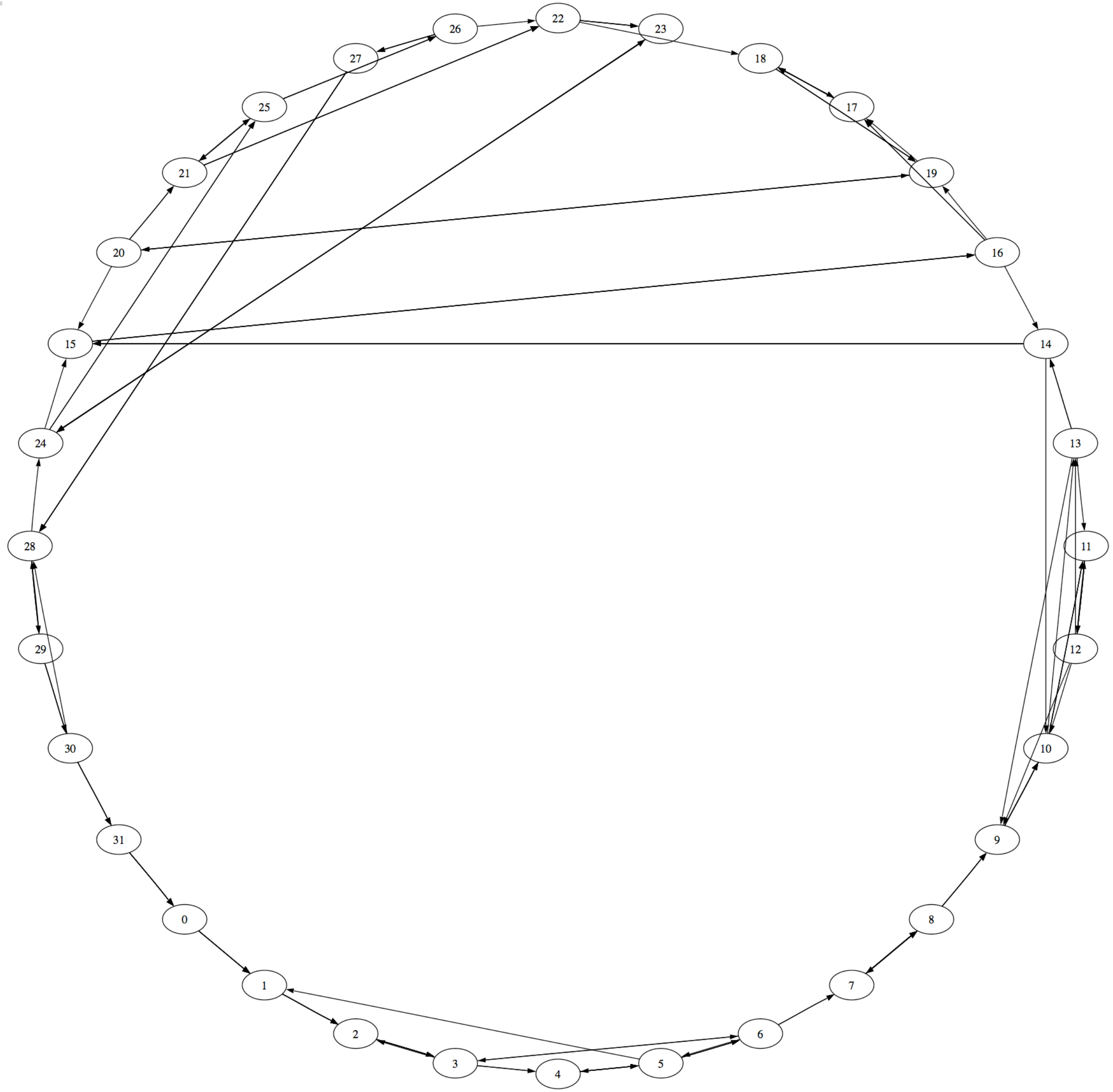
outdegree d = 4

hop distance h = 3

