Taking Graph Metrics a Step Further: MOPS

- P2P Systems master *primary* problems (connectivity, location) well

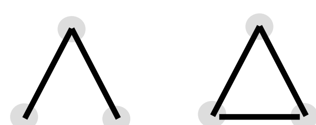
- *But:* raise a plethora of secondary problems (response times, recall, precision, availability, *load distribution*,…)

- Traditional solutions:
  1. Build a sophisticated monitoring
  2. Exchange lots of information to establish idea of global state
  3. Introduce protocol to optimize (mitigate…)

- How about:
- *Analyze subgraph of neighbors and optimize locally?*

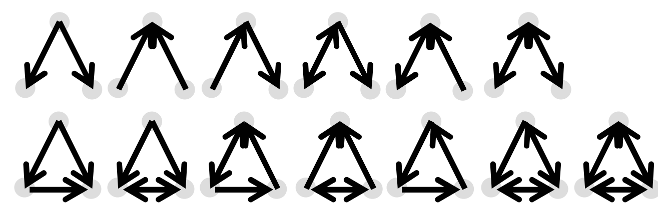
Motifs: Local Structures of Networks

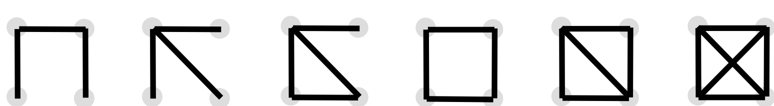
- Global metrics for network analyzis are expensive (and tedious to calculate)
- Local structures are surprisingly characteristic for complex networks
- Analyze direct neighborhood:
  - Permutation of all pairs (triads) / triples (tetrades) / … of neighbors
  - Check interconnectivity (Motif)
  - Count occurrence of characteristic subgraphs
- Calculate statistical significance of occurences



(2)

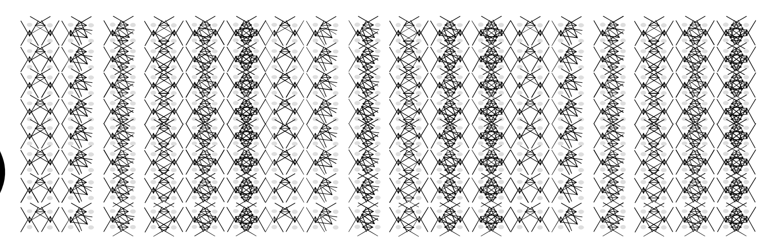
(13)



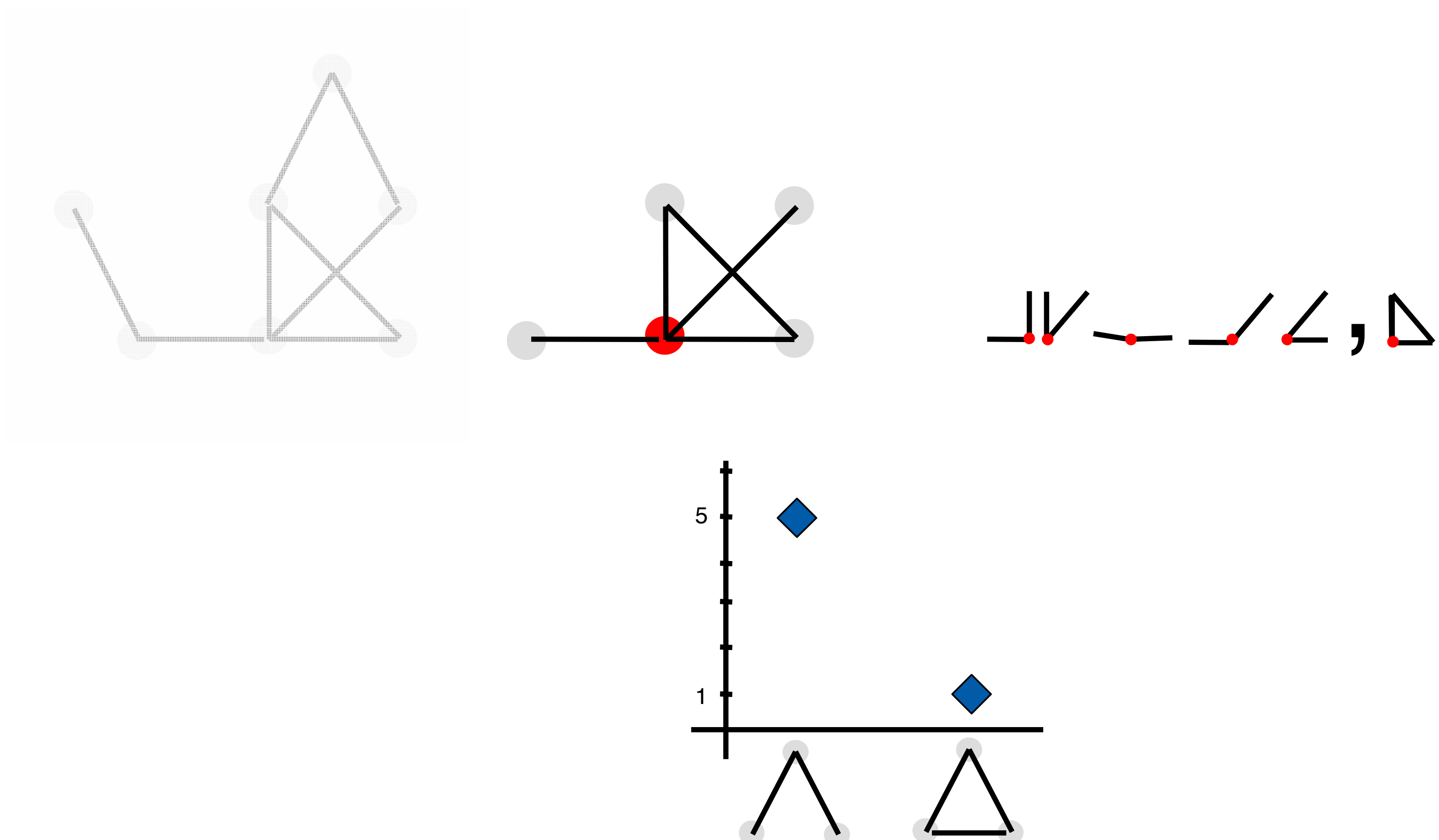


(6)

