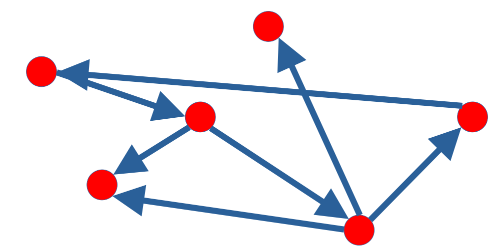
Peer sampling service

- Produces a random sample of peers for dissemination / aggregation
- Naive approach: Uniform sample from complete list of peers
- “Node churn”
  – Costly to store and update (monitoring)
  – Wasteful, as the same peers should be reused for multiple iterations (network connections, diagnostics, …)

Overlay networks

- The paths of epidemic dissemination implicitly define a random graph overlayed on the physical network

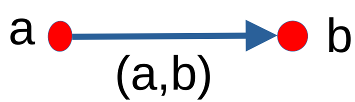


- Peer sampling is equivalent to creating a <u>random overlay network</u>
- Do it by incrementally and with local information by attaching new nodes to an existing network
- Terminology: Peer sample == neighborhood == view

- Desirable graph properties?

Graphs

- <u>Nodes</u> and <u>edges</u>: G = (V,E)

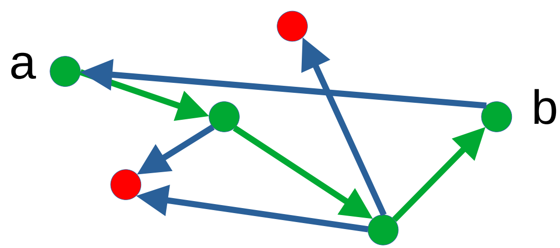


- <u>Directed</u> vs undirected:

- Relevance for epidemic dissemination: local knowledge

Connectivity

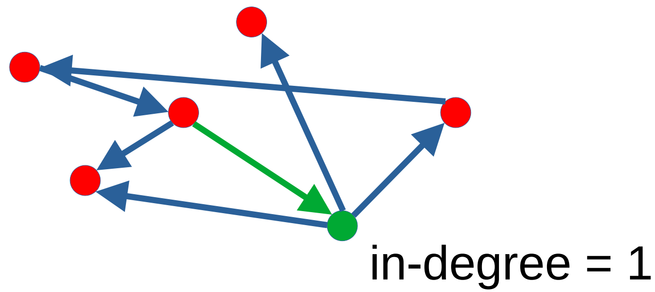
- Path:

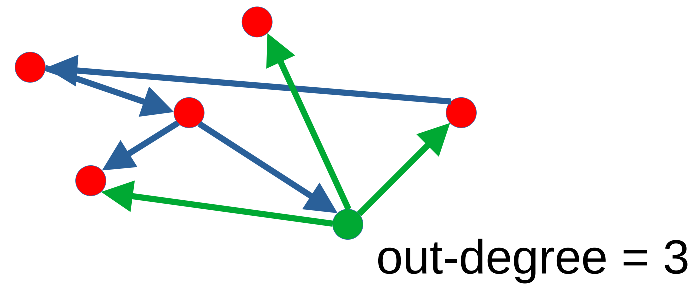


- Strongly connected if there is a path from any node *a* to any other node *b*
- Relevance for epidemic dissemination: Atomic delivery

Degree

- In-degree and out-degree:





- Measure: <u>degree distribution</u>
- Relevance for epidemic dissemination:
  - Load balancing
  - Reliability (isolated nodes)

Expansion

- Minimum number of edges across all possible partitions of nodes in two sets:

- Hard to measure...
- Relevance: Reliability (isolated components)

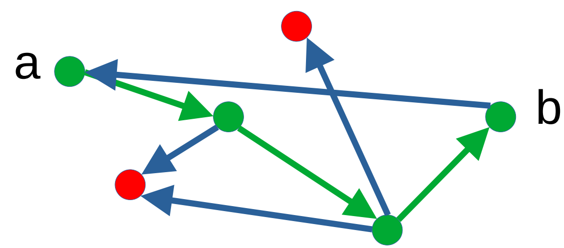
Clustering coefficient

- Proportion of edges amont neighbors

- Measure: average clustering coefficient
- Relevance: Reliability (good proxy for expansion)