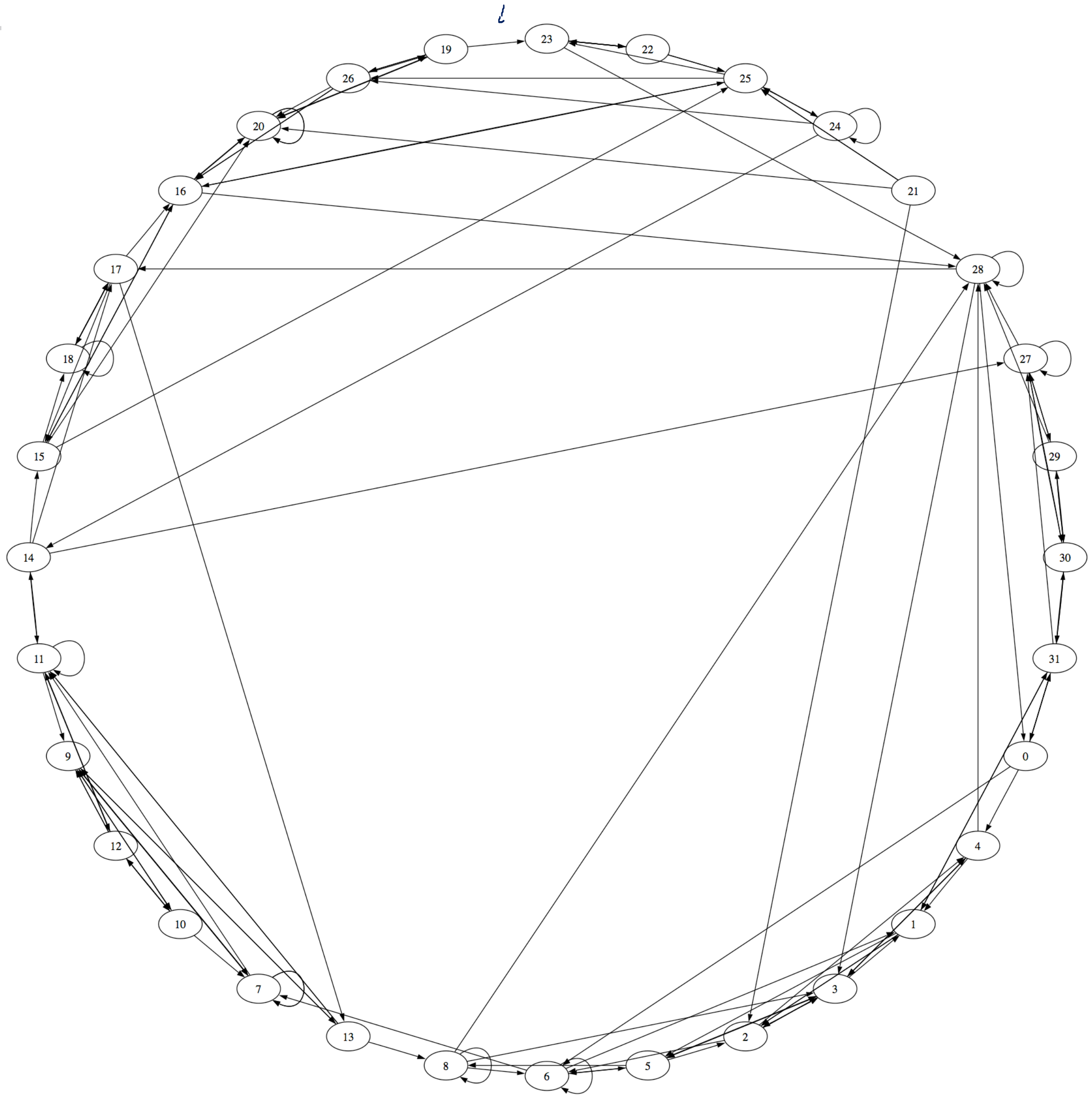


1 Iteration Pull ...

= n Pull-opcal,