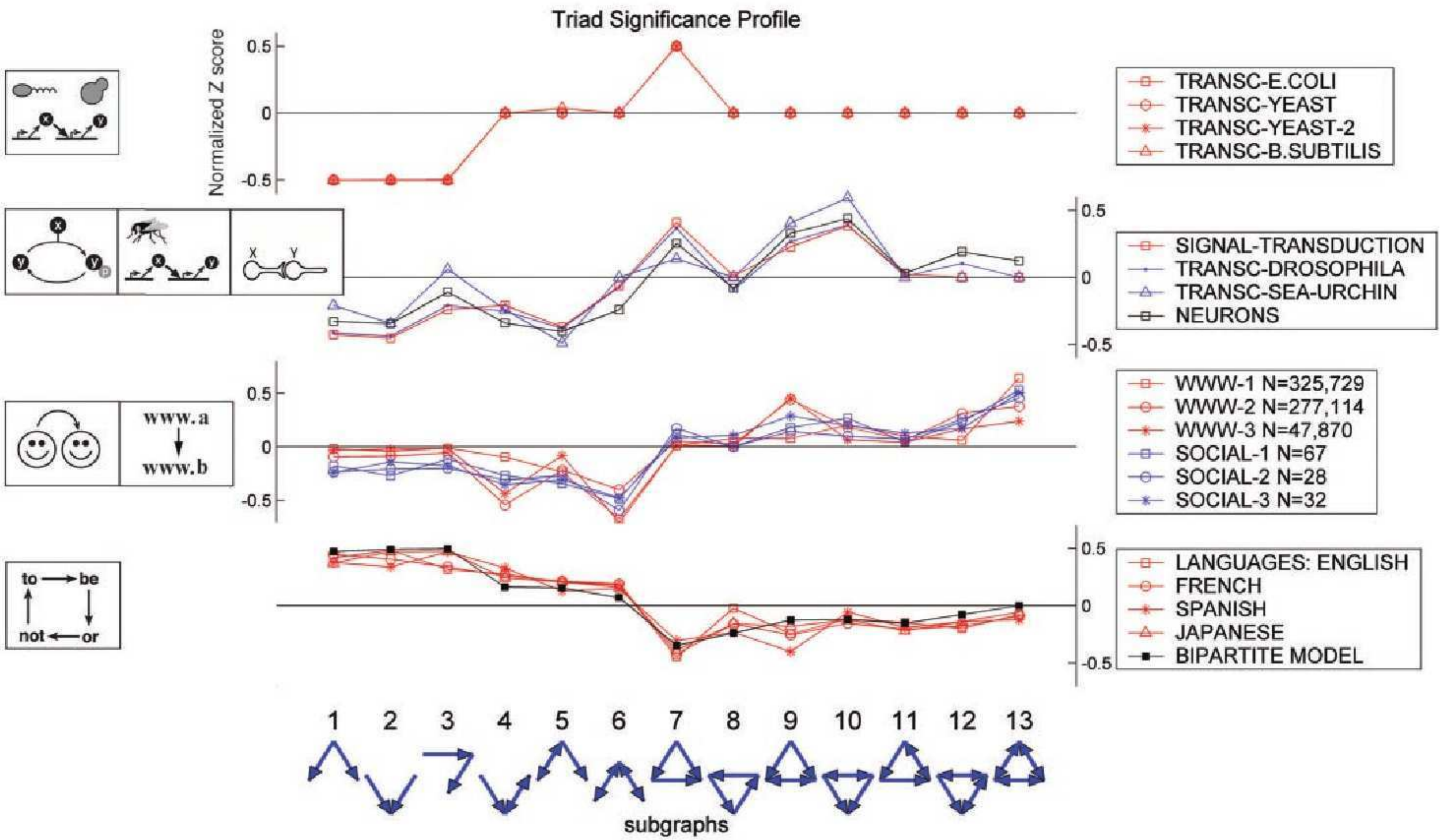
(192)



A Quick Example of Motifs



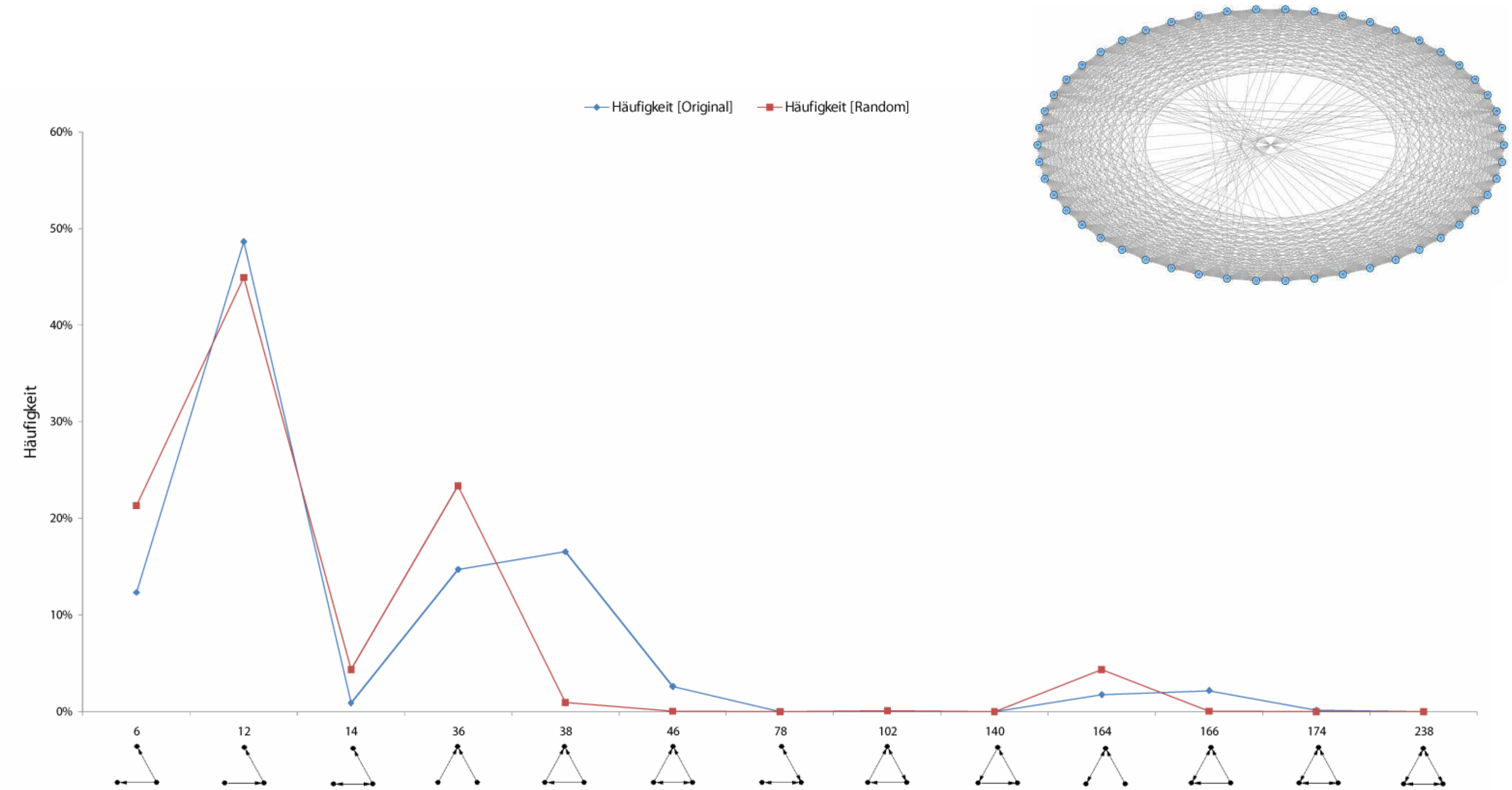
Network Characteristics and Significance Profiles