Distance

- Number of edges in shortest path between two nodes



distance(a,b) = 3

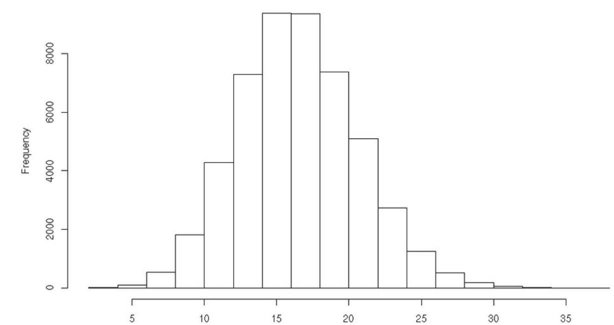
- Measures: diameter (largest distance) and average path length
- Relevance:
  - Delivery latency

Uncertainty and faults

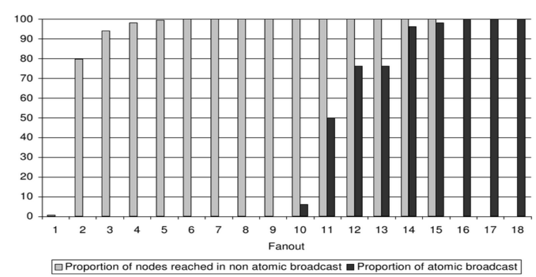
- Each node holds a local belief about the graph
  - After node failures, edges to non-existing nodes
  - Accuracy is the ratio of edges to existing nodes
- Impossibility of agreement when updating local knowledge:
  - There are no undirected graphs
  - Symmetry still desirable

Random graph (Erdos-Renyi)

- An Erdos-Renyi random graph *G(n,p)* has:
  - *n* nodes
  - each edge exists with probability *p* (i.e. *n(n-1)p* edges)
- Degree distribution:

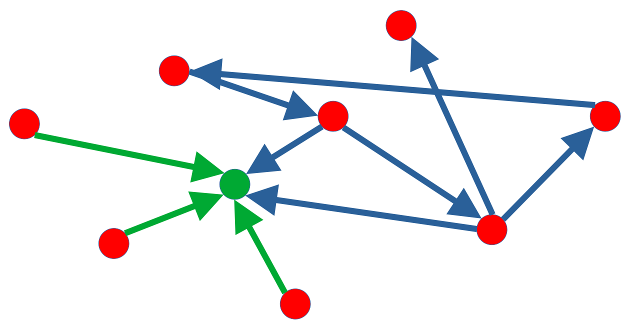


- Low clustering coefficient
- Average path length: O(log n)
- Connectivity:



Naive approach

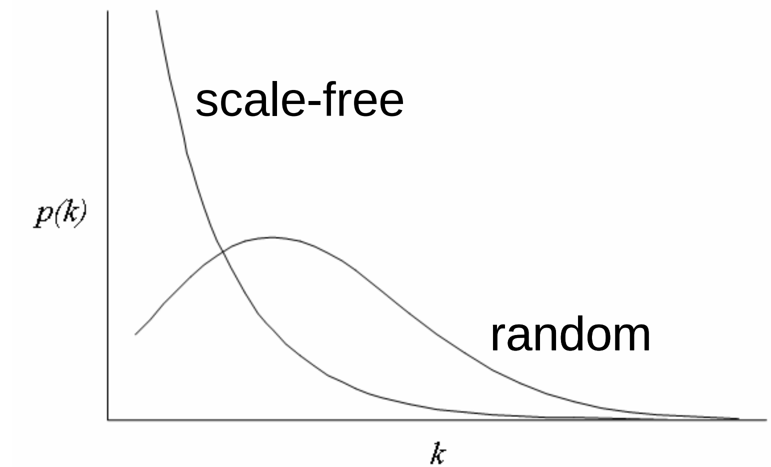
- How to connect to the network? Ask someone for help: connect to some node, then to its neighbors...



