Z-Ring: Fast Prefix Routing via a Low Maintenance Membership Protocol

Qiao Lian, Wei Chen, Zheng Zhang

Microsoft Research Asia, Beijing China

Shaomei Wu and Ben Y. Zhao

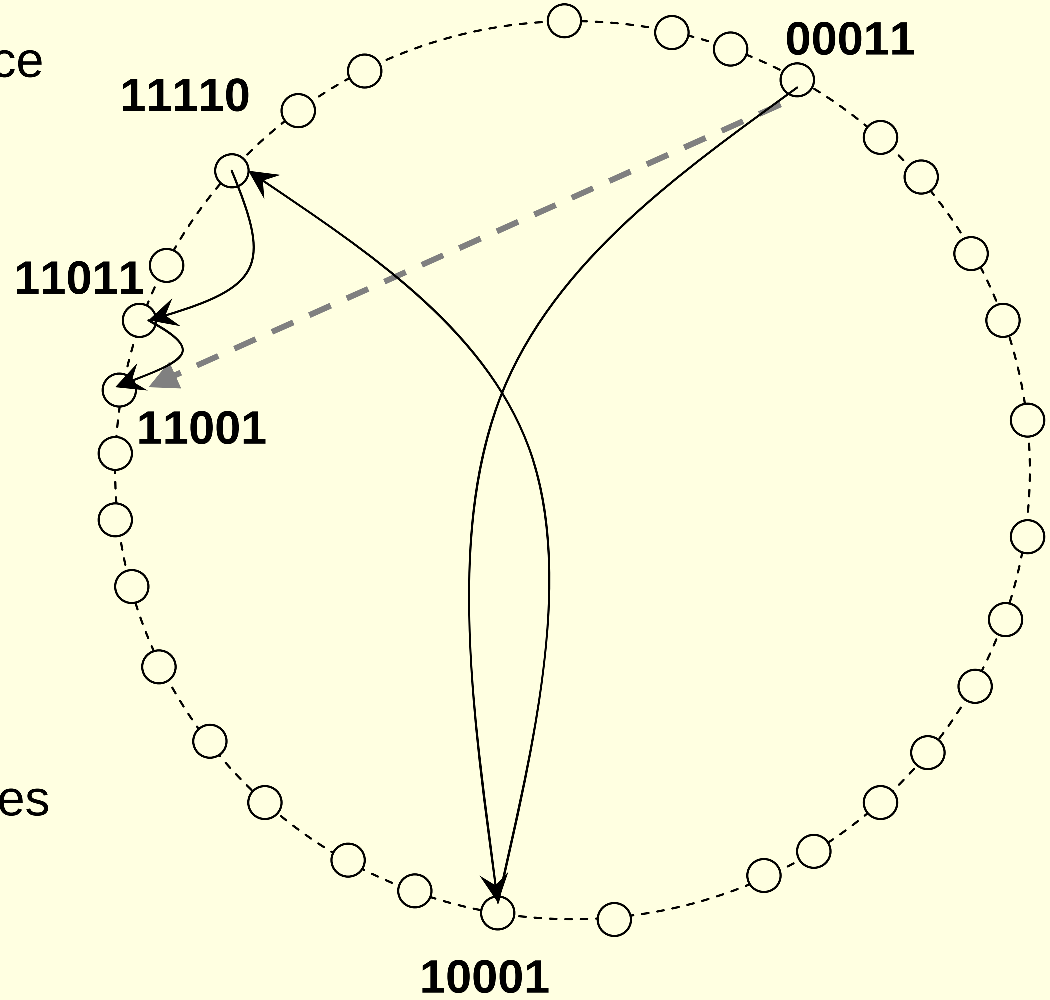
U. C. Santa Barbara

Structured Peer-to-Peer Overlays

- Scalable and resilient overlay networks
  - node to node routing in log *N* hops
  - per-node state ~ log *N*
  - support key-based routing (KBR)
    - route to any key *K*, arrive at *root (K)*
    - consistent across entire network
- Leverage KBR for network applications
  - DHT: store data at *root (K)*
  - DOLR: store directory data on way to *root (K)*
  - rendezvous at *root (K)*: network indirection, multicast, pub/sub

Prefix Routing: Pastry / Tapestry

- Incremental prefix routing
  - recursive routing from source
  - 00011→1XXXX→11XXX →110XX→11001
- Routing table
  - Each i^{th} level keeps nodes matching at least *i* digits to destination
  - Total table size: $b * \log_b n$
- Maintenance cost
  - Need to keep $b * \log_b n$ neighbors via periodic probes



What If…

- What if overlay apps get REALLY BIG?
  - Millions of peers across hundreds of AS's
  - E.g. a cellular p2p network, Windows PNRP
- $\log_b N$ hops is now too many hops
  - Maybe we can increase *b*? (ideally, to b $\geqslant 2^{10}$)
  - Bigger *b* means more routing state, higher maintenance cost
- One approach
  - use static hierarchy (a la One-hop Routing)
  - downside: static and non-adaptive

Z-Ring at 10,000 Feet

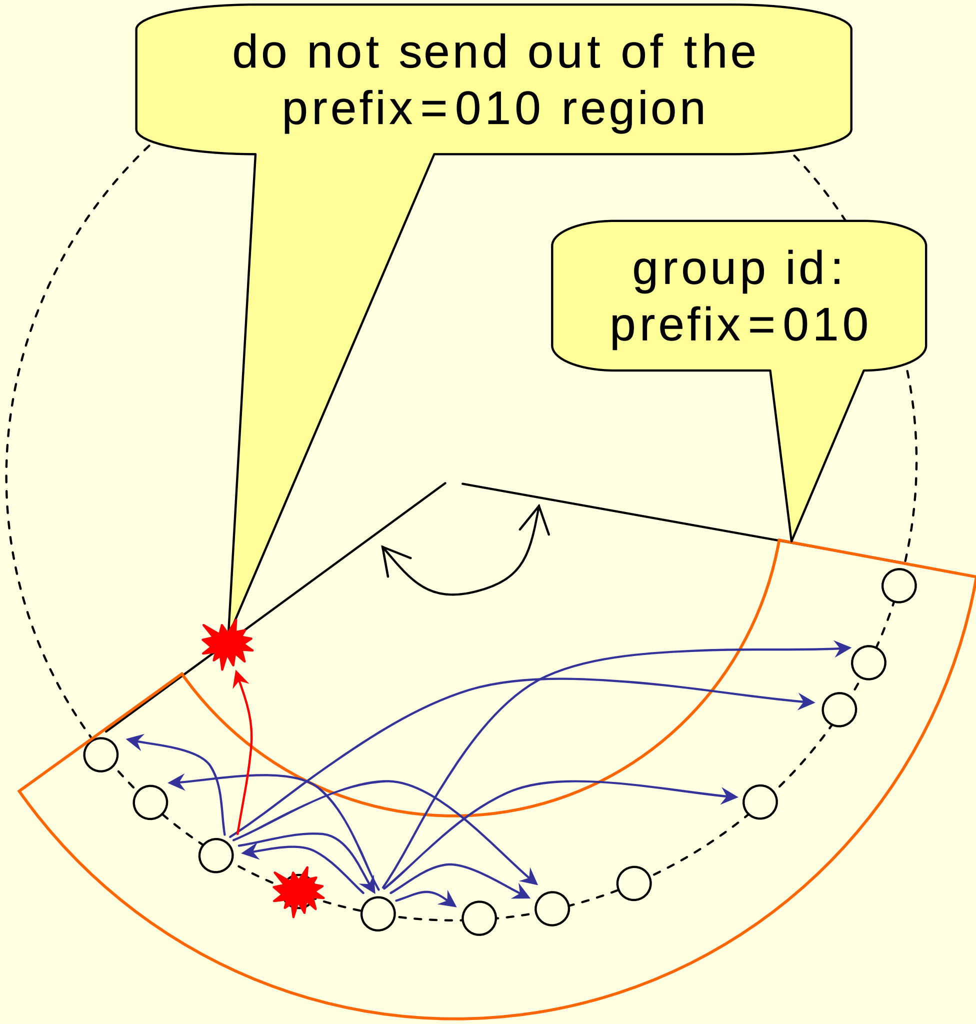
- Prefix routing (a la Tapestry / Pastry)
  - Similar to prefix routing with a REALLY LARGE base *b* (4K?)
  - One hop covers <= 4,000 peers
  - Two hops covers single autonomous system
  - Three hops covers significant portion of the Internet
- Reducing maintenance overhead
  - Make neighbors for a single hop members of an inclusive membership group (e.g. |group| ≈ 4096)
  - A node keeps *full* knowledge of all other nodes in its group
  - Maintenance inside each group like its own p2p network
    - actively monitor small local leafsets (4-5 peers)
    - disseminate join/leave to entire group via routing table entries

Outline

- Motivation
- Z-Ring
  - Cost efficient membership protocol
  - Acceleration on X-group routing
  - Acceleration on Y-group routing
- Inexact routing
- Experimental evaluation

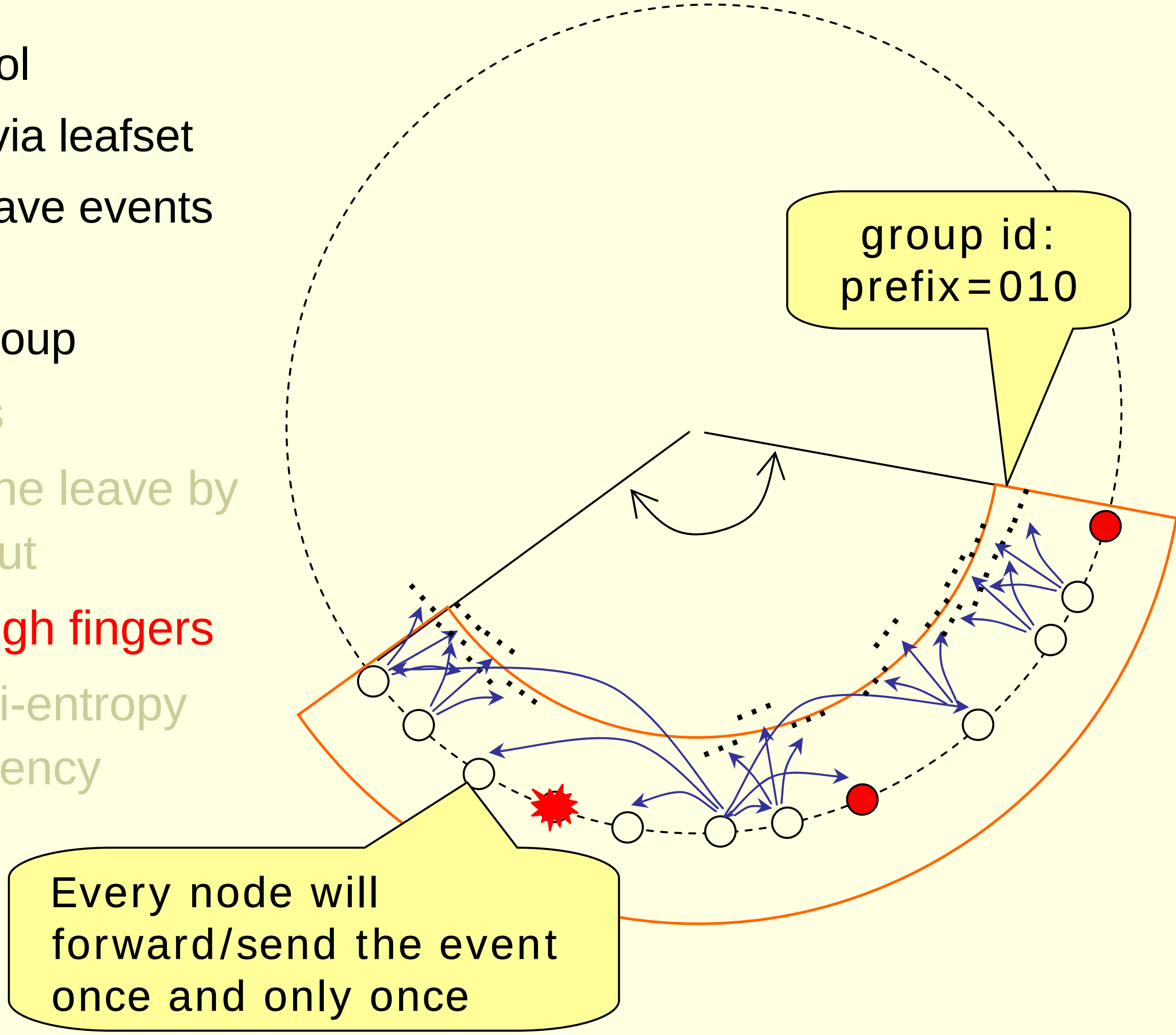
Cost-efficient Membership Protocol

- Probabilistic protocol
- Monitor neighbors via leafset
- Disseminate join/leave events via Pastry fingers
- Example: build a group
  1. Node X crashes
  2. Leafset detects the leave by heartbeat timeout
  3. Broadcast through fingers
  4. Background anti-entropy solves inconsistency