[Milo et al. '04]

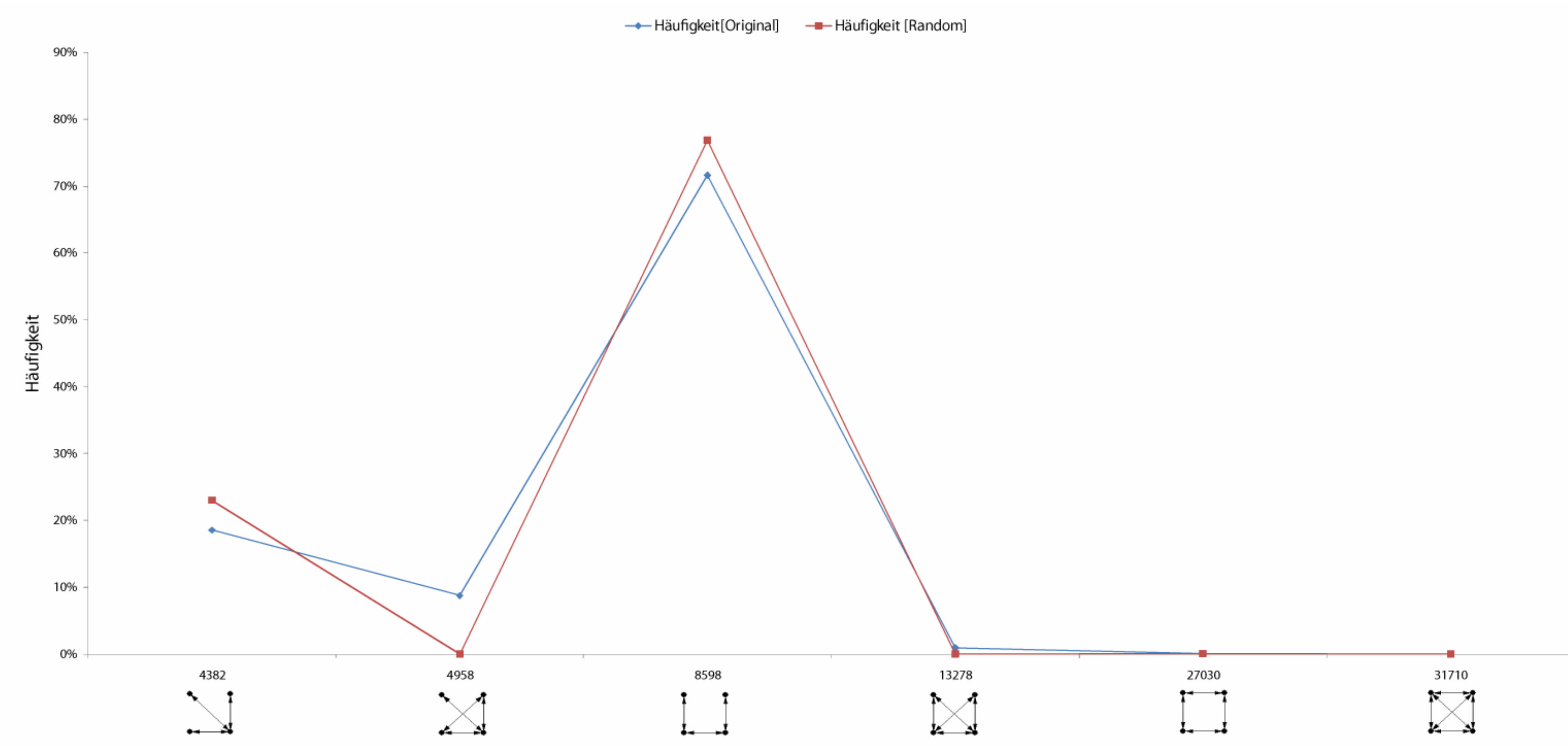
Motifs in Chord

- Ring structure, finger tables



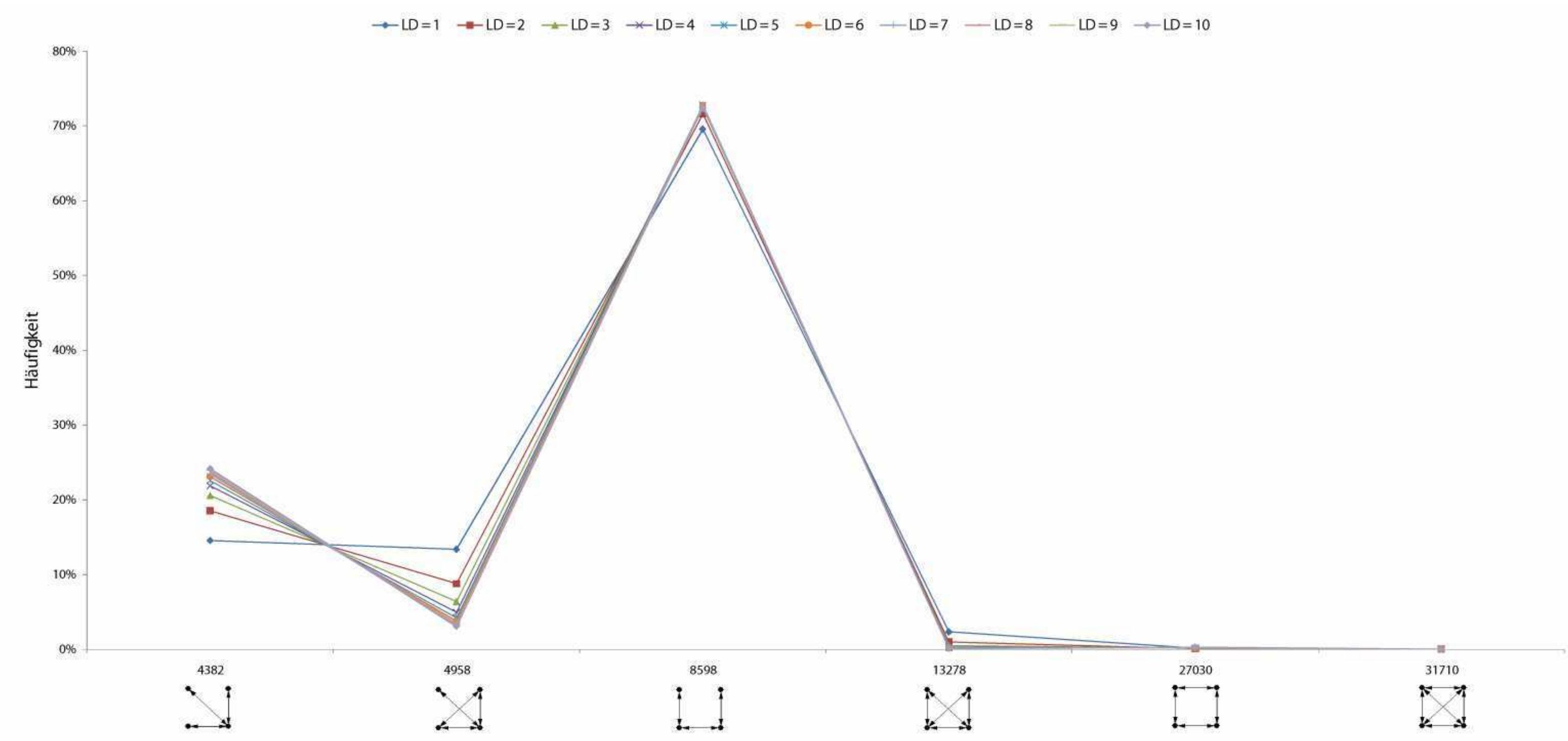
Quick Example DHT: Symphony