- Probablity of picking a node ~ in-degree
- This is called “preferential attachment” (a.k.a. “the rich get richer”)

Scale-free network

- Skewed degree distribution
  - Excessive load in some nodes
  - Other nodes can easily become disconnected



- High clustering coefficient
  - Likely to created disconnected components
- Average path length is good (i.e. at most log(n)), at the expense of some nodes

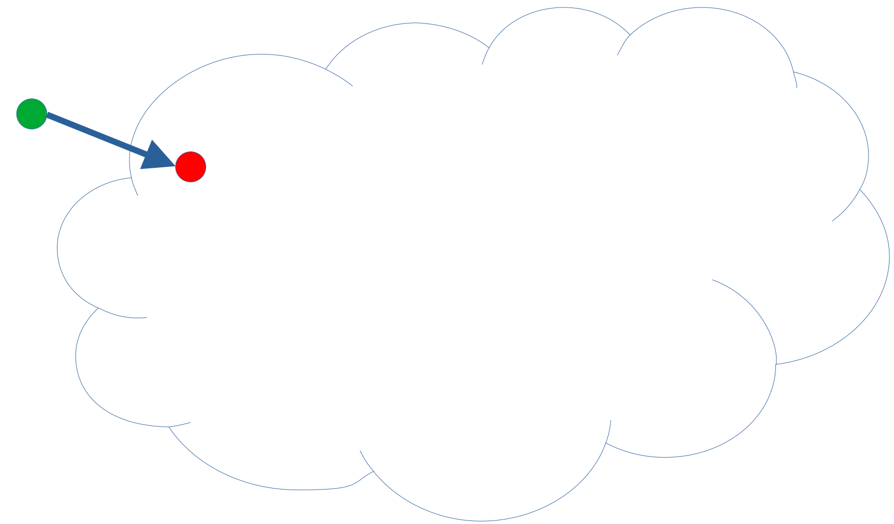
Random walk

- Select an entry node
- Choose an out-edge at random
- Repeat *t* ~ *log(n)* times
- Select the final node

- Indistinguishable from uniform random sampling from *n* nodes

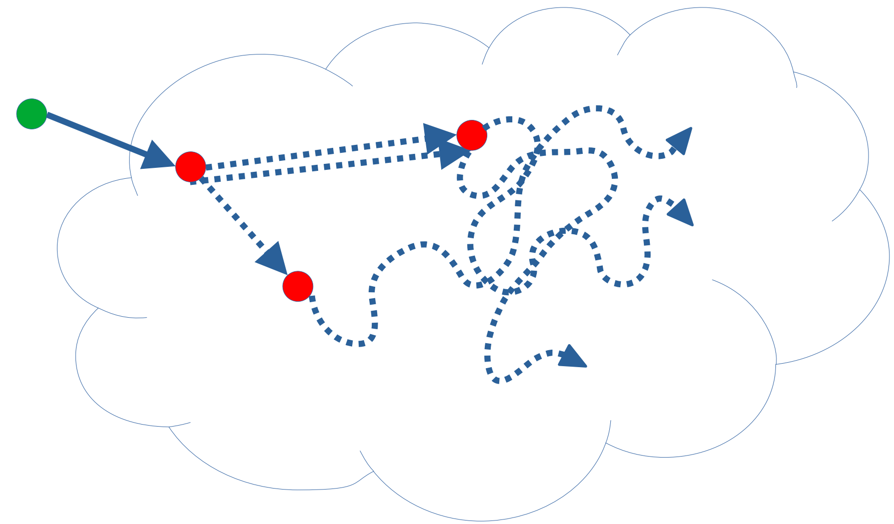
SCAMP

- Send subscription to an arbitary contact node
  - Not necessarily random!