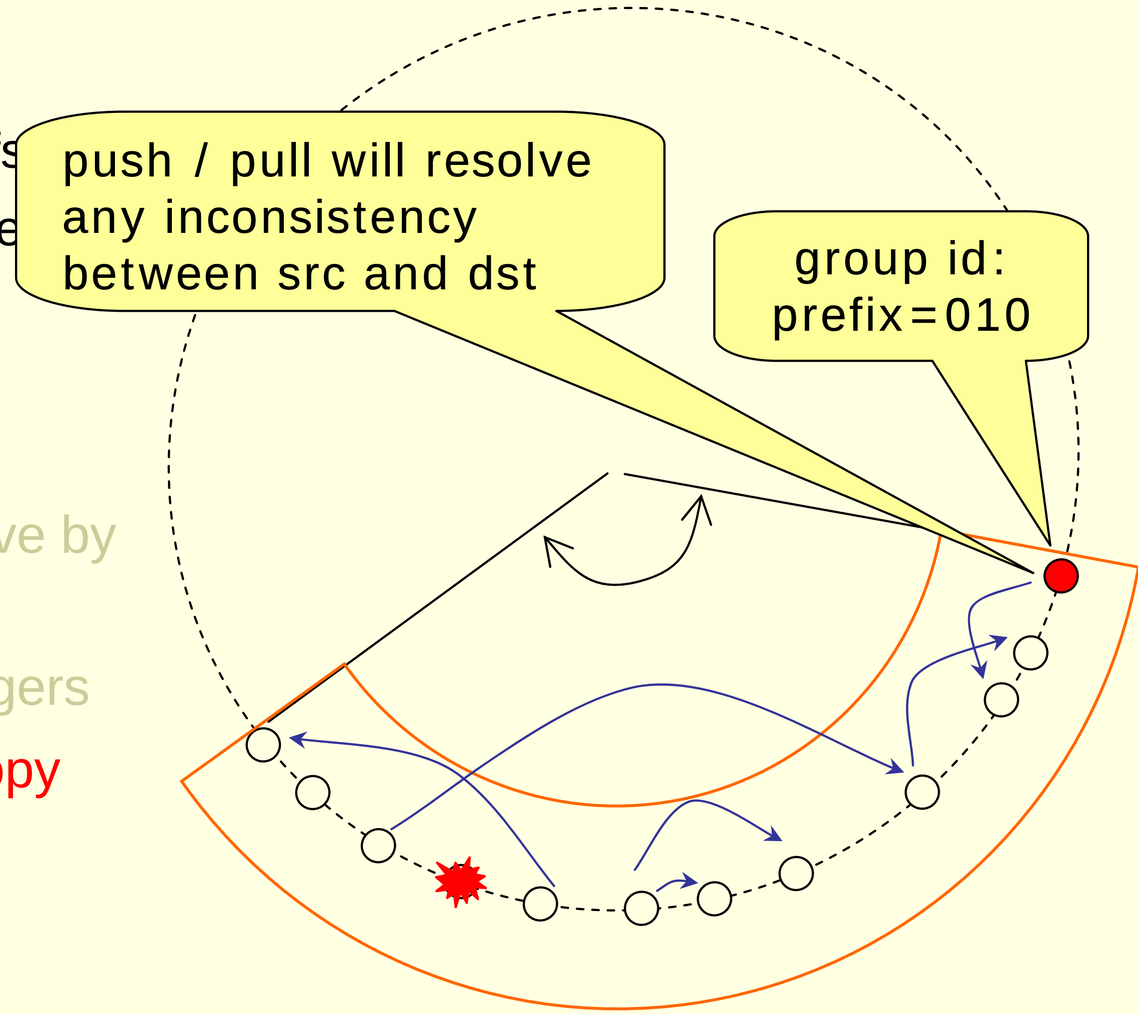
Cost-efficient Membership Protocol

- Probabilistic protocol
- Monitor neighbors via leafset
- Disseminate join/leave events via Pastry fingers
- Example: build a group
  1. Node X crashes
  2. Leafset detect the leave by heartbeat timeout
  3. Broadcast through fingers
  4. Background anti-entropy solves inconsistency



Cost-efficient Membership Protocol

- Probabilistic protocol
- Monitor neighbors via leafs
- Disseminate join/leave eve via Pastry fingers
- Example: build a group
  1. Node X crashes
  2. Leafset detect the leave by heartbeat timeout
  3. Broadcast through fingers
  4. Background anti-entropy solves inconsistency

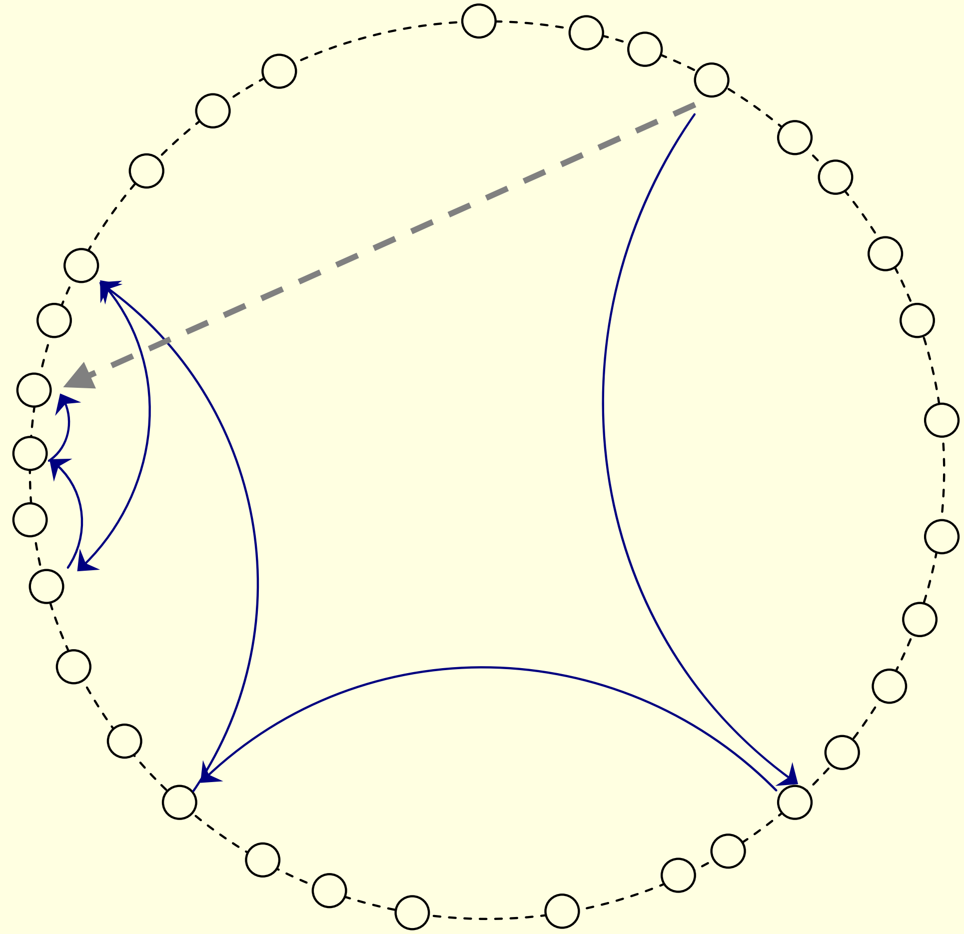


Outline

- Motivation
- Z-Ring
  - Cost efficient membership protocol
  - Acceleration on X-group routing
  - Acceleration on Y-group routing
- Inexact routing
- Experimental evaluation

Accelerate routing with X-groups

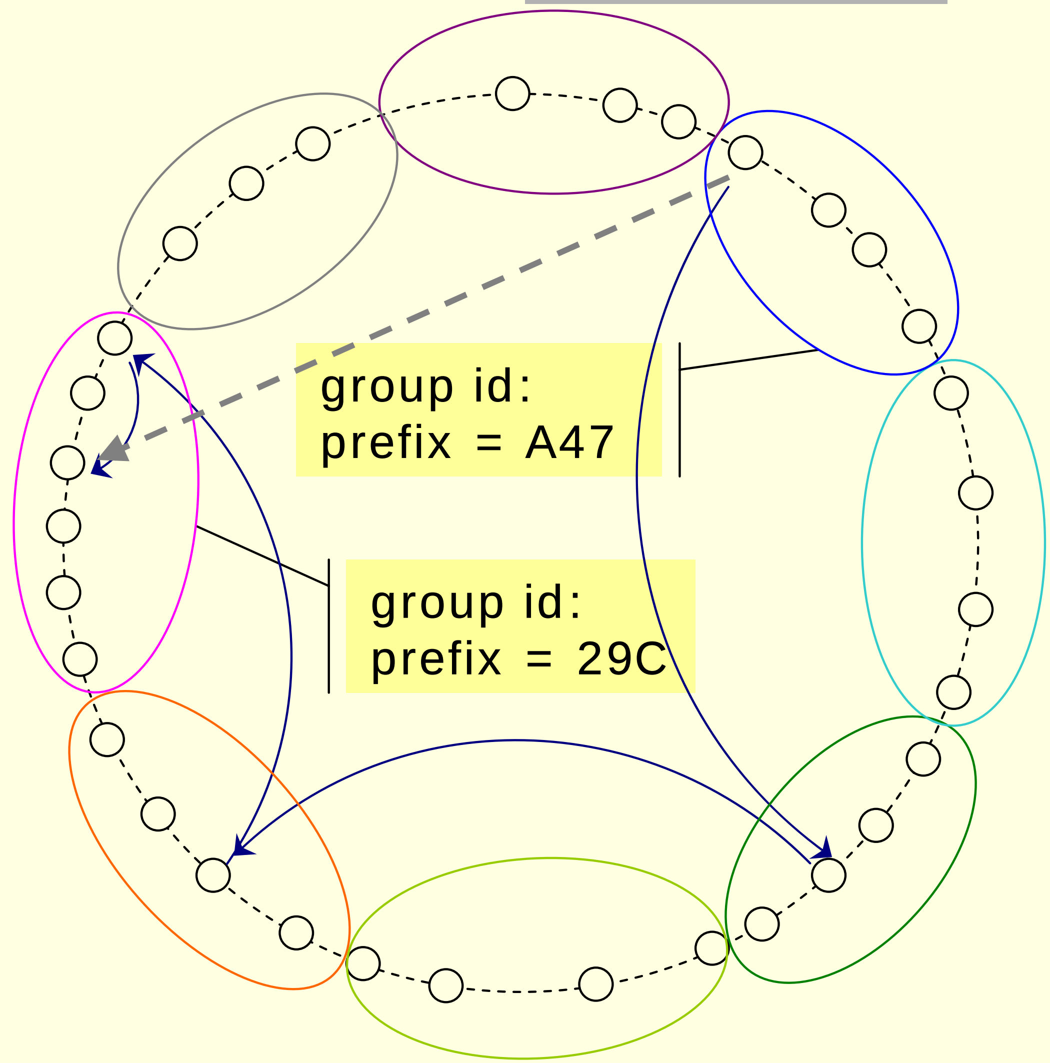
- “Slow” prefix routing: digit by digit
- X-group routing: resolves multiple bits in last hop (e.g. group size=4,096 matches 12 bits)