- Ring based, random long distance links (LD)
- (Watch out! It‘s frequency, not significance in these cases!)



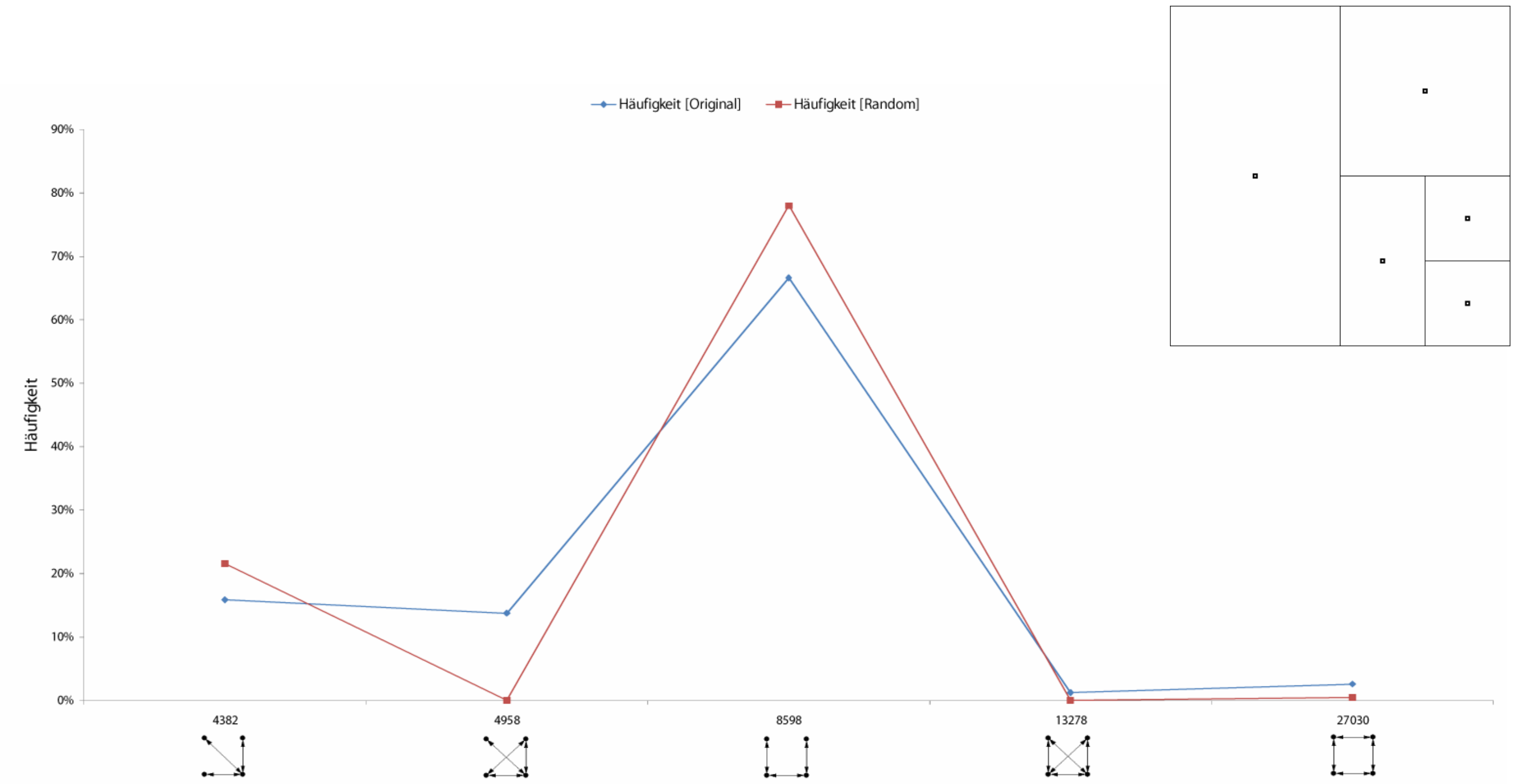
Increasing Long-Distance Links in Symphony