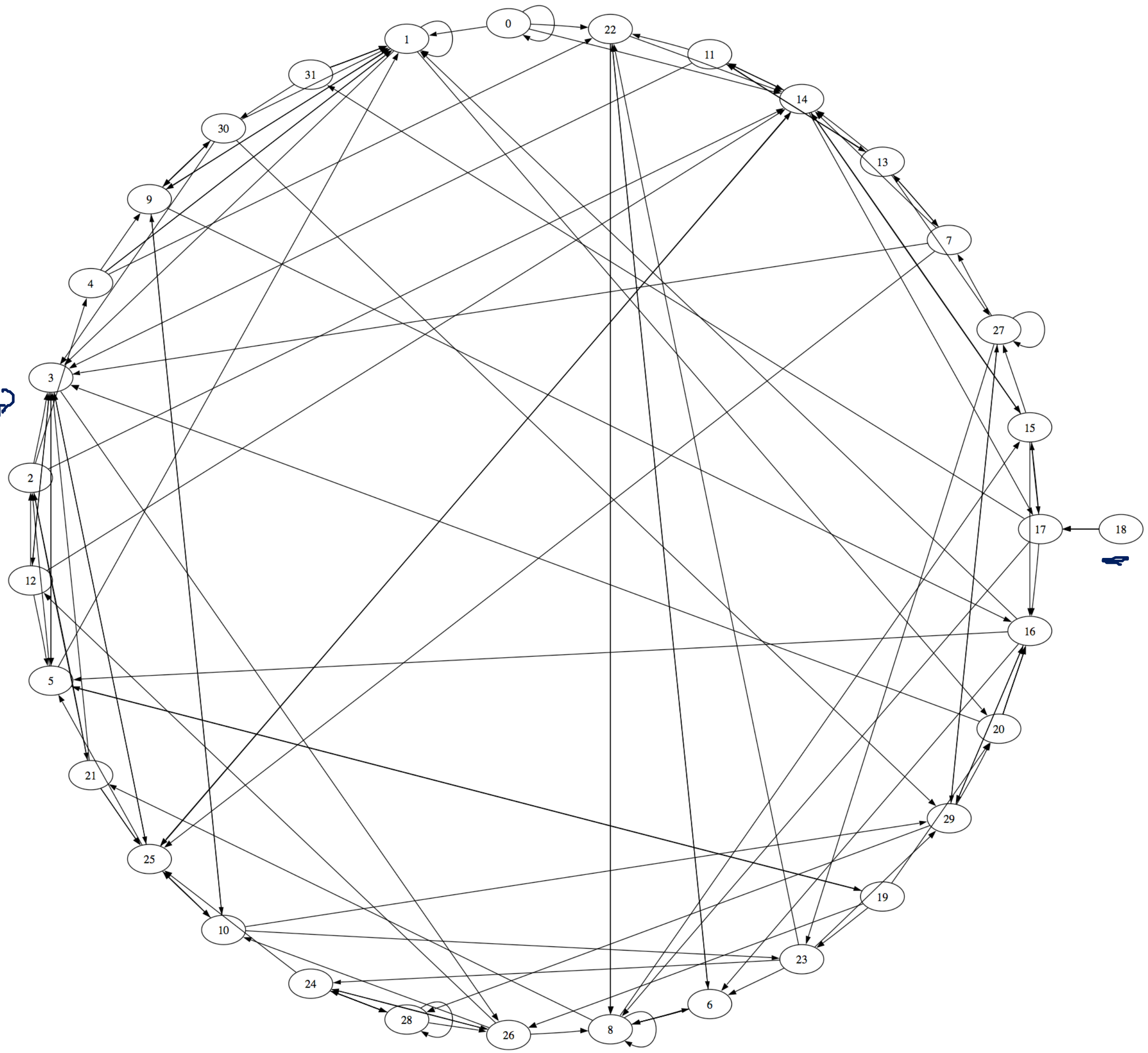


10 Iterations Pull ...

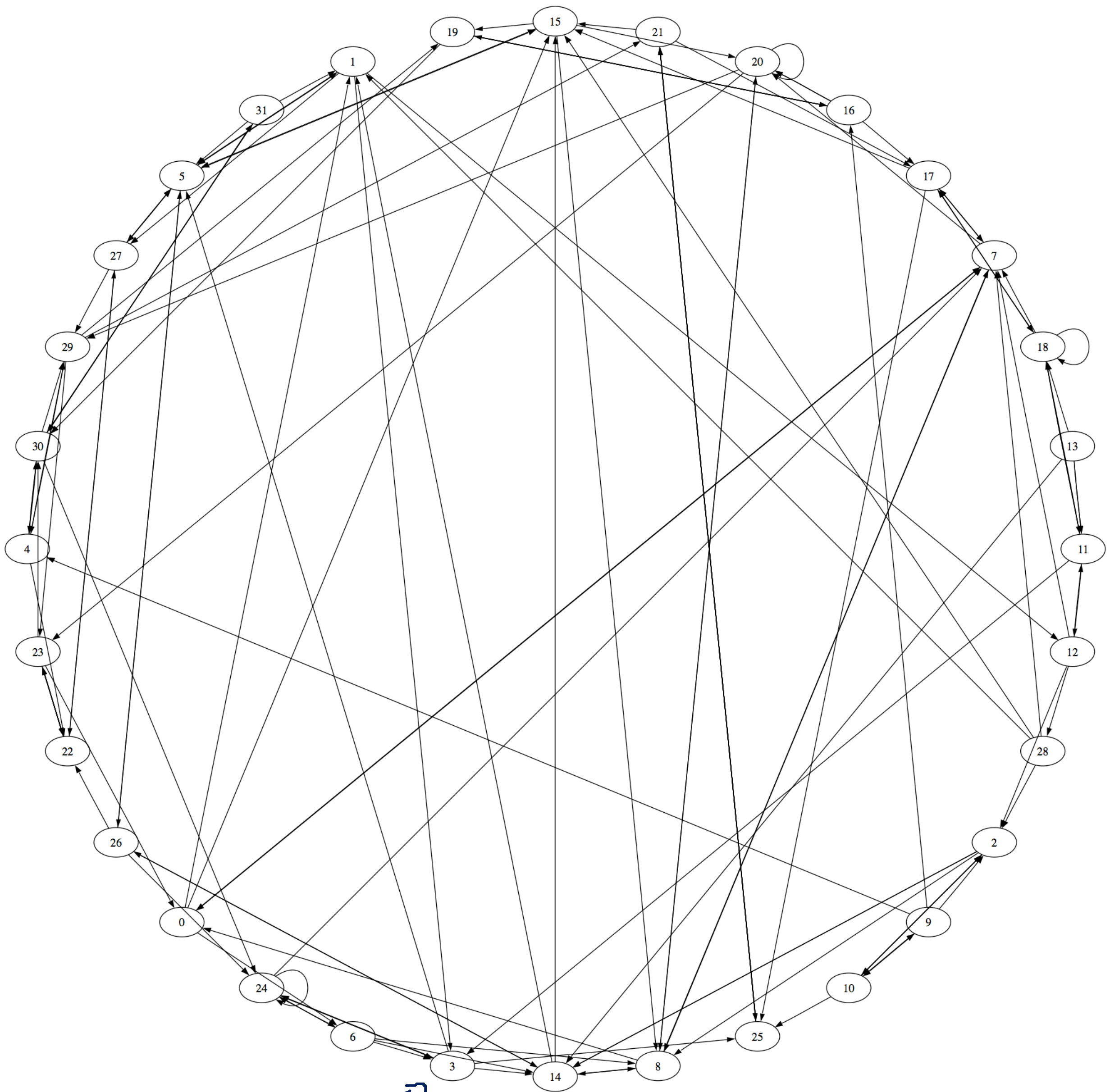