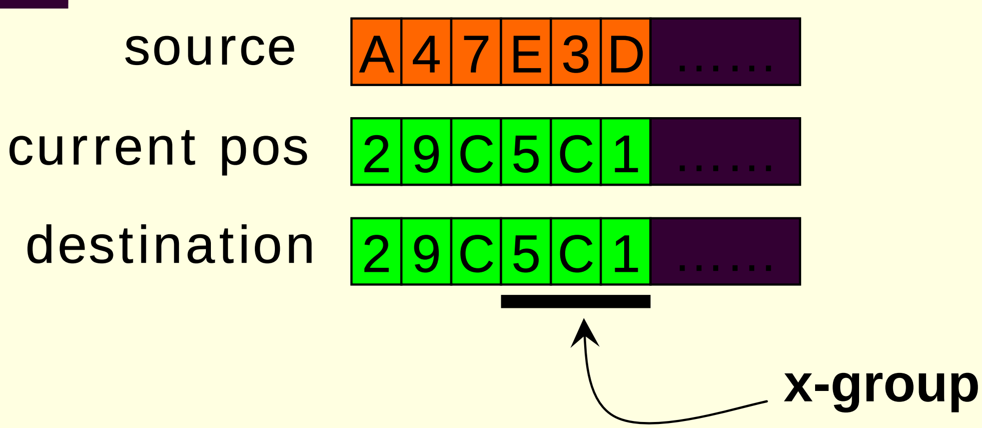


source	A	4	7	E	3	D	......
current pos	2	9	C	5	C	1	......
destination	2	9	C	5	C	1	......

Accelerate routing with X-groups

- “Slow” prefix routing: digit by digit
- X-group routing: resolves multiple bits in last hop (e.g. group size=4,096 matches 12 bits)



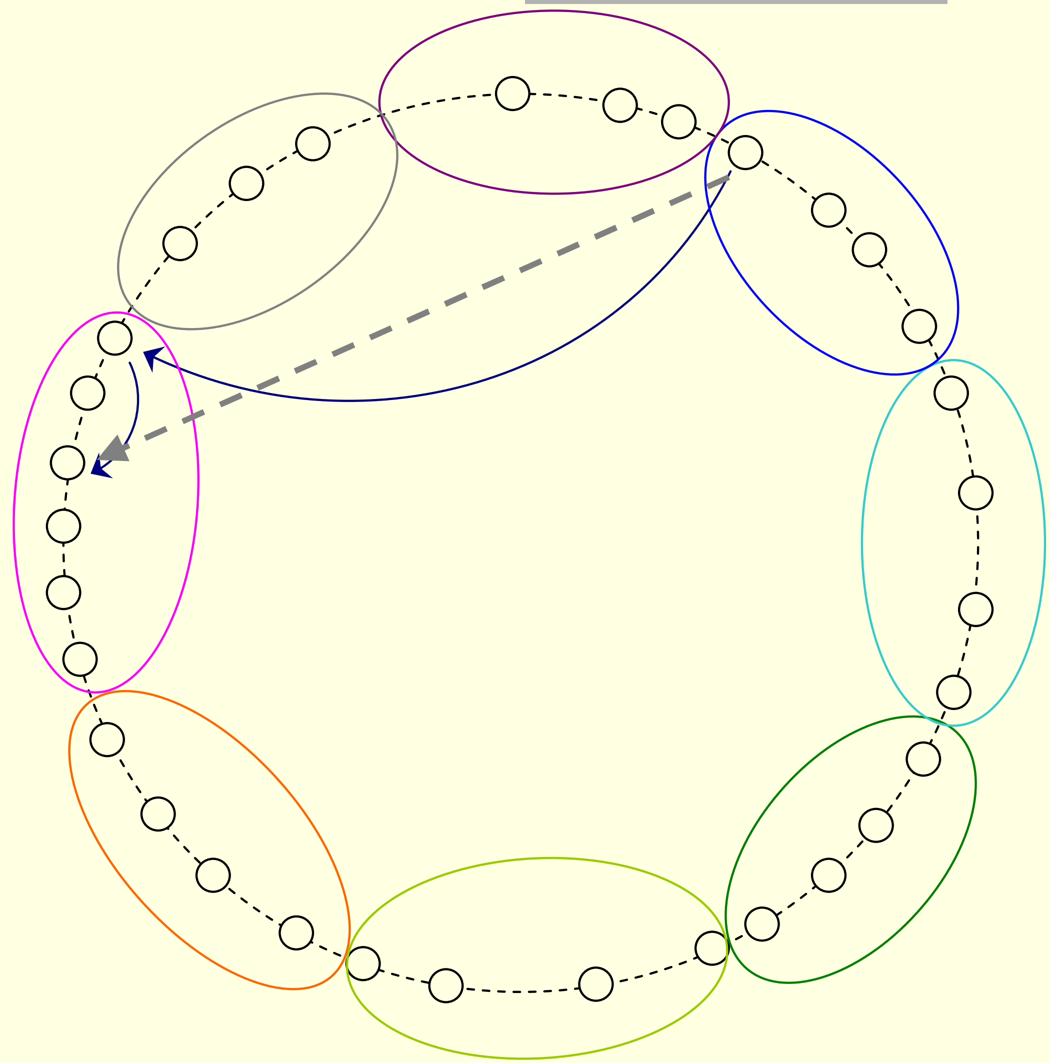


Outline

- Motivation
- Z-Ring
  - Cost efficient membership protocol
  - Acceleration on X-group routing
  - Acceleration on Y-group routing
- Inexact routing
- Experimental evaluation

Accelerate routing with Y-groups

- The problem:
  - Still multiple hops to get to destination X-group
  - How do we speed this up?
  - We use a “Y-group” for the first few digits

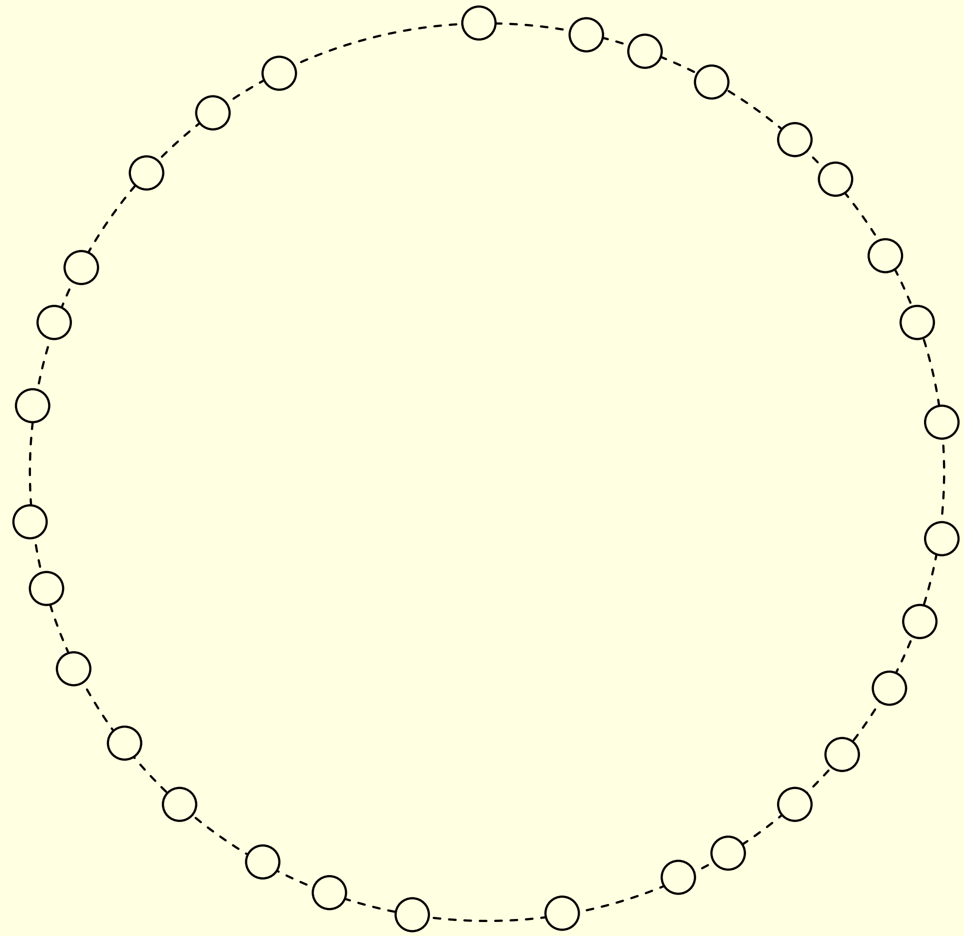


source	A	4	7	E	3	D	……
current pos	2	9	C	E	3	D	……
destination	2	9	C	5	C	1	……

The solution: Y-group routing

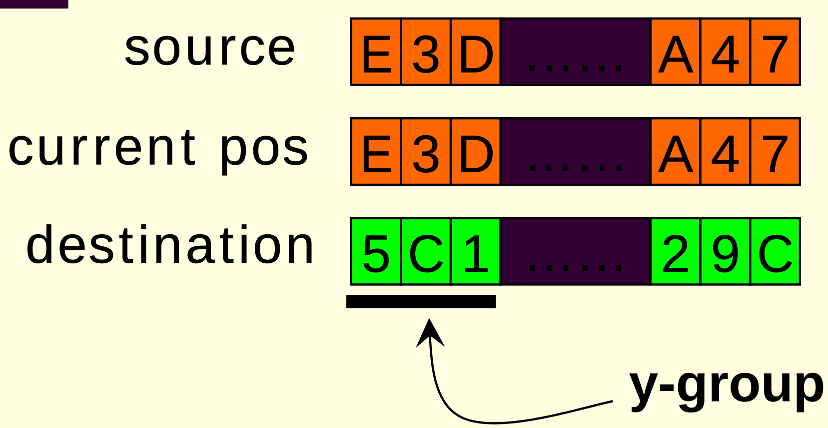
- Build Y-group routing table
  - Transform the key
  - Build another Pastry
  - Setup Y-groups
- Routing with Y-group
  - Return to original key
  - Y-group are sparsely distributed
  - Resolve multiple bits while keeping other bits the same
  - X-group routing