- Lower clustering, converge to random



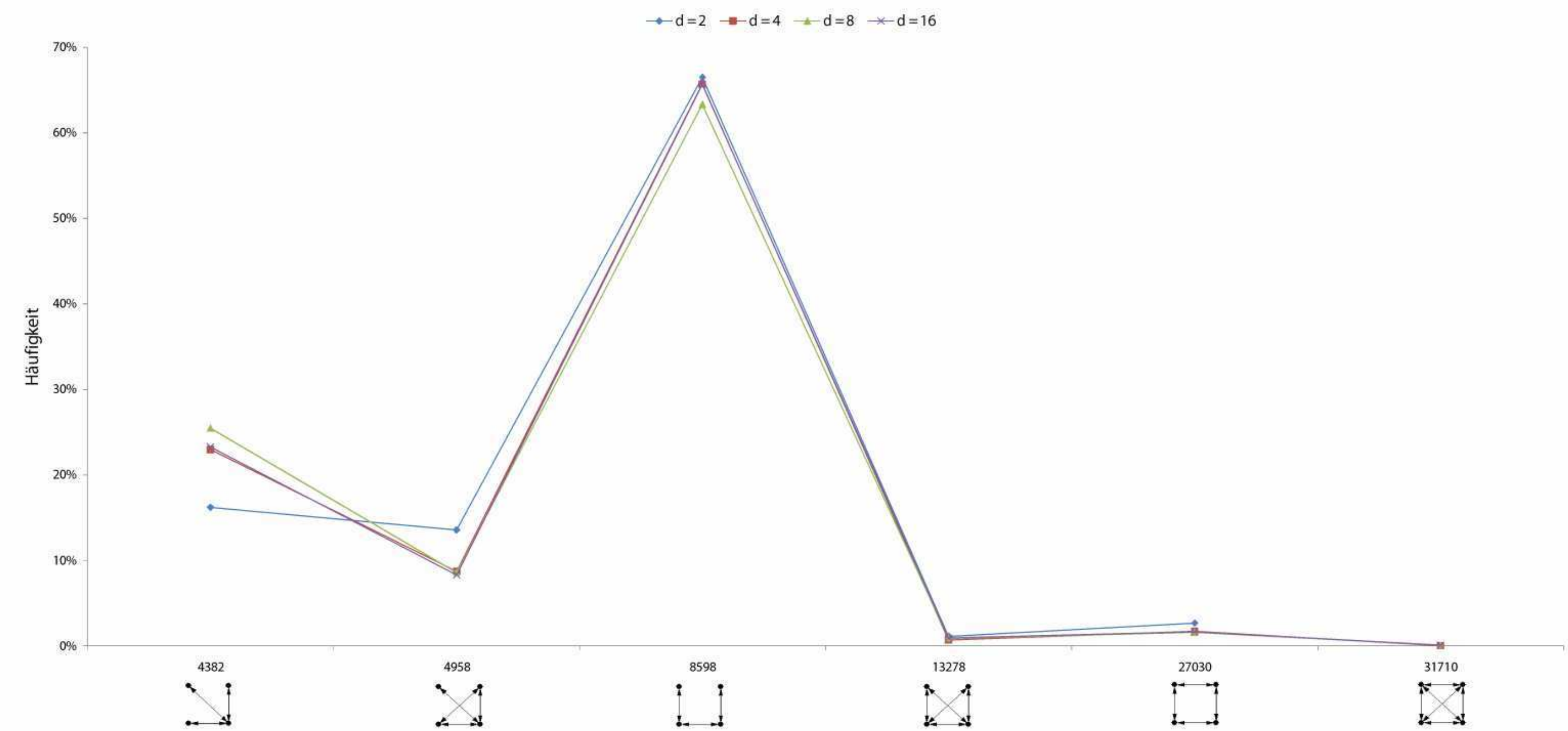
Another Example: CAN

- Key-space d dimensional Torus



Increasing Dimensions of CAN

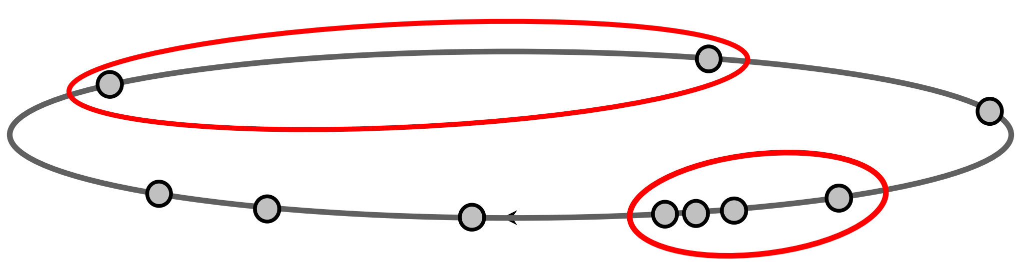
- Less clustering, only slight changes

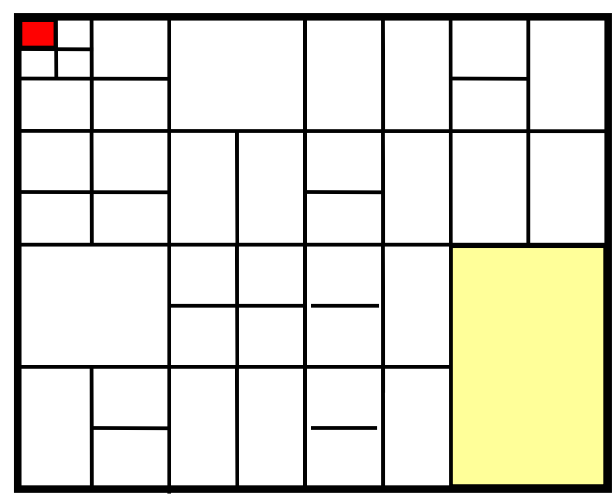


Common (secondary) Problems with P2P

- Properties of DHT:
  - Random assignment (ID/coordinate) → unequal assignment of key space
  - Heterogeneity of nodes, popularity of content
  - Dynamic arrival and departure of nodes (churn)…