20 Iterations Pull ...

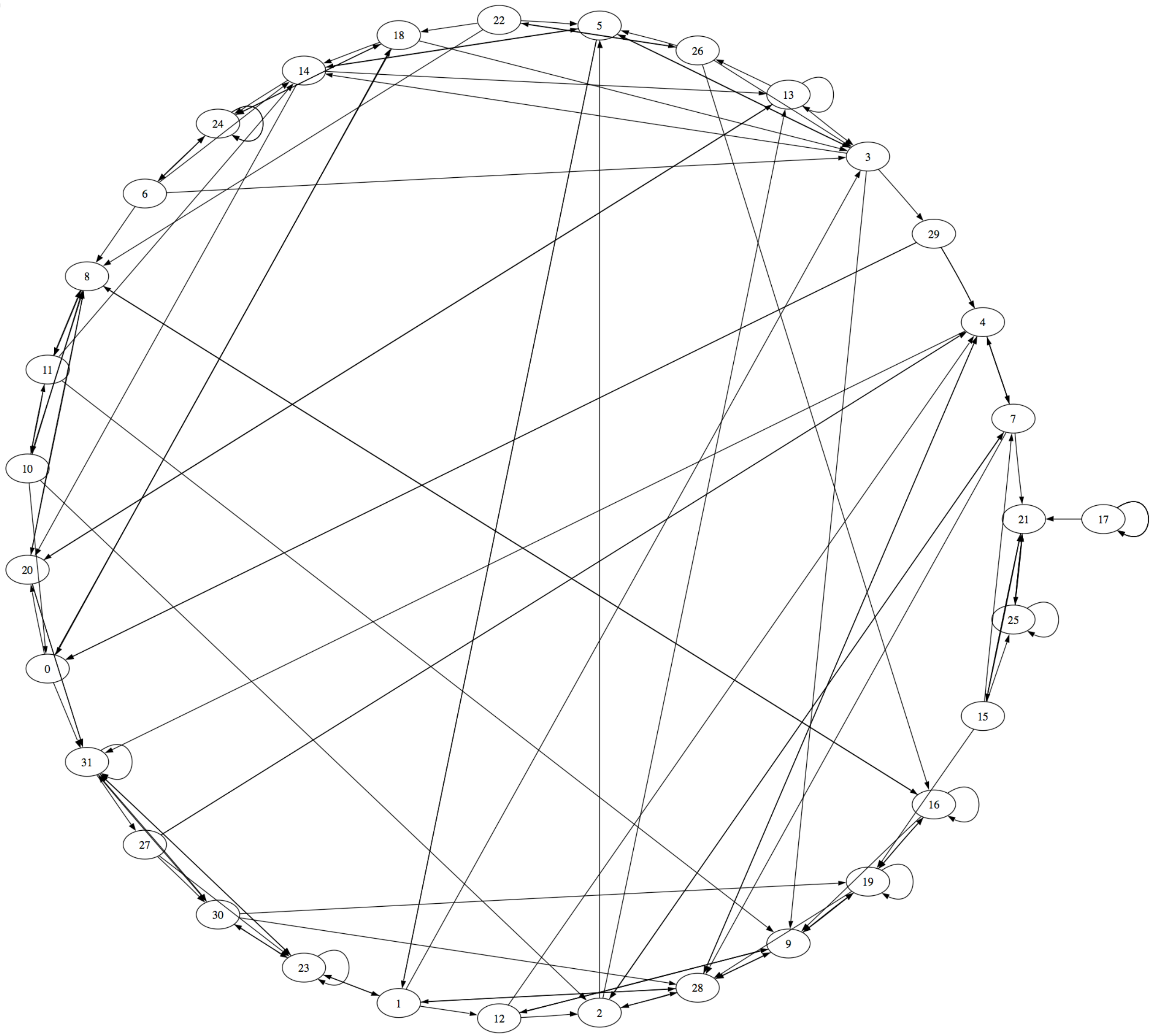