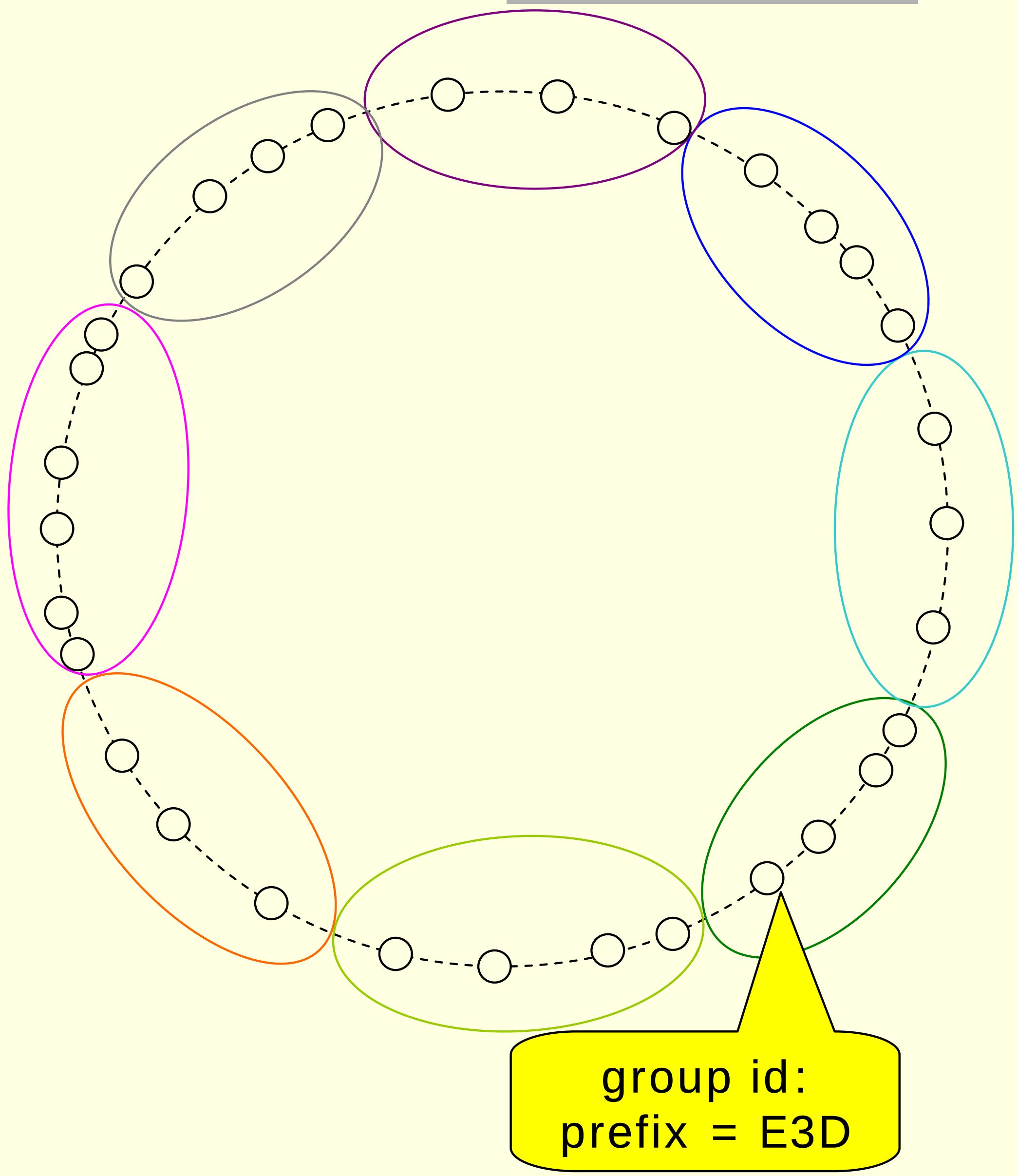
source	A	4	7	E	3	D	......
current pos	A	4	7	E	3	D	......
destination	2	9	C	5	C	1	......



The solution: Y-group routing

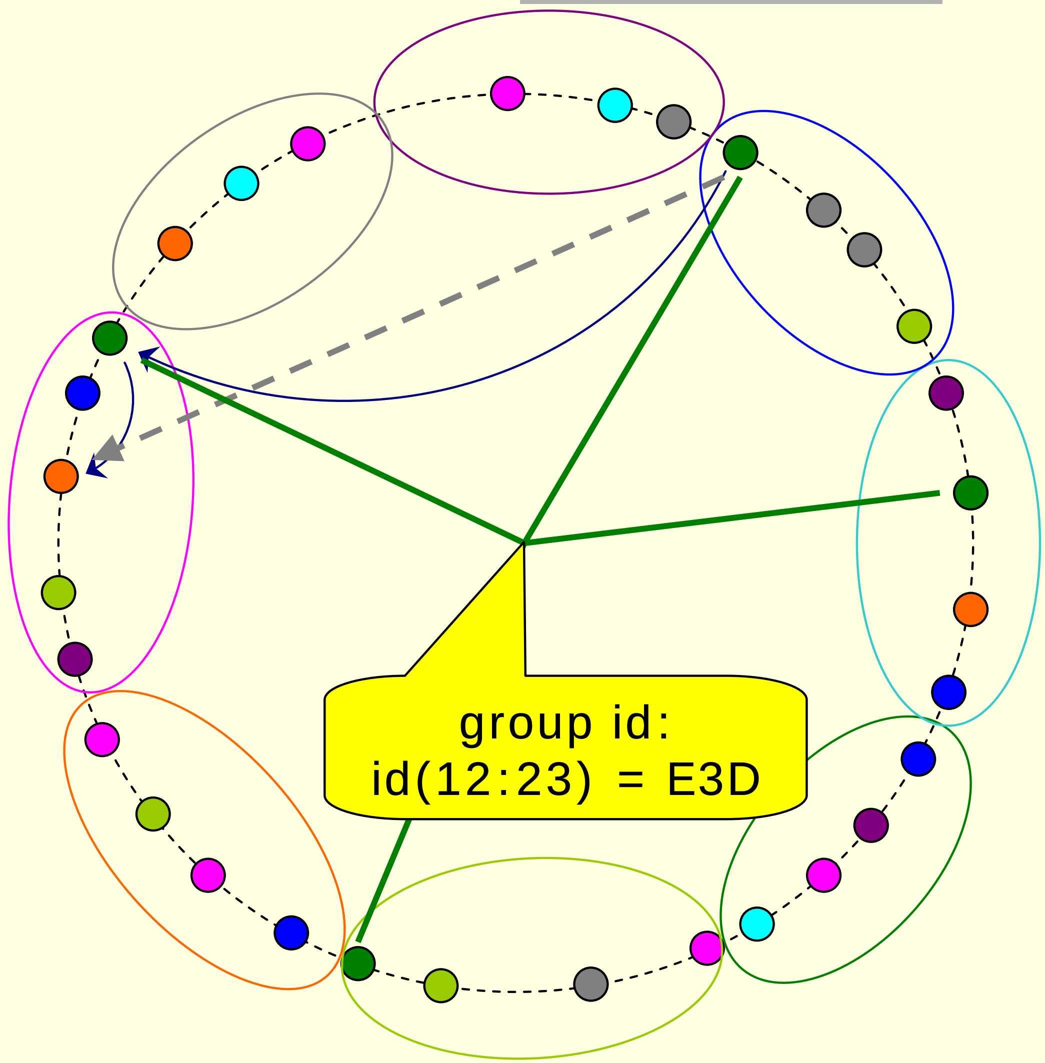
- Build Y-group routing table
  - Transform the key
  - Build another Pastry
  - Setup Y-groups
- Routing with Y-group
  - Return to original key
  - Y-group are sparsely distributed
  - Resolve multiple bits while keeping other bits the same
  - X-group routing





The solution: Y-group routing

- Build Y-group routing table
  - Transform the key
  - Build another Pastry
  - Setup Y-groups
- Routing with Y-group
  - Return to original key
  - Y-group are sparsely distributed
  - Resolve multiple bits while keeping other bits the same
  - X-group routing



source	A	4	7	E	3	D	......
current pos	2	9	C	5	C	1	......
destination	2	9	C	5	C	1	......

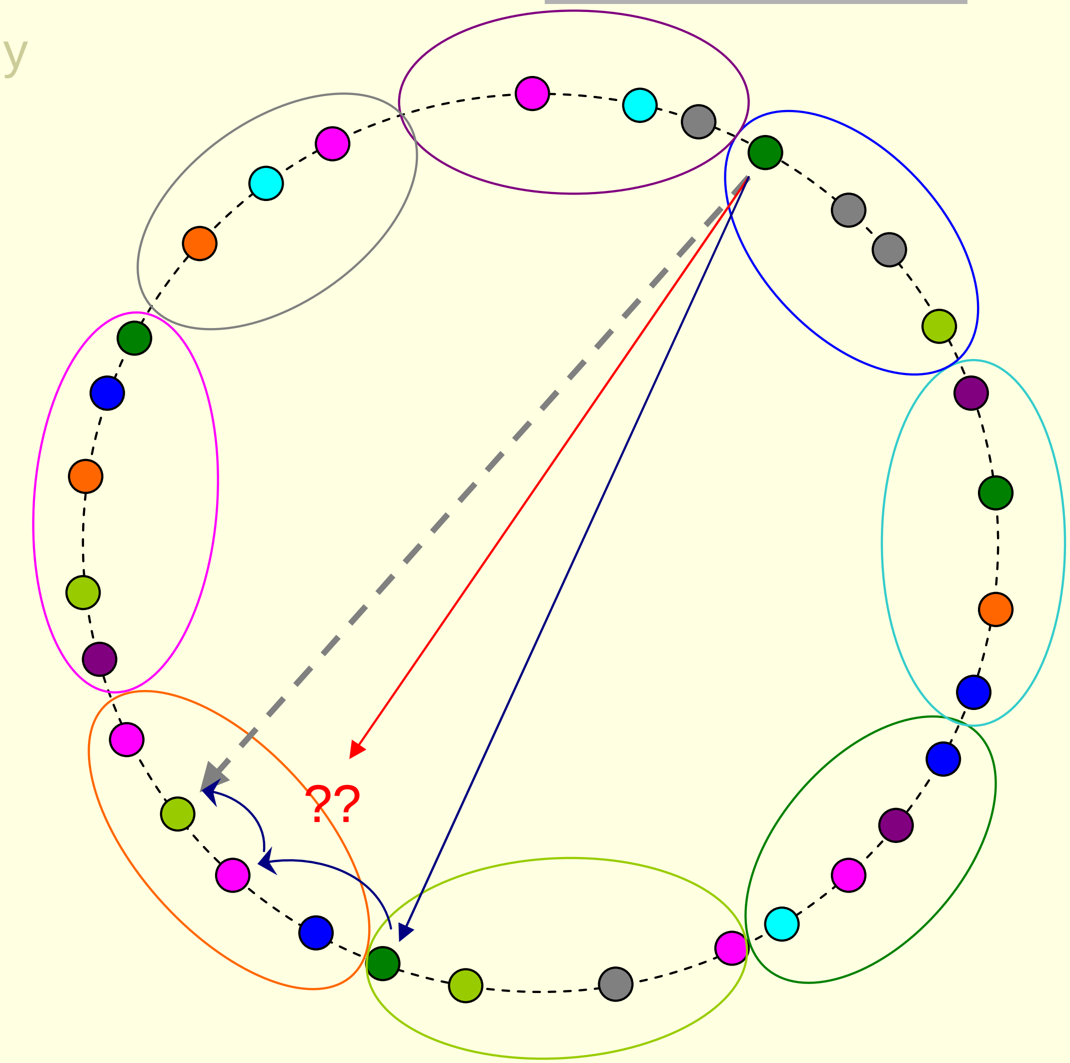
For bigger networks, we can use the same technique for more groups, Z, α, β, …

Outline

- Motivation
- Z-Ring
  - Cost efficient membership protocol
  - Acceleration on X-group routing
  - Acceleration on Y-group routing
- Inexact routing
- Experimental evaluation