- Secondary problems:
  - response times, recall, precision, availability, *load distribution, …*





Motif-based Optimization of P2P-Substrates

- Each node knows its neighborhood (Motif-signature)
- Objectives:
  - **Optimize overall overlay by optimizing neighborhood!**
  - **One system for all P2P substrates (not derivatives of all DHT…)**
- **Implement plugin for all DHT:**
  - Intercept join messages
  - Check local state
  - stick to protocol (do nothing) ***OR***: divert from protocol
    - Terminate join early (assigning different ID space)
    - Drop message
    - Trigger re-join from scratch (target area is optimal, max a times)
- **Manually construct an overlay topology „optimal" to some metric**
- **Calculate characteristic motif significance profile (average of motif signatures)**
- **Derive locally desired motif signatures**
- **Feed target motif signature(s) into plugin**

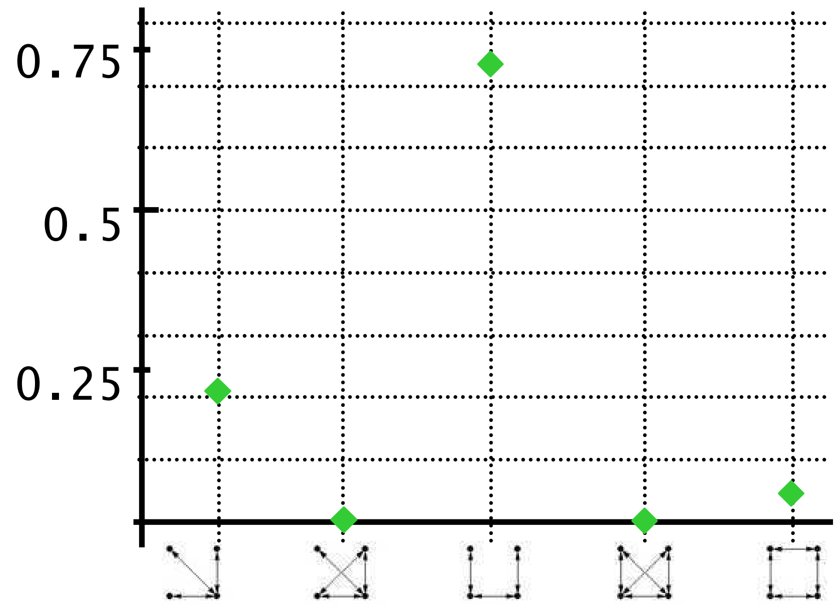
Optimizing a CAN…

- Assumption:
  - **„CAN is optimal if all nodes are assigned scope of equal size (*or *0.5, *2*)“**

- Directed tetrade motif signature is characteristic for all nodes

- Parametrize Plugin:
- Target Motif signature

- Plugin-Strategy:
- Terminate join early
- Trigger re-join (max a times)



Preliminary Evaluation: Questions asked

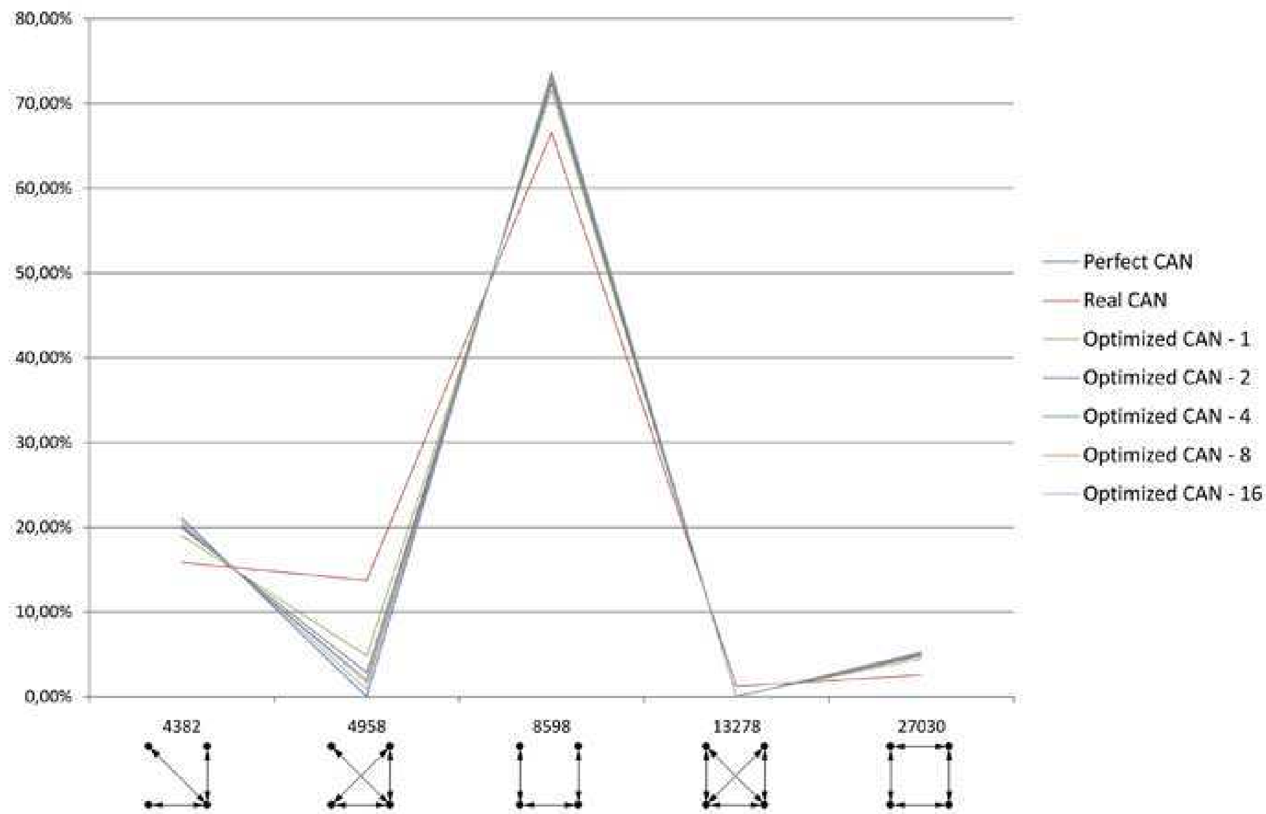
Implemented as plugin for existing simulation framework (planet sim)

- Simulation study to check:
  1. Does the Motif-based optimization converge to globally desired Motif significance profile?
  2. How good is the overlay after optimization?

