Peer-to-Peer Networks