Y-group routing failures

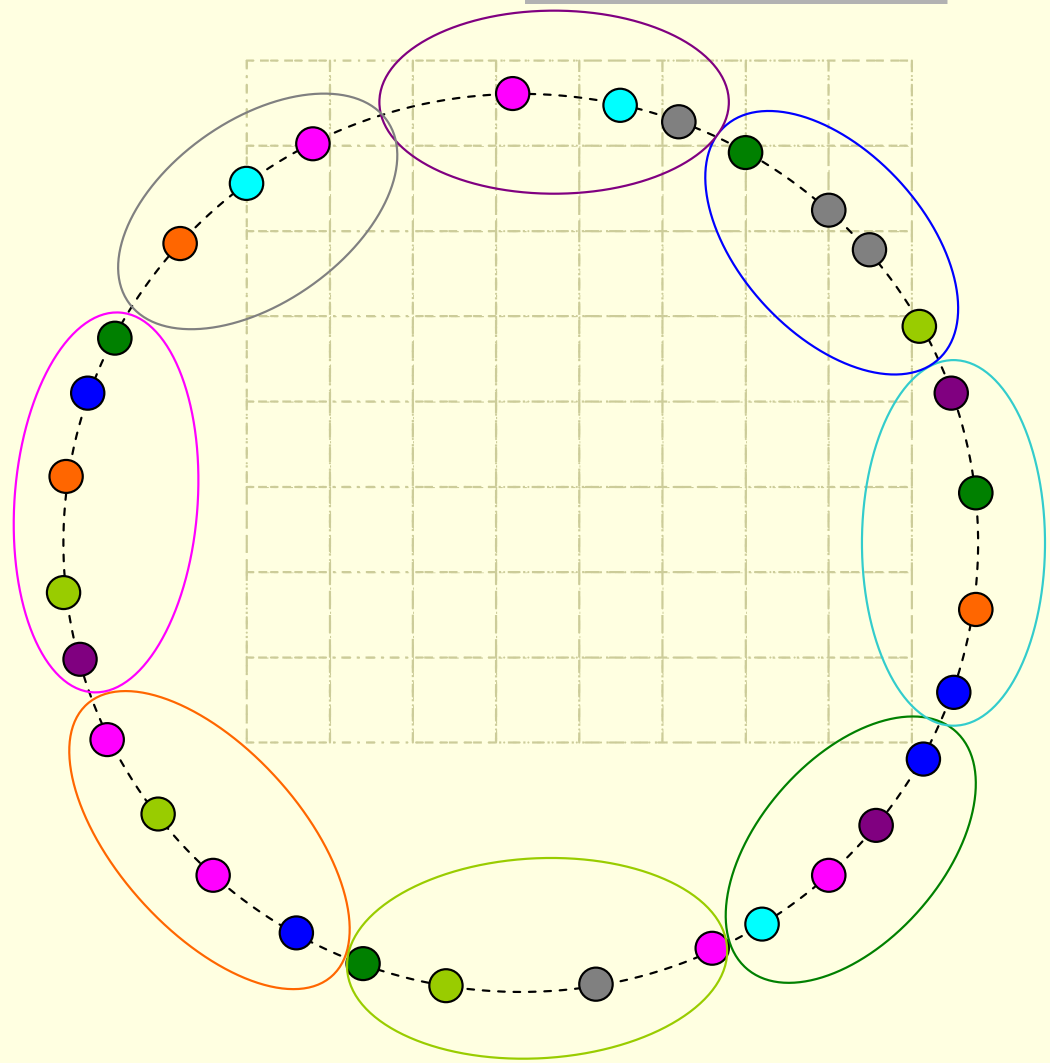
- Failure: there is no Y-group buddy in the dst X-group
- Y-group routing greedily goes to closest (resolves *n*-*i* bits)
- Supplementary Pastry finger hop
- X-group routing resolves remaining bits



source	A	4	7	E	3	D	......
current pos	5	A	2	8	1	E	......
destination	5	A	2	8	1	E	......

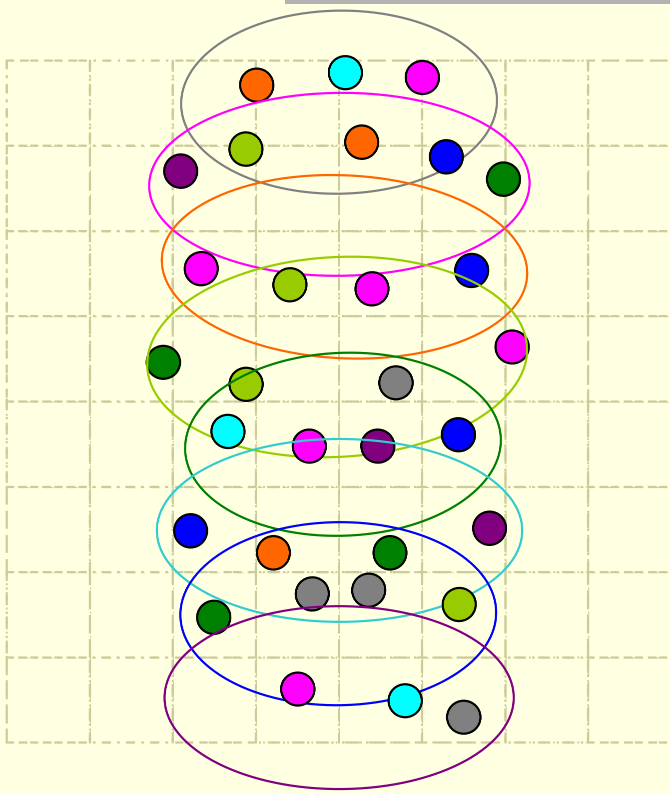
Another View of Inexact Routing

1. Rearrange the nodes in matrix view
2. Y-group routing is one vertical move
3. X-group routing is one horizontal move
4. Y-group routing fails when there is no node in the conjunction cell of src's Y-group and dst's X-group



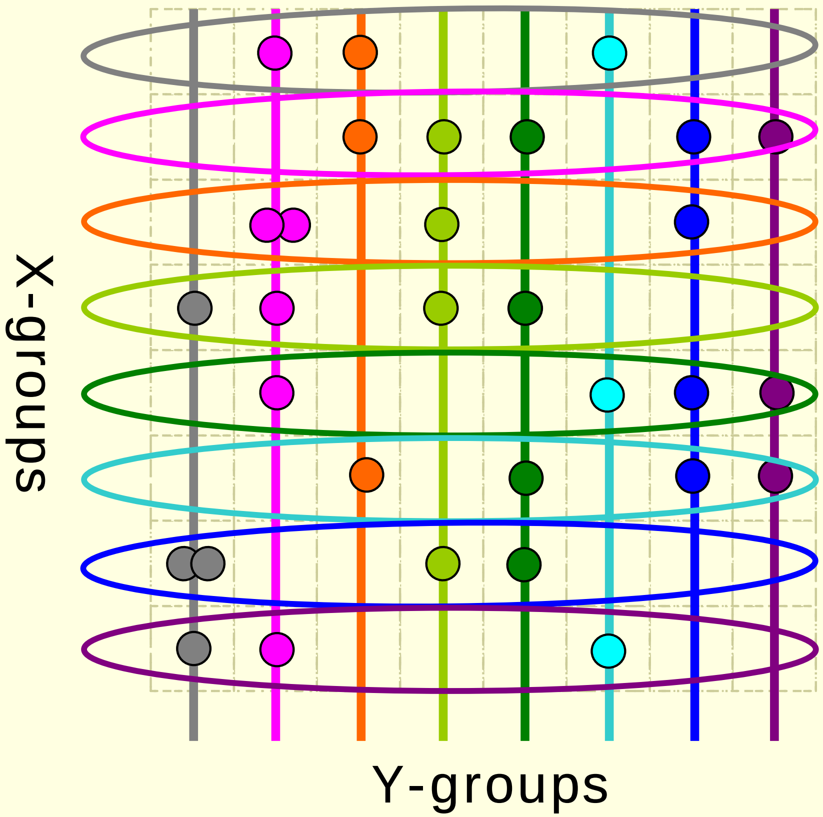
Another View of Inexact Routing

1. Rearrange the nodes in matrix view
2. Y-group routing is one vertical move
3. X-group routing is one horizontal move
4. Y-group routing failed when there is no node in the conjunction cell of src's Y-group and dst's X-group



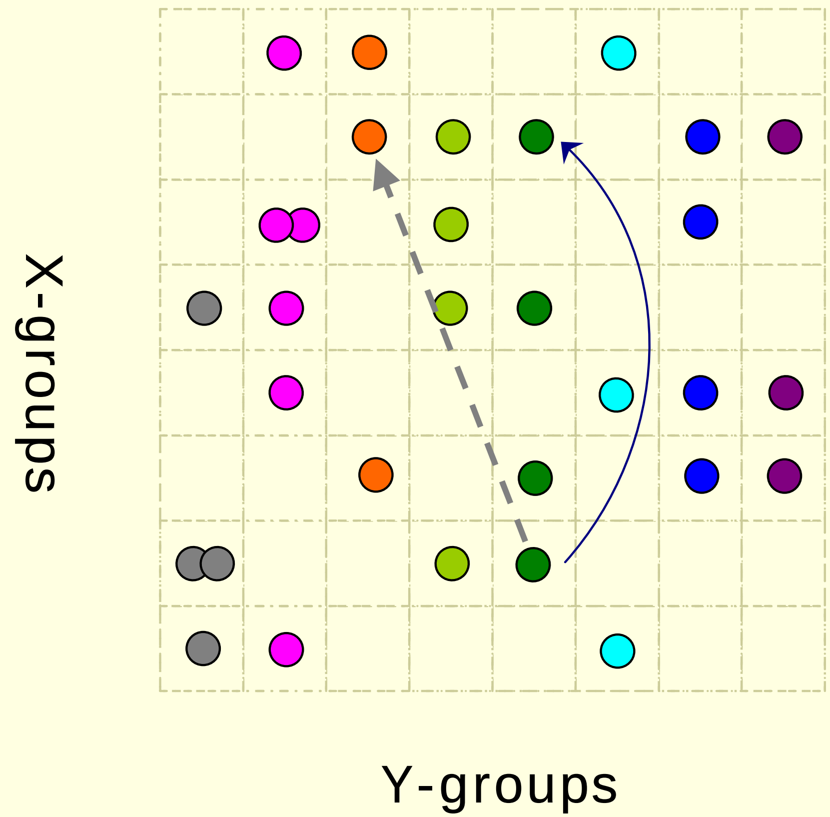
Another View of Inexact Routing

1. Rearrange the nodes in matrix view
2. Y-group routing is one vertical move
3. X-group routing is one horizontal move
4. Y-group routing failed when there is no node in the conjunction cell of src's Y-group and dst's X-group



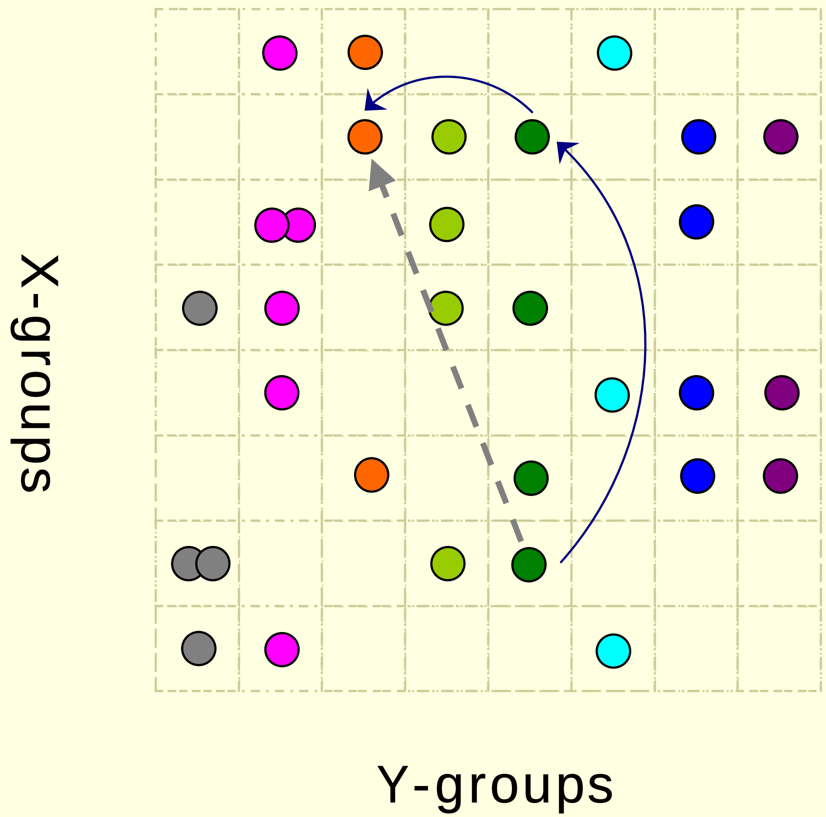
Another View of Inexact Routing

1. Rearrange the nodes in matrix view
2. Y-group routing is one vertical move
3. X-group routing is one horizontal move
4. Y-group routing failed when there is no node in the conjunction cell of src's Y-group and dst's X-group