- Simulation study:
  - Plain simulation of overlay
  - CAN vs. Optimized CAN
  - Variation of group sizes: $2^{10} – 2^{16}$ (How well do we balance?)

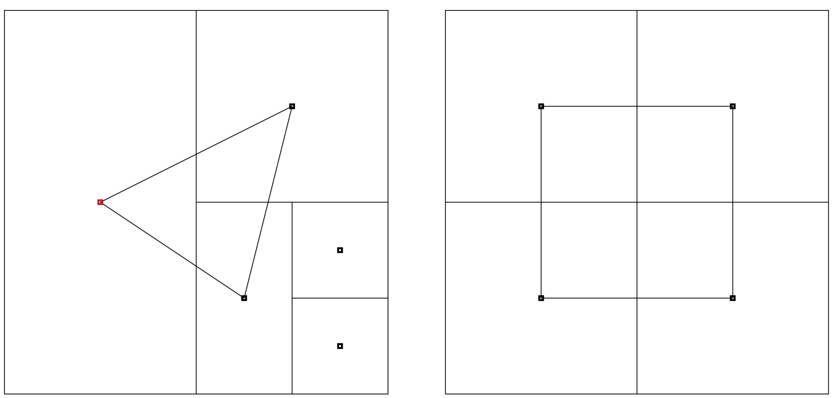
Which Global Impact has Local Optimization?

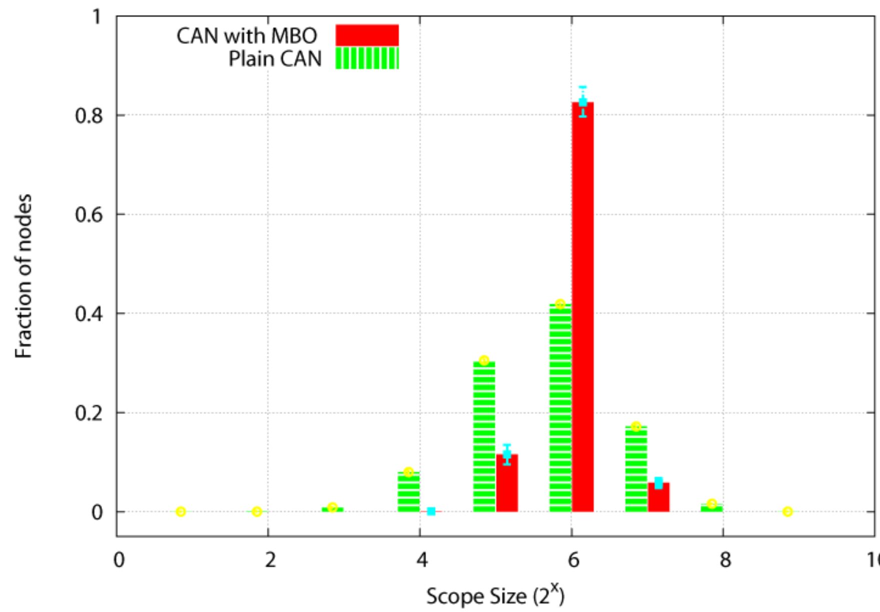
- **„Does Motif-based optimization converge to global desired Motif significance profile?“**



Do we enhance the Load-Balancing of CAN?

- Topological changes lead to better balanced allocation / incoming requests
- Check direct environment and optimize to local balance
  - Motif-based is one option





Does it Work With Other P2P Systems (KAD)?

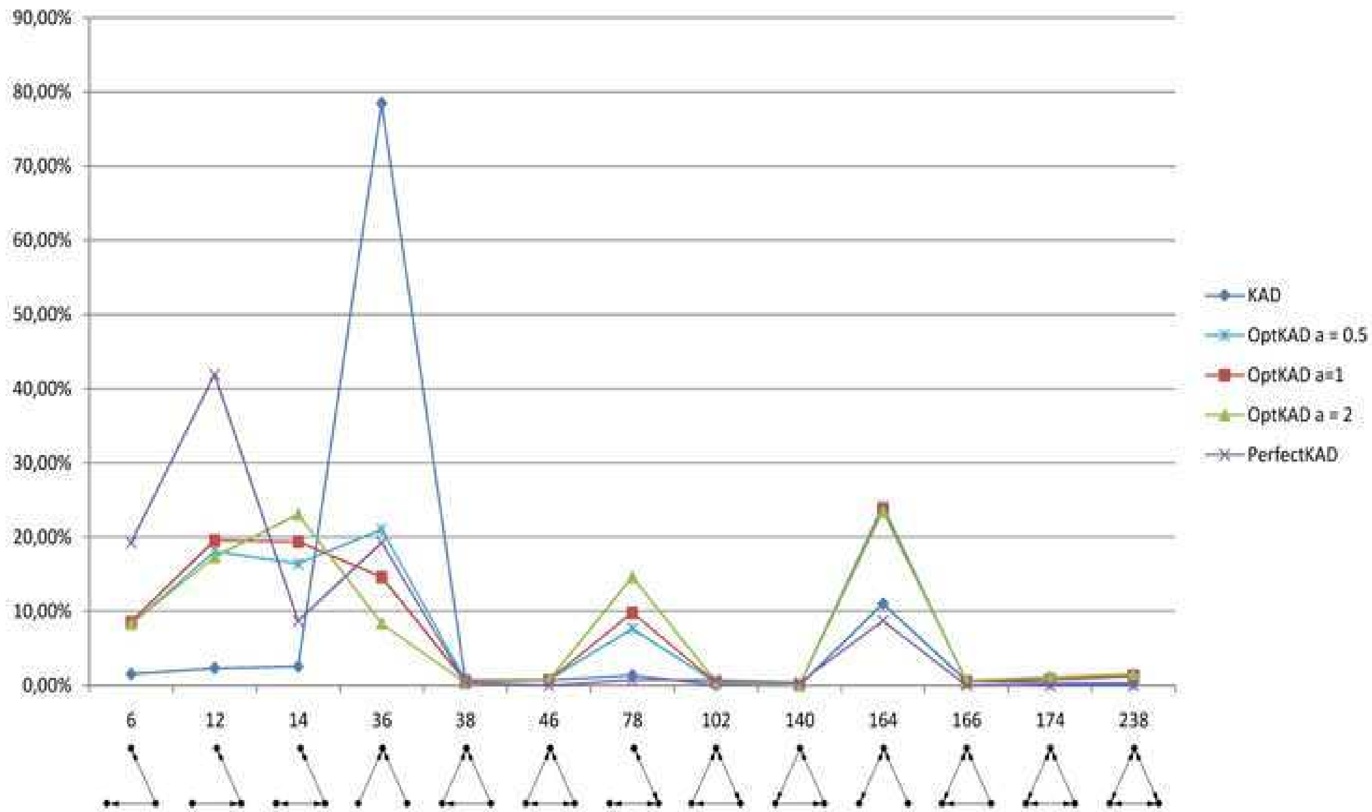
- Kademlia:
  - **Namespace is distributed evenly**
  - **Neighbor selection per XOR-metric (binary tree, connect to k nodes in „opposite“ branches)**
  - **Protocol leads to preferential attachment and unbalanced service requests**