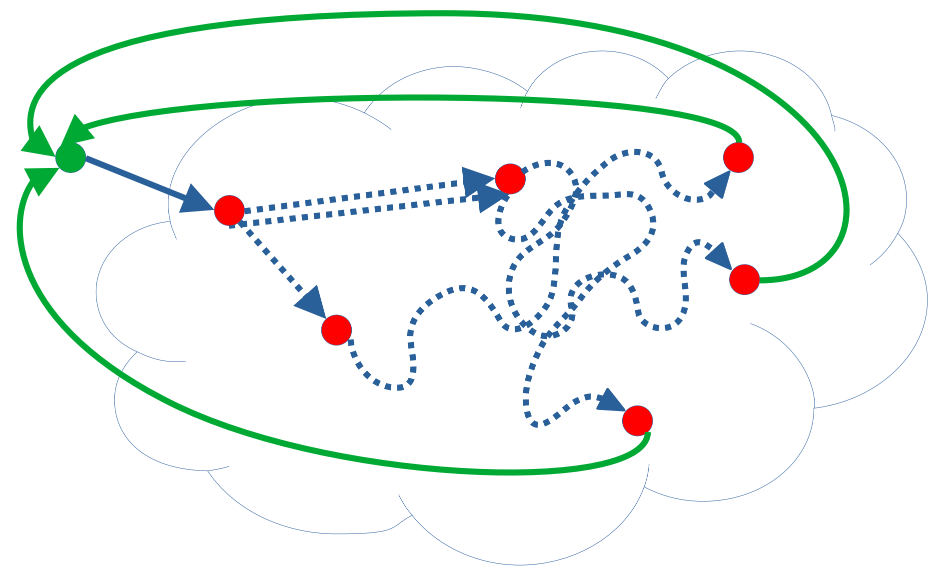
SCAMP

- Contact node initiates random walks to (see 1):
  - All out-edges (~ log(n)!)
  - Additional *c* to random out-edges
- *c* is a parameter needed for:
  - tolerating faults
  - selecting a contact in the lower end of the degree distribution



SCAMP

- Stop random walk with *p ~ 1/out-degree* (see 2)
- Add edges to the new node



- Notes:
  - (1) balances the in-degree of new nodes
  - (2) balances the out-degree of existing nodes

SCAMP

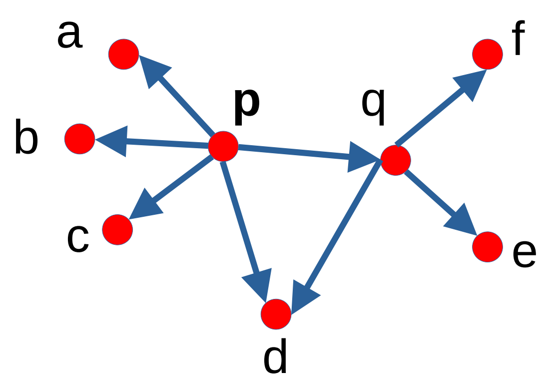
- Approximates Erdos-Renyi random graph (as the network grows)
- What if the network is shrinking?
  - Both in-degree and out-degree become unbalanced (higher variability)
  - No mechanism to maintain accuracy (monitoring)
- <u>Reactive strategy</u>: Network changes only on explicit request

Shuffling

- Basic idea: Periodically, pairs of nodes combine and then split local views

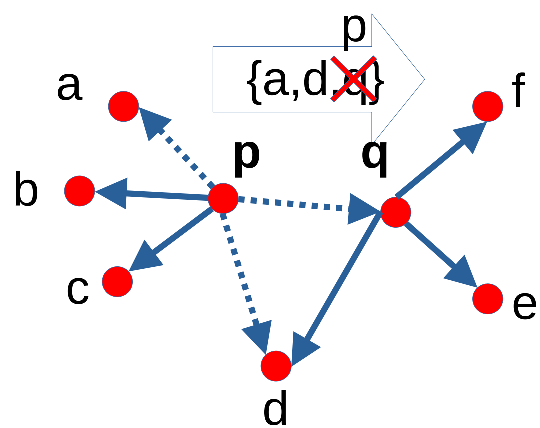
Basic shuffling

- Node *p* (initiator) has view {a,b,c,d,q} (up to *c* nodes)
- Selects subset {a,d,q} (up to *l* nodes)