20 m Pulls



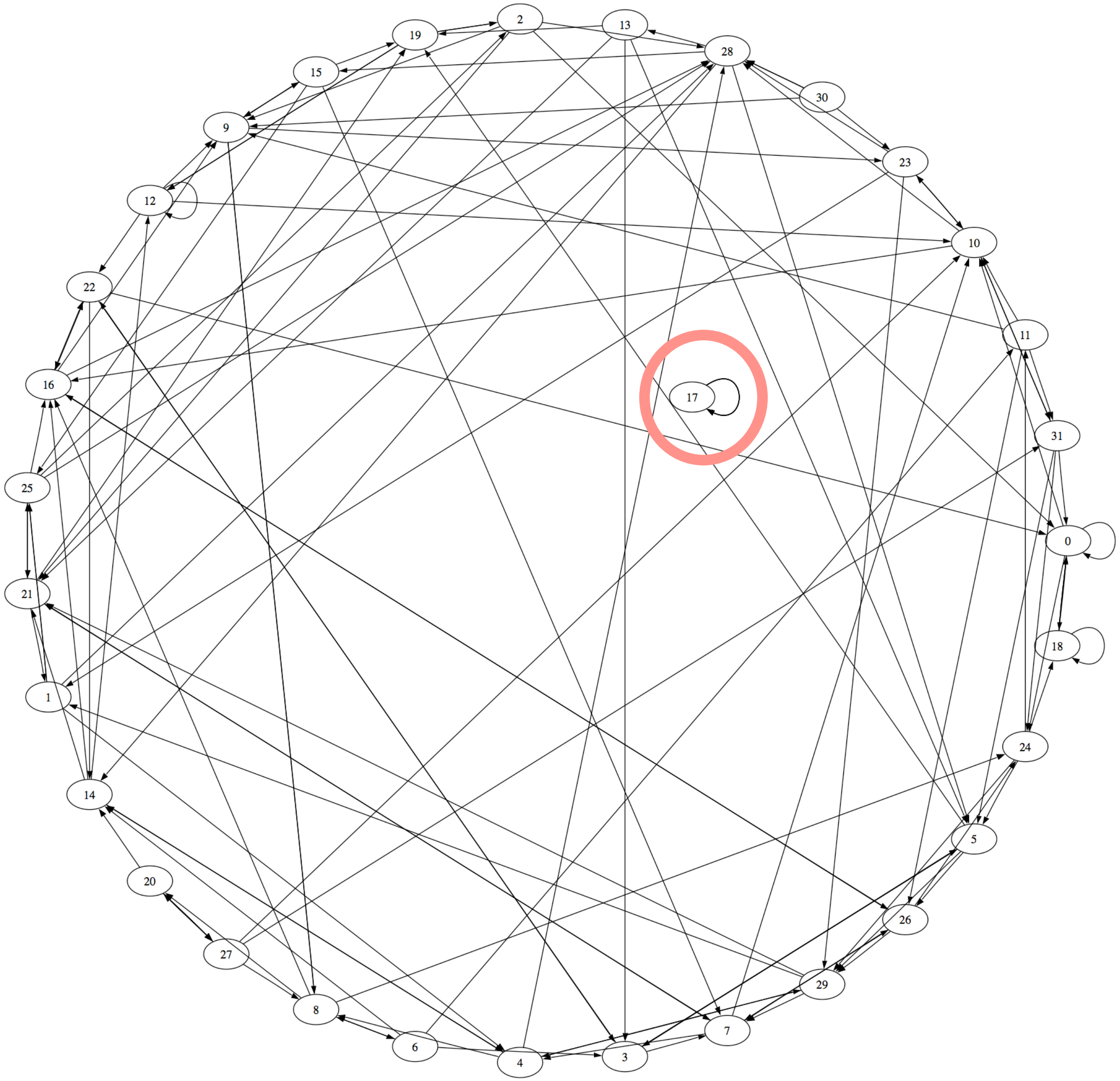
30 Iterations Pull ...