- Assumption:
  - **„KAD is optimal when in-degree is balanced“**

- Directed triade motif signature is characteristic

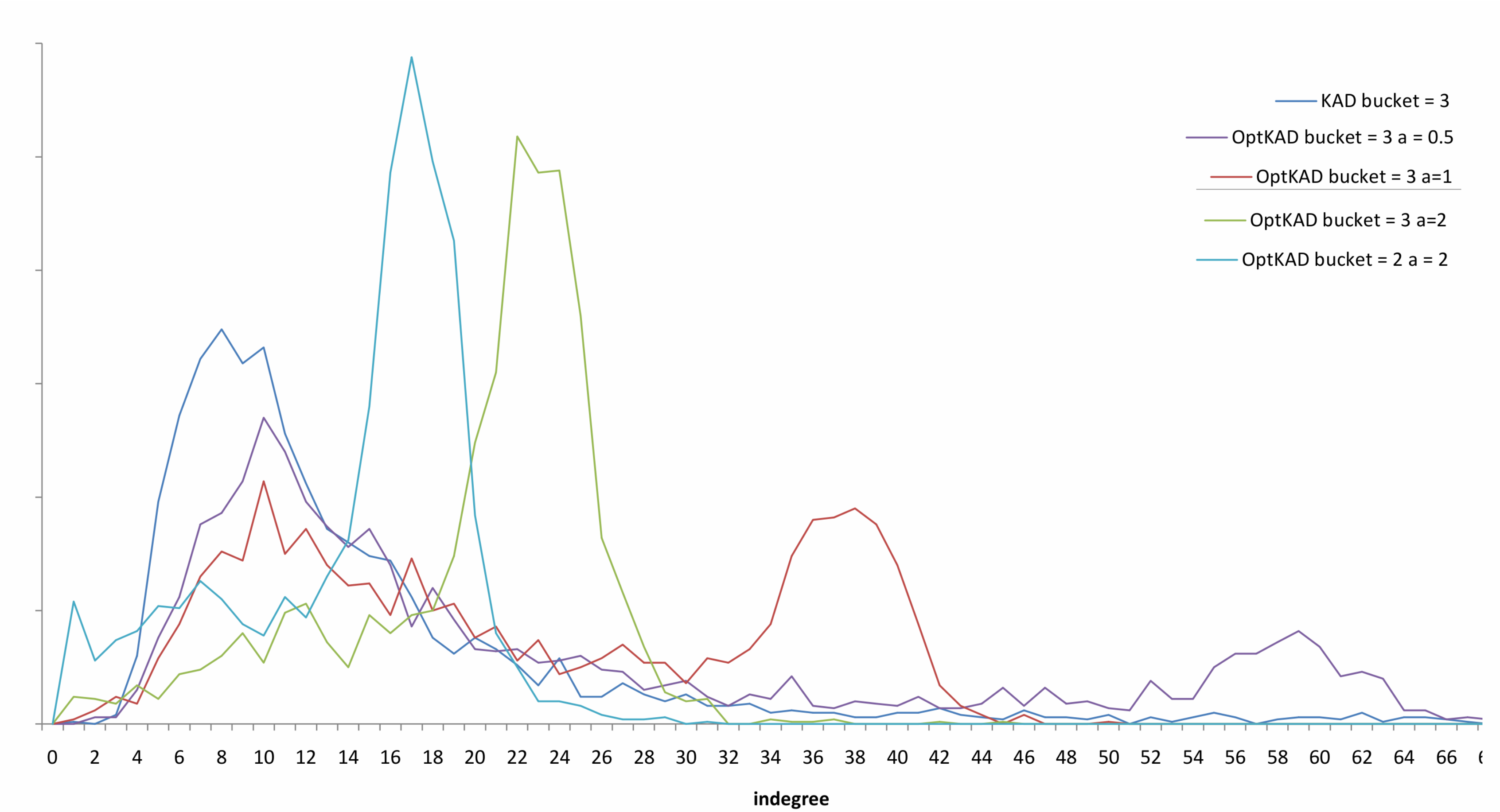
- Plugin-strategy:
  - **„Hide“ (drop message, avoid increase of in-degree)**

Significant Motif Profile successfully estimated?



n=2048, 32 runs, significant motif signature

Is the in-degree balanced?



n=2048, 32 runs, mean distribution of in-degree

Balancing Kad Request Processing

- Same idea for Kad:
  - Balance the in-degree
  - the same number of requests are expected for all
  - Problem: what is the mean? How would it be balanced?