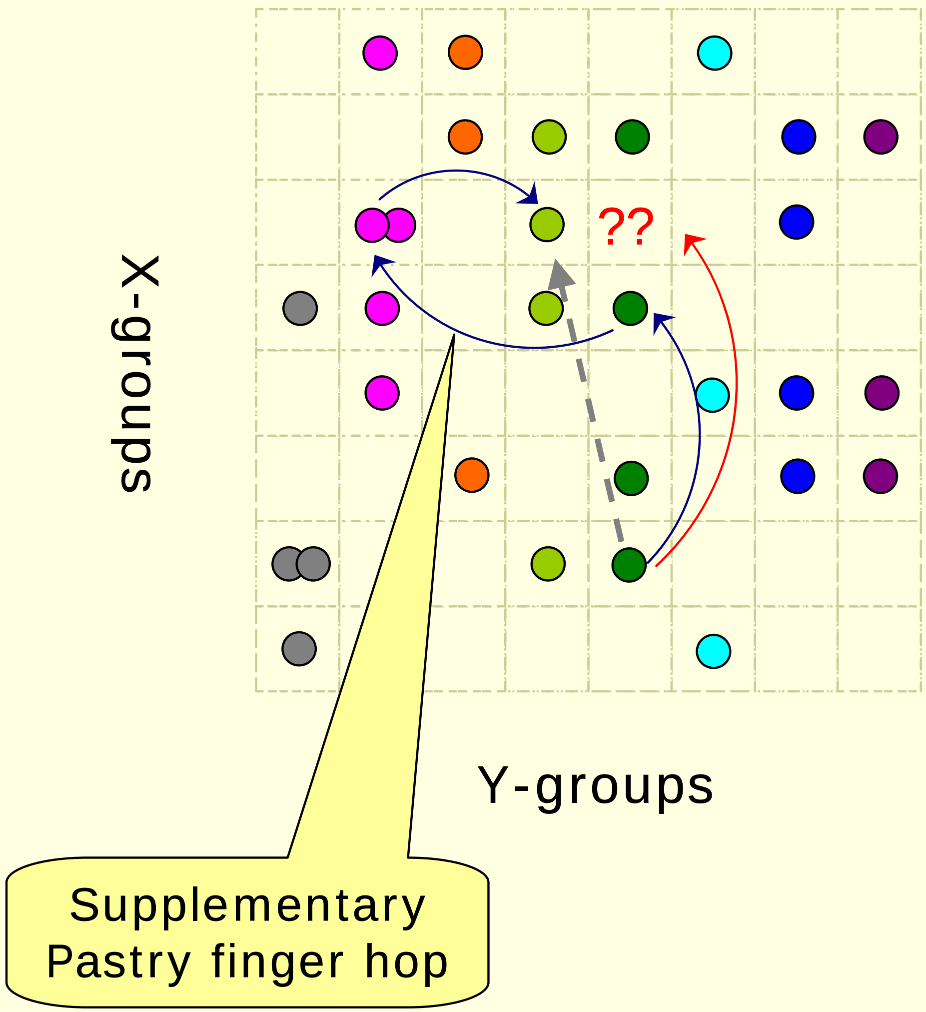
Another View of Inexact Routing

1. Rearrange the nodes in matrix view
2. Y-group routing is one vertical move
3. X-group routing is one horizontal move
4. Y-group routing failed when there is no node in the conjunction cell of src's Y-group and dst's X-group



Another View of Inexact Routing

1. Rearrange the nodes in matrix view
2. Y-group routing is one vertical move
3. X-group routing is one horizontal move
4. Y-group routing failed when there is no node in the conjunction cell of src's Y-group and dst's X-group



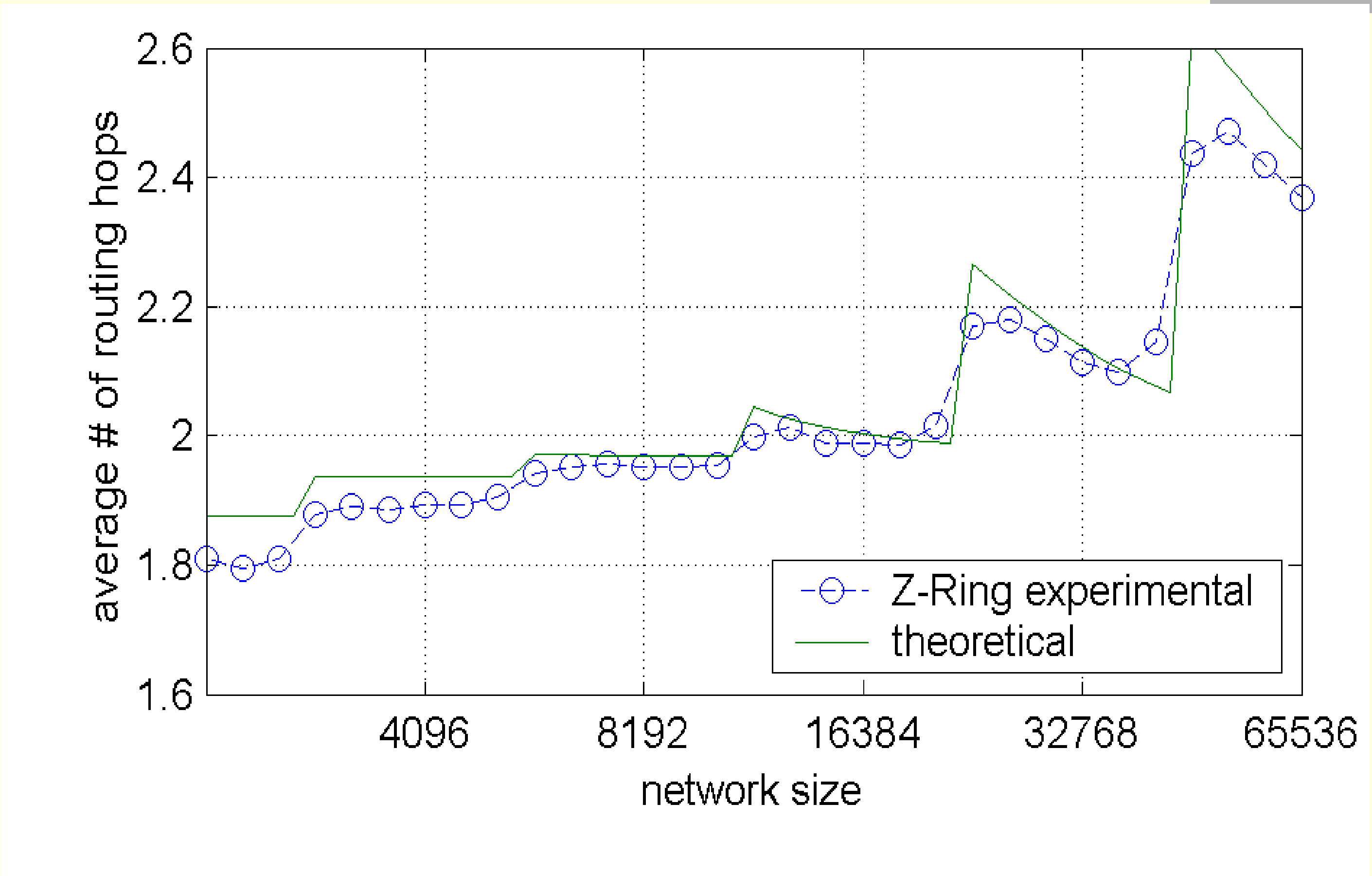
Z-Ring Routing Recap

- Organize name resolution into large “digits”
  - Each “digit” represented by 12 bits
  - Each 12 bits constitutes a “group”
  - Each group has mini-Pastry, with one leafset per node
- If no exact match across groups, route to nearby group, use normal “Pastry” to cross over
- Adaptive to changes in network size
  - Groups use high/low watermarks to split/join
  - # of groups can grow if network grows by factor of *G*
  - Details in paper

Outline

- Motivation
- Z-Ring
  - Cost efficient membership protocol
  - Acceleration on X-group routing
  - Acceleration on Y-group routing
- Inexact routing
- Experimental evaluation

Experiment: routing performance



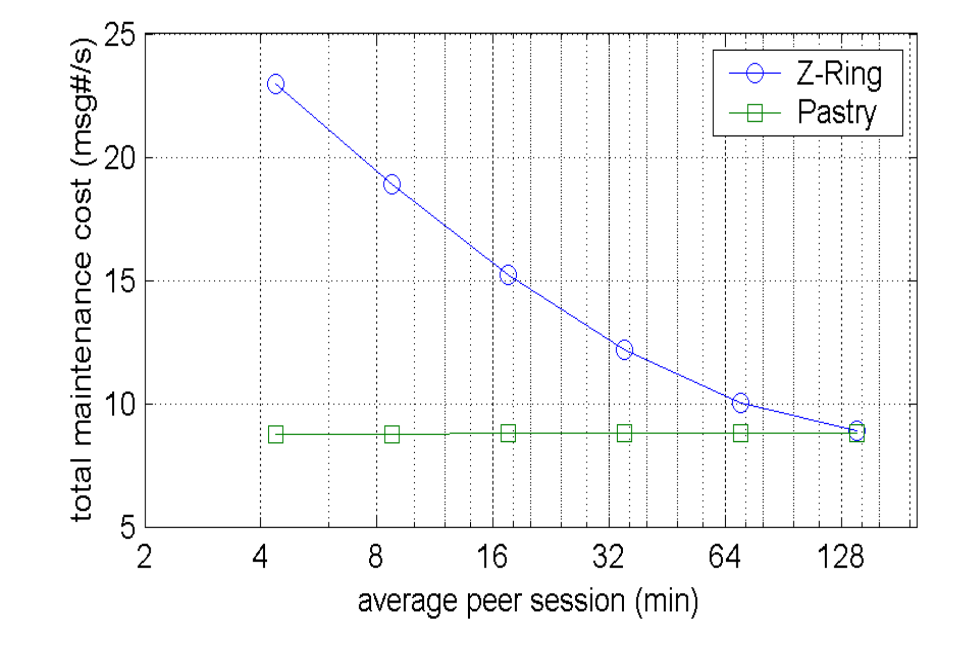
Simulations on WiDS distributed emulation framework

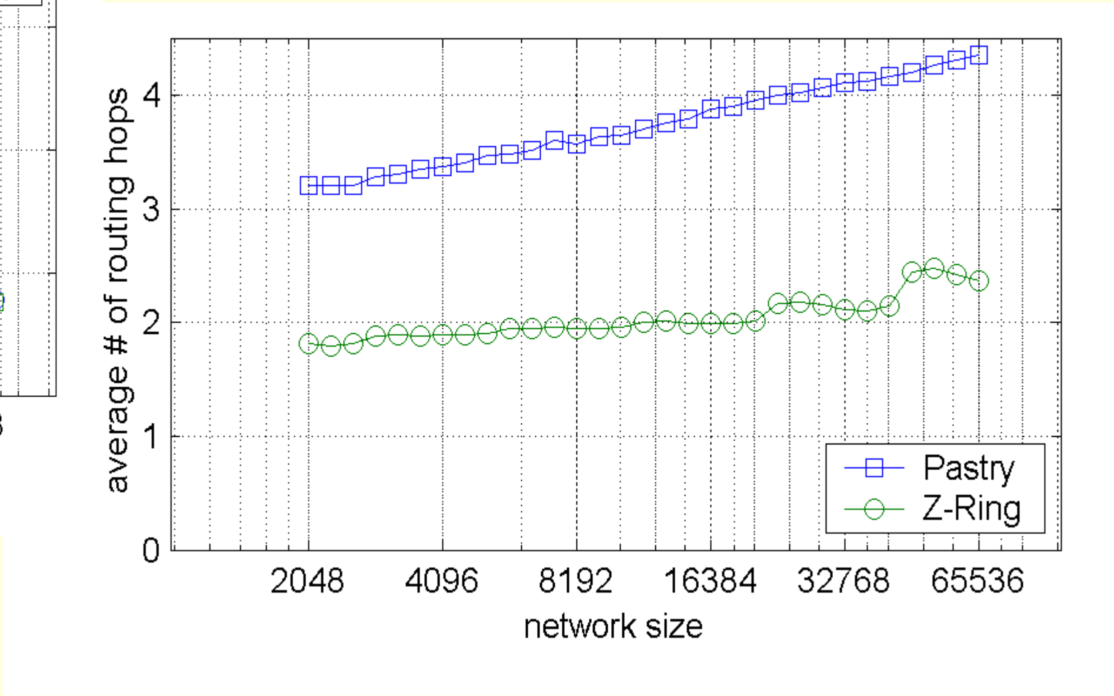
Each point is average from 2000 simulation runs (G=256)