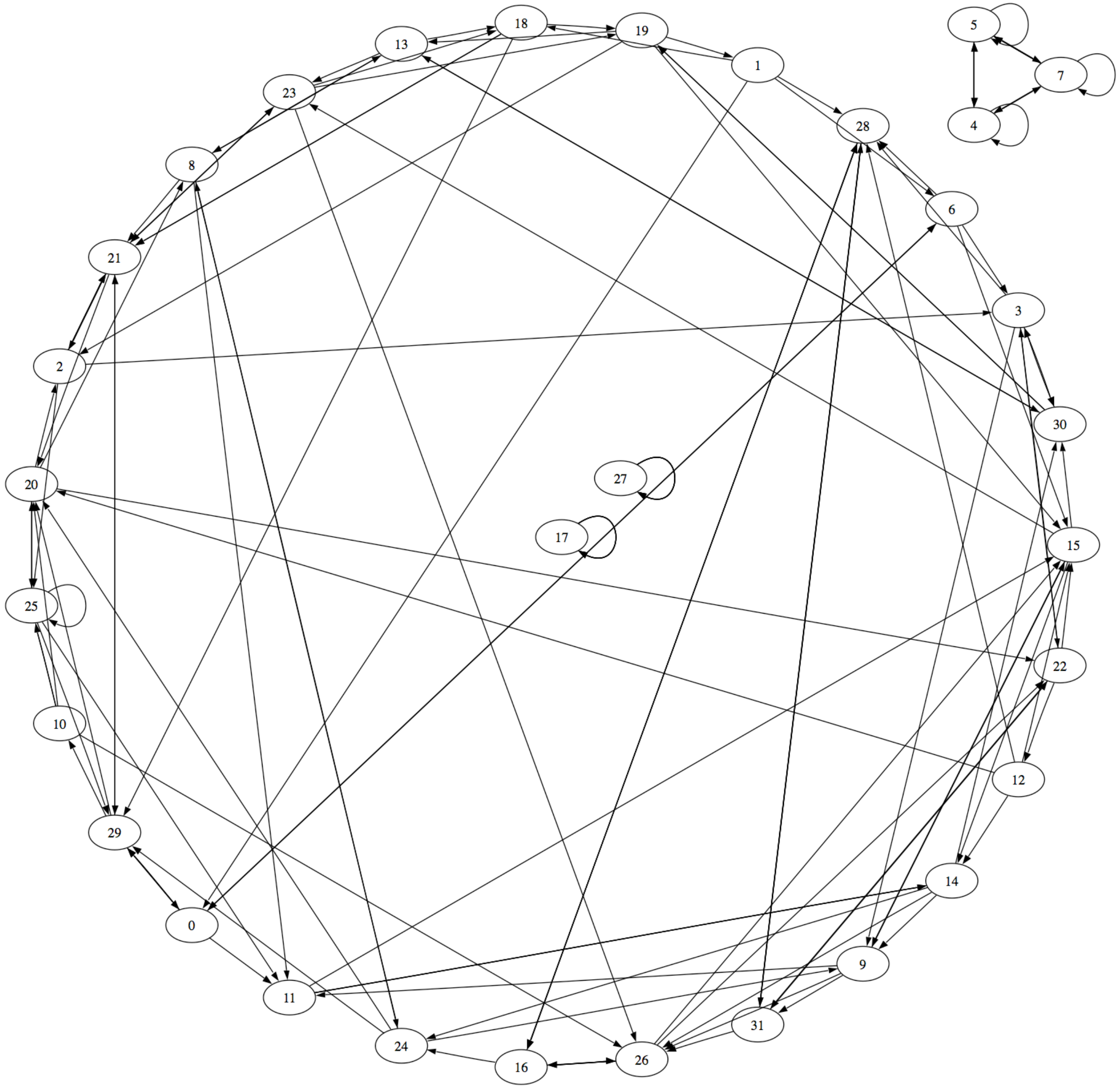
40 Iterationen Pull ...