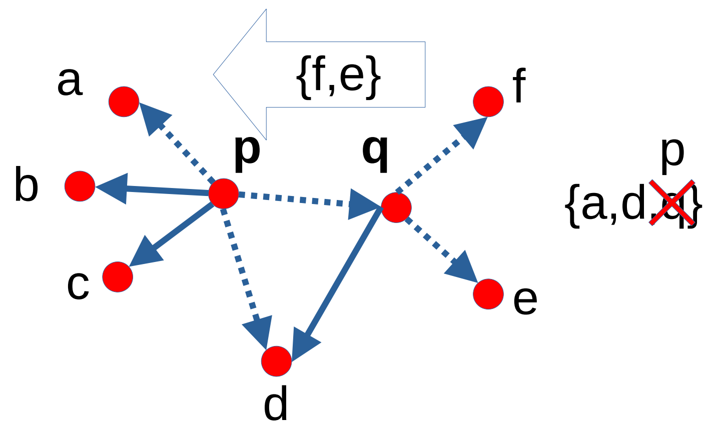
Basic shuffling

- Selects 1 node as the target from subset: q
- Replaces it with its own and sends it: {a,d,p}



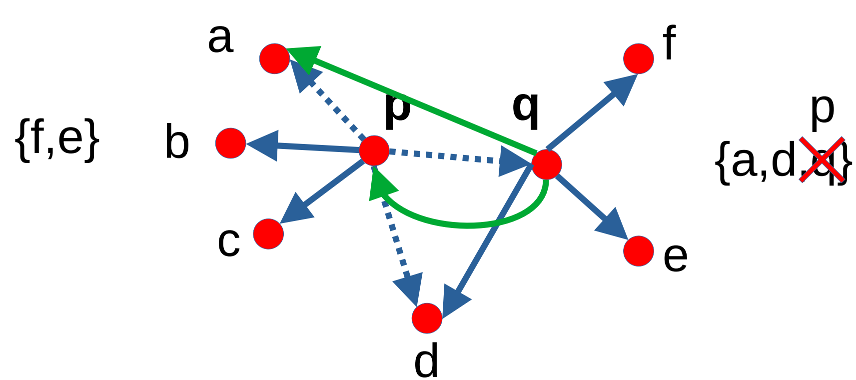
Basic shuffling

- Target *q* also selects a random subset and returns it: {f,e}



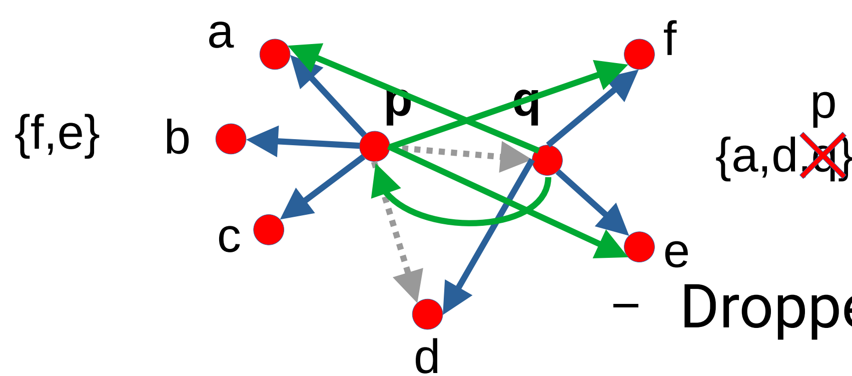
Basic shuffling

- Each node adds merges received subset:
  - Discarding duplicates and self references
  - Discarding nodes sent if not enough space