Implementation results match analytical results

Experiment: comparing to Pastry

Maintenance cost comparison (G=256)
Network size = 65536

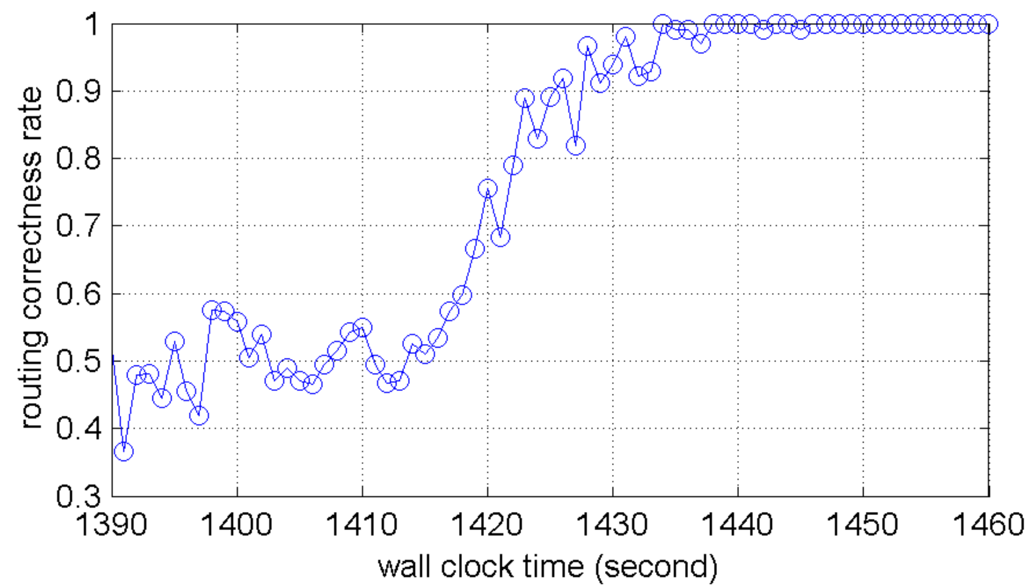




Routing hops comparison (G=256)
Compared to Pastry (base=16)

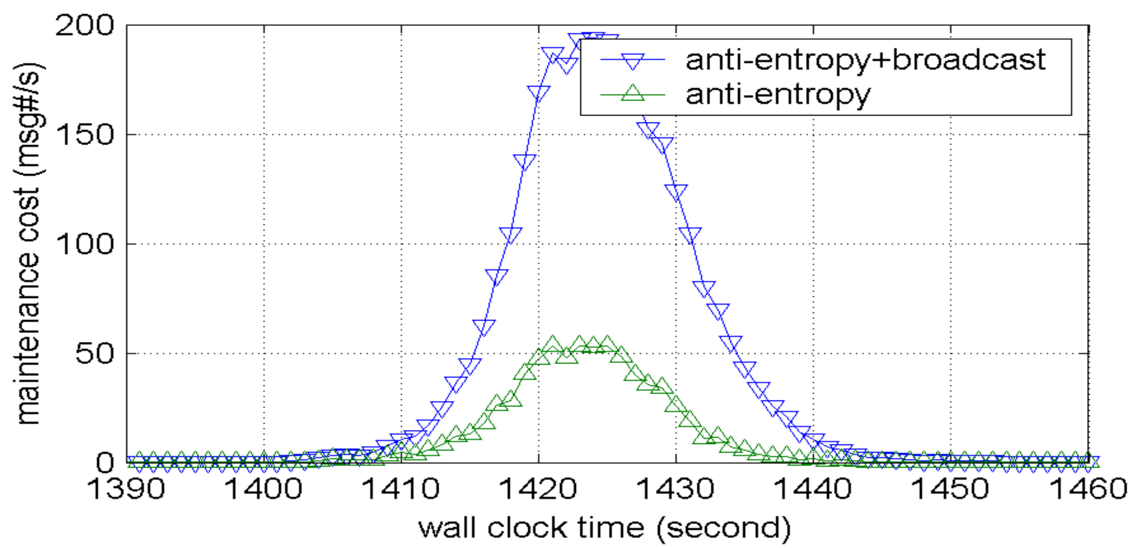
Experiment: fast partition healing

Split 65536 network into 2,
Partition healed after 50s
(5 heartbeat periods)



Routing correctness (G=256)
Pairwise routing every 10ms

Broadcast, anti-entropy
messages / s per peer
(G=256)



In Conclusion…

- Z-Ring uses large groups to accelerate P2P routing
  - Internal group membership protocols improve stability and reduce maintenance traffic
  - Size (and #) of groups can grow adaptively
- Evaluation via implementation and emulation
  - Fast routing performs as expected
  - Fast membership updates enables fast partition healing
  - More experimental results in paper

Thank you…

Questions?

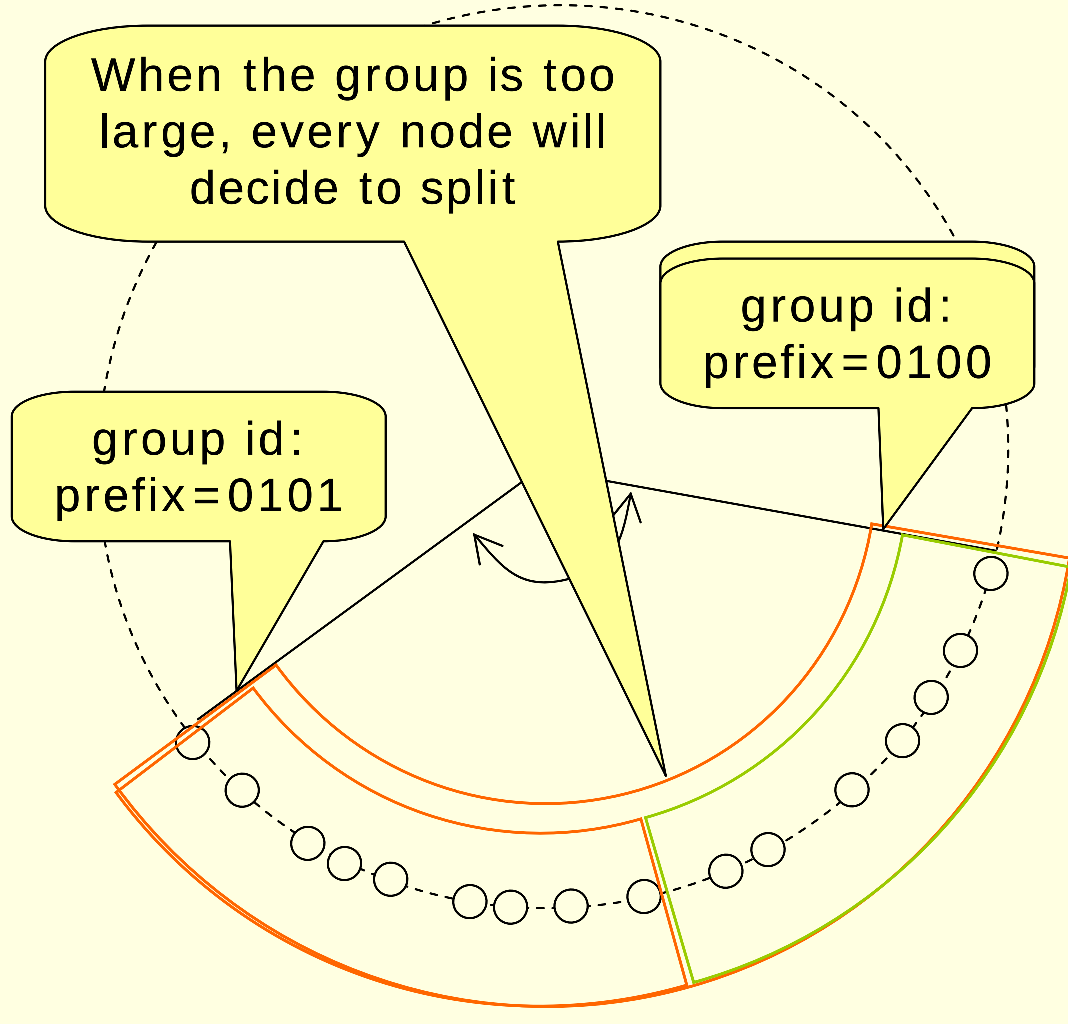
{qiaol,weic,zzhang}@microsoft.com

{shaomei,ravenben}@cs.ucsb.edu

Backup Slides Follow…

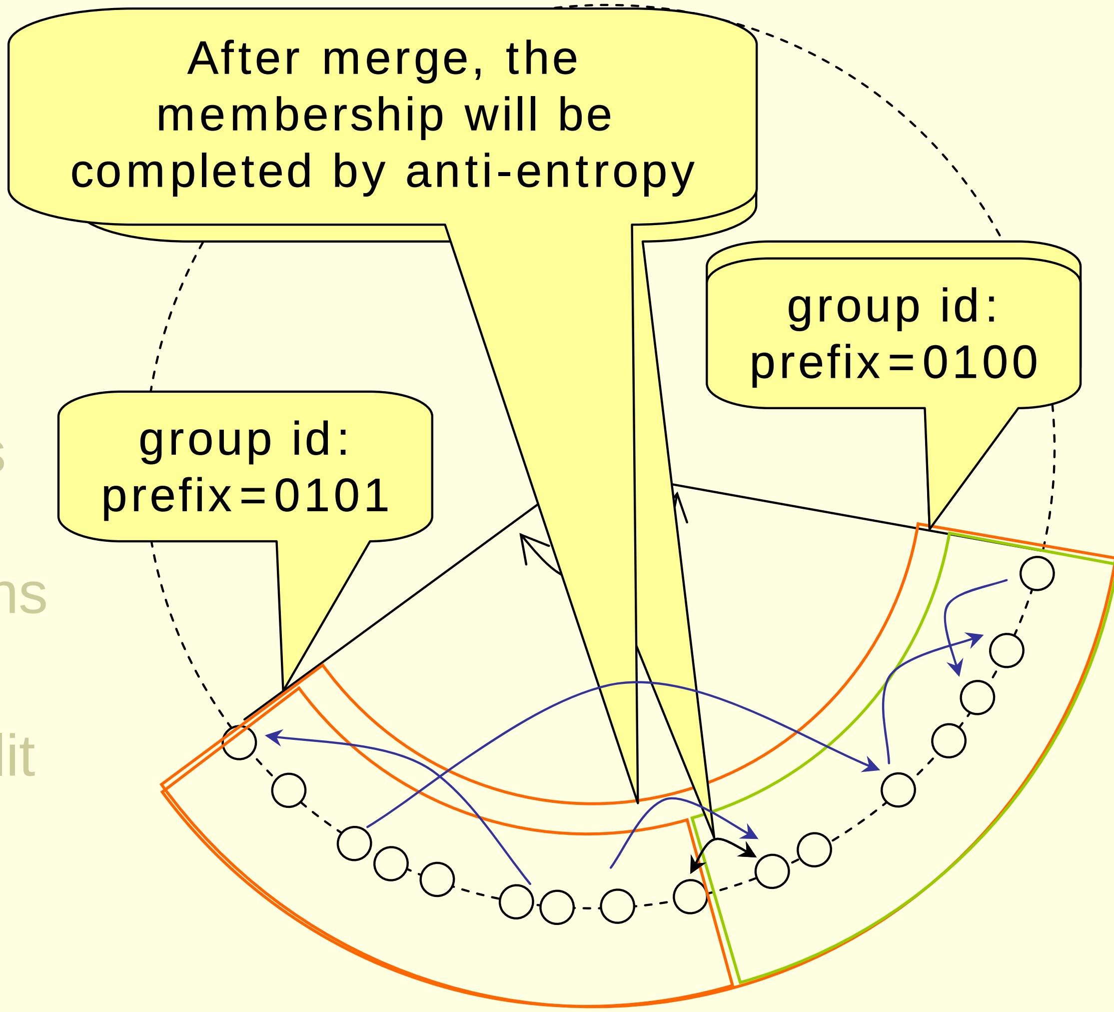
Adaptation and group mngt.