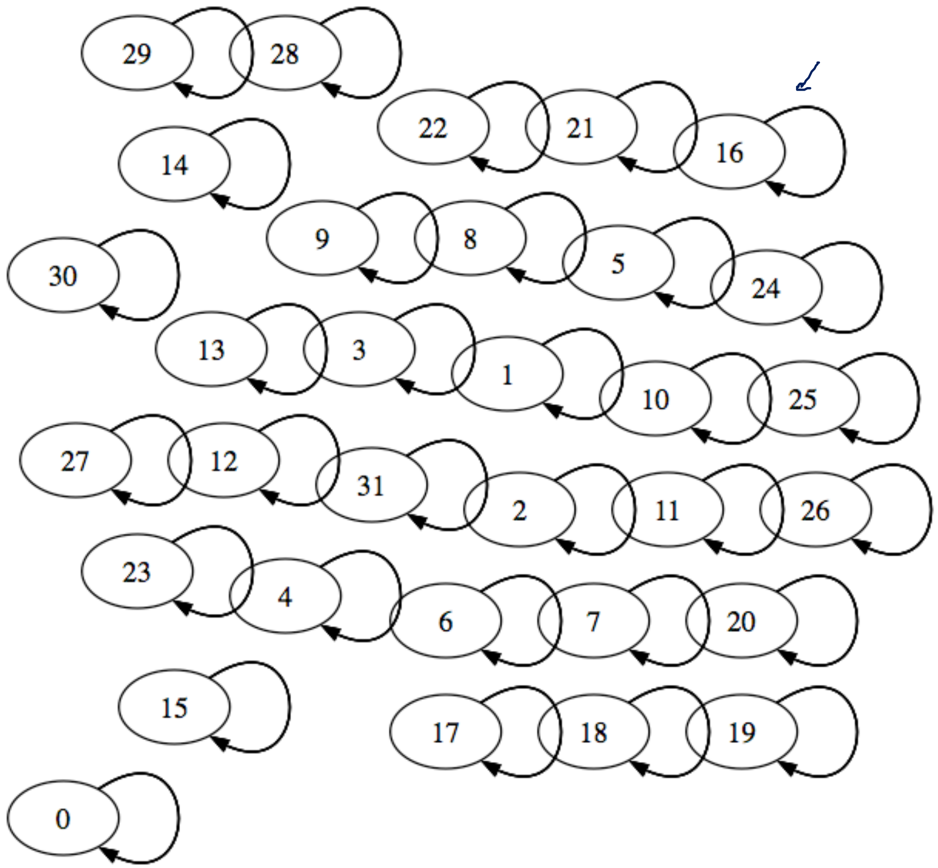
50 Iterations Pull ...



500 Iterations Pull ...



5000 Iterations Pull ...



Combination of Push and Pull