Basic shuffling

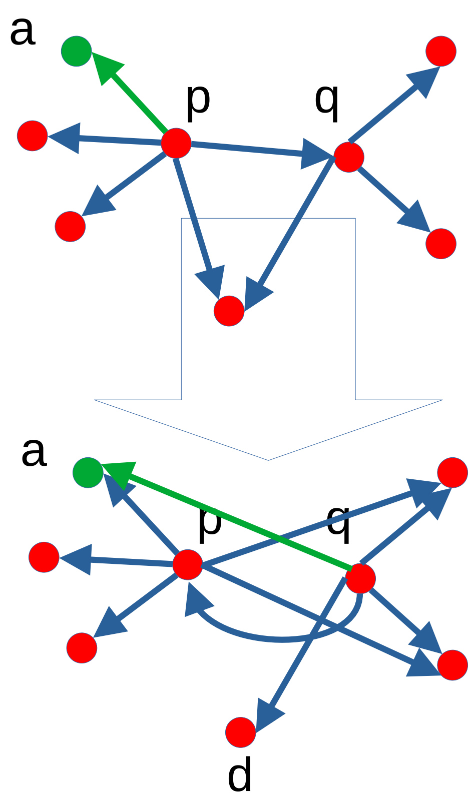
- Balancing of out-degrees:
  - Node *p* keeps {a,b,c,e,f}
  - Node *q* keeps {a,d,e,f,p}
- What about in-degrees
  - Nodes with high in-degrees chosen for often as targets



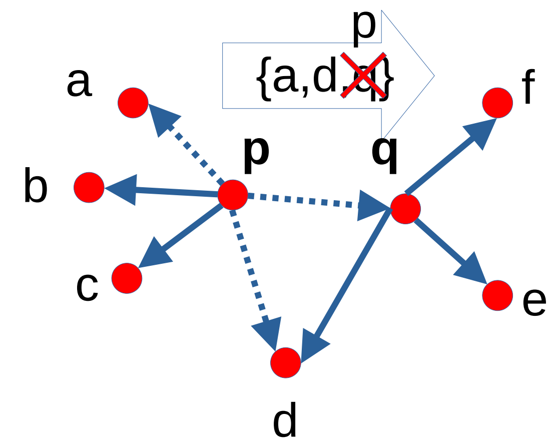
  - Dropped from views
  - In-degree decreases
  - And...

Understanding shuffling

- For each node exchanged:
  - “step in a r.w.”



- For each shuffle initiated:
  - “r.w. started”



- Each cycle ~= a new r.w. + a batch of r.w. steps
  - Balances in-degree

Basic shuffling

- Cyclic strategy: View changes periodically

- What about accuracy?
- A node that fails stops “initiating new random walks”
- There is a chance of being selected as target and discovered dead:
  - Slowly fades away from views

- Can we make it faster?

Enhanced shuffling (CYCLON)

- Tag each edge with age:
  - 0 when *p* adds itself to shuffle subset
  - Increment all each shuffling period
- Select oldest as *q* (remember: *q* is going to be discarded by *p*!)
- In each cycle:
  - Each live *p* node creates a new reference to itself
  - Somewhere in the network, some reference to *p* is the oldest and is discarded

More...

- Hybrid strategy (HyParView):
  - Reactive strategy to maintain a small symmetric active view
  - Cyclic strategy to maintain a large passive view
- Byzantine fault tolerance (Brahms):
  - Malicious nodes: Sybills, eclipse, …
  - Random sampling from a biased stream

References

- A. J. Ganesh, A.-M. Kermarrec, and L. Massoulié, "**SCAMP: Peer-to-Peer Lightweight Membership Service for Large-Scale Group Communication**," in Proceedings of the Third International COST264 Workshop on Networked Group Communication, Nov. 2001
  https://dl.acm.org/doi/10.5555/648089.747488
- S. Voulgaris, D. Gavidia, and M. van Steen, "**CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays**," Journal of Network and Systems Management, vol. 13, no. 2, pp. 197–217, Jun. 2005
  https://doi.org/10.1007/s10922-005-4441-x