- Group splits
  - Nodes join makes group oversized
  - Group split happens
- Group merges
  - Nodes leaves makes group undersized
  - Group merge happens
- Impl. Tip
  - Delayed merge & split prevents fluctuation

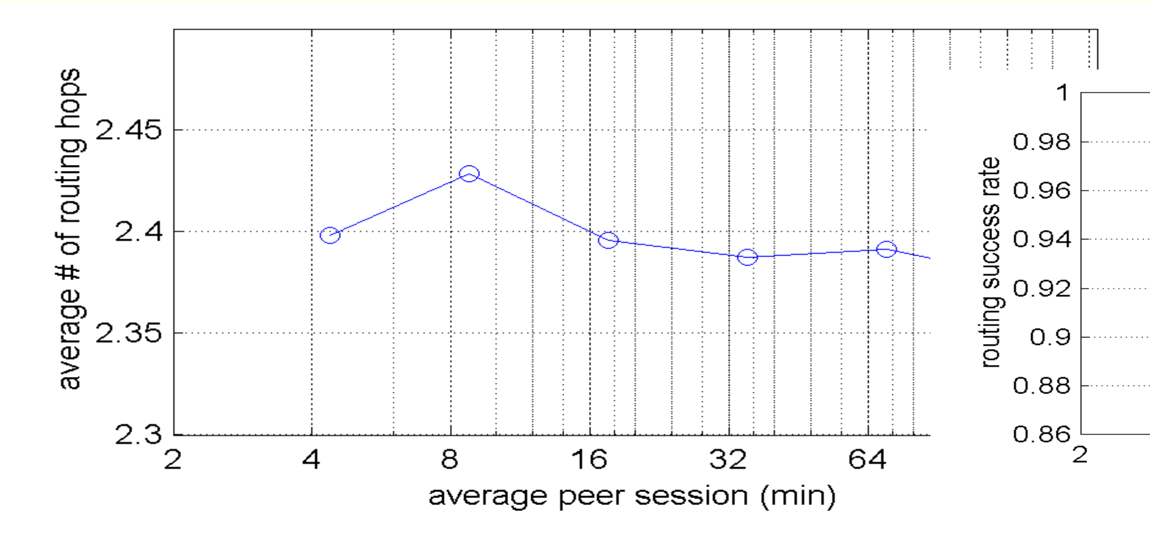


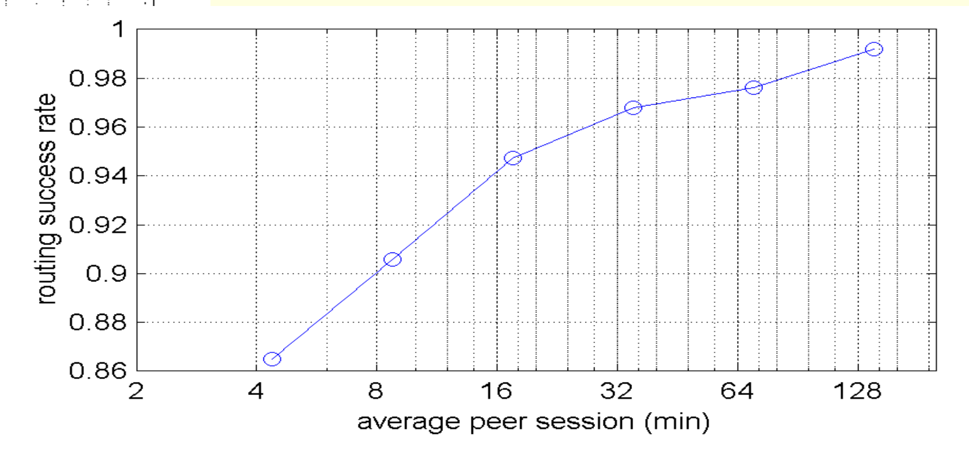
Adaptation and group mngt.

- Group splits
  - Nodes join makes group oversized
  - Group split happens
- Group merges
  - Nodes leaves makes group undersized
  - Group merge happens
- Impl. Tip
  - Delayed merge & split prevents fluctuation



Experiment: under churn

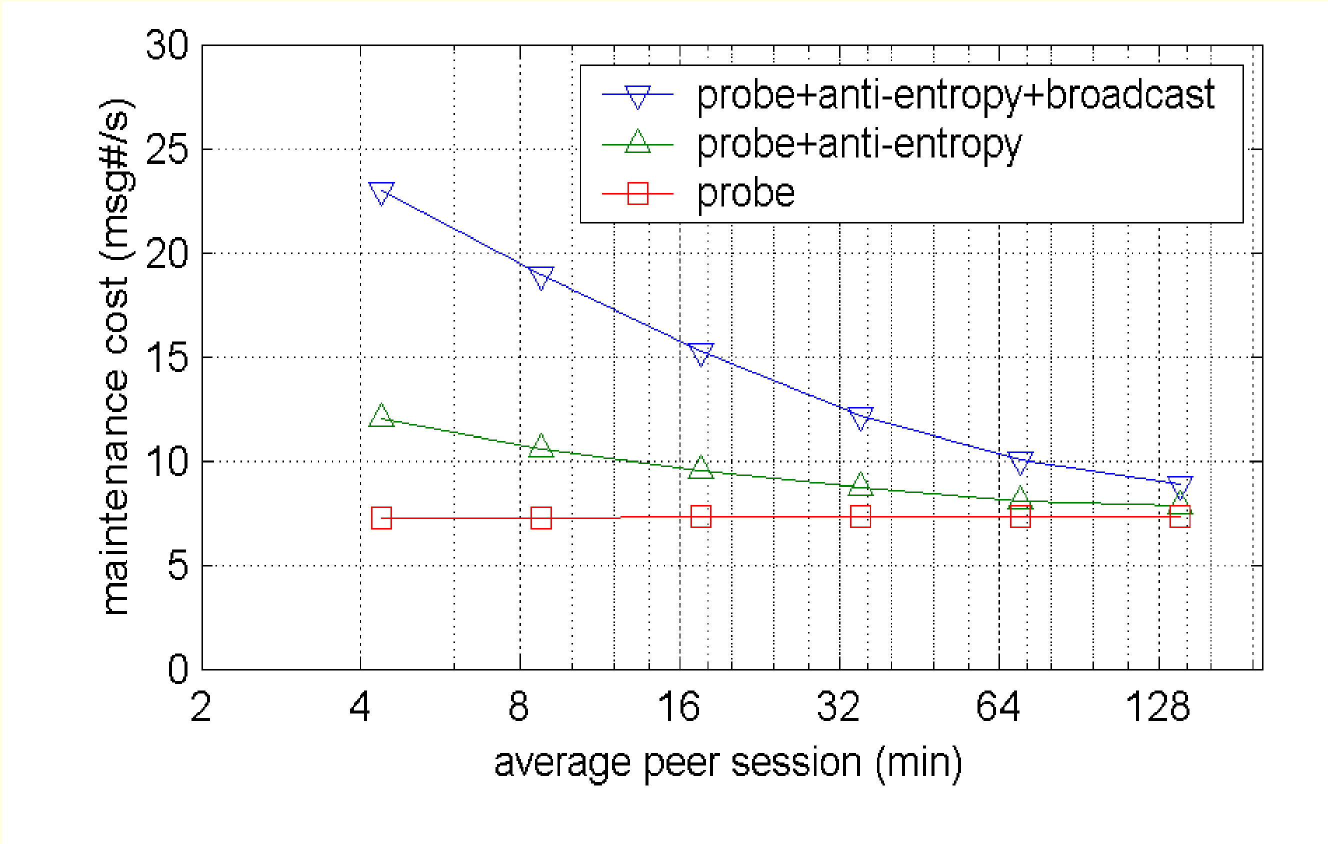




Routing hops success rate

Each point takes average on 1000 random routings (G=256)

Experiment: maintenance cost



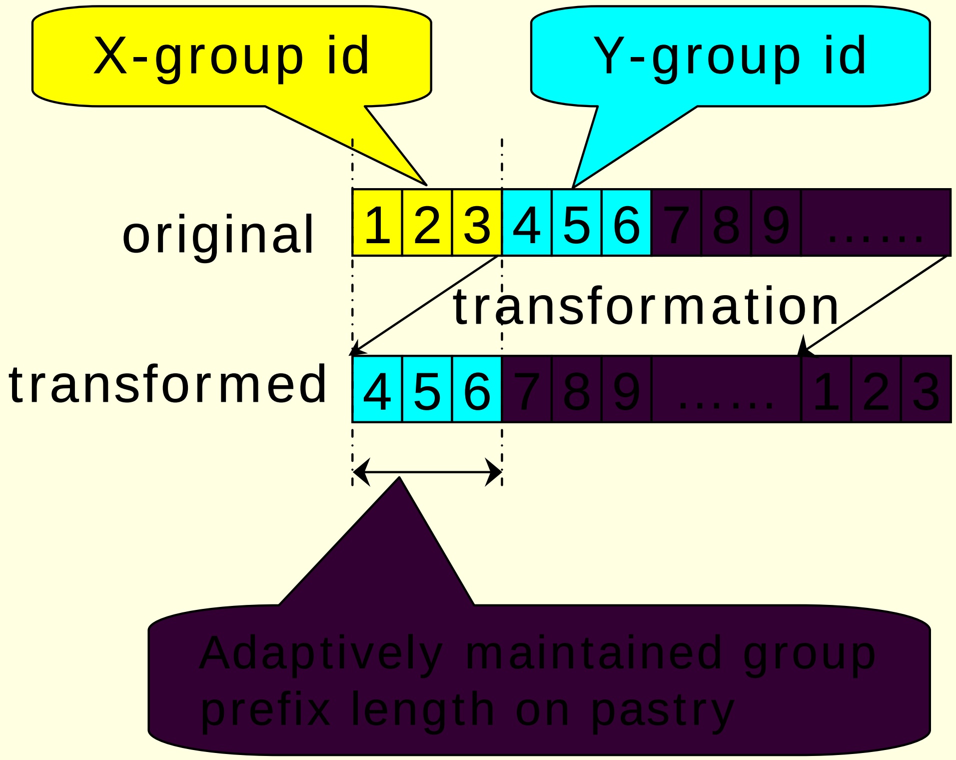
When average peer session = 30min,
total net I/O cost is 1.3KB/s for G=256 (in figure)
total net I/O cost is 10KB/s for G=4096 (predicted)

Outline

- Prefix routing: the fast one vs. the “slow” one
- Cost efficient membership protocol
- Accelerate with X-group routing
- Accelerate with Y-group routing
- Y-group routing failures and average performance
- Adaptation and group mngt.
- Various and larger scales
- Experiment results

Various and larger scales

- Various scales
  - Full scaled at G^2 (G: group size)
  - Scale < G^2 (e.g. scale=$G^{5/3}$)
- Larger scale
  - Add another Pastry by another key transformation



Various and larger scales

- Various scales
  - Full scaled at G^2 (G: group size)
  - Scale < G^2 (e.g. scale=$G^{5/3}$)
- Larger scale
  - Add another Pastry by another key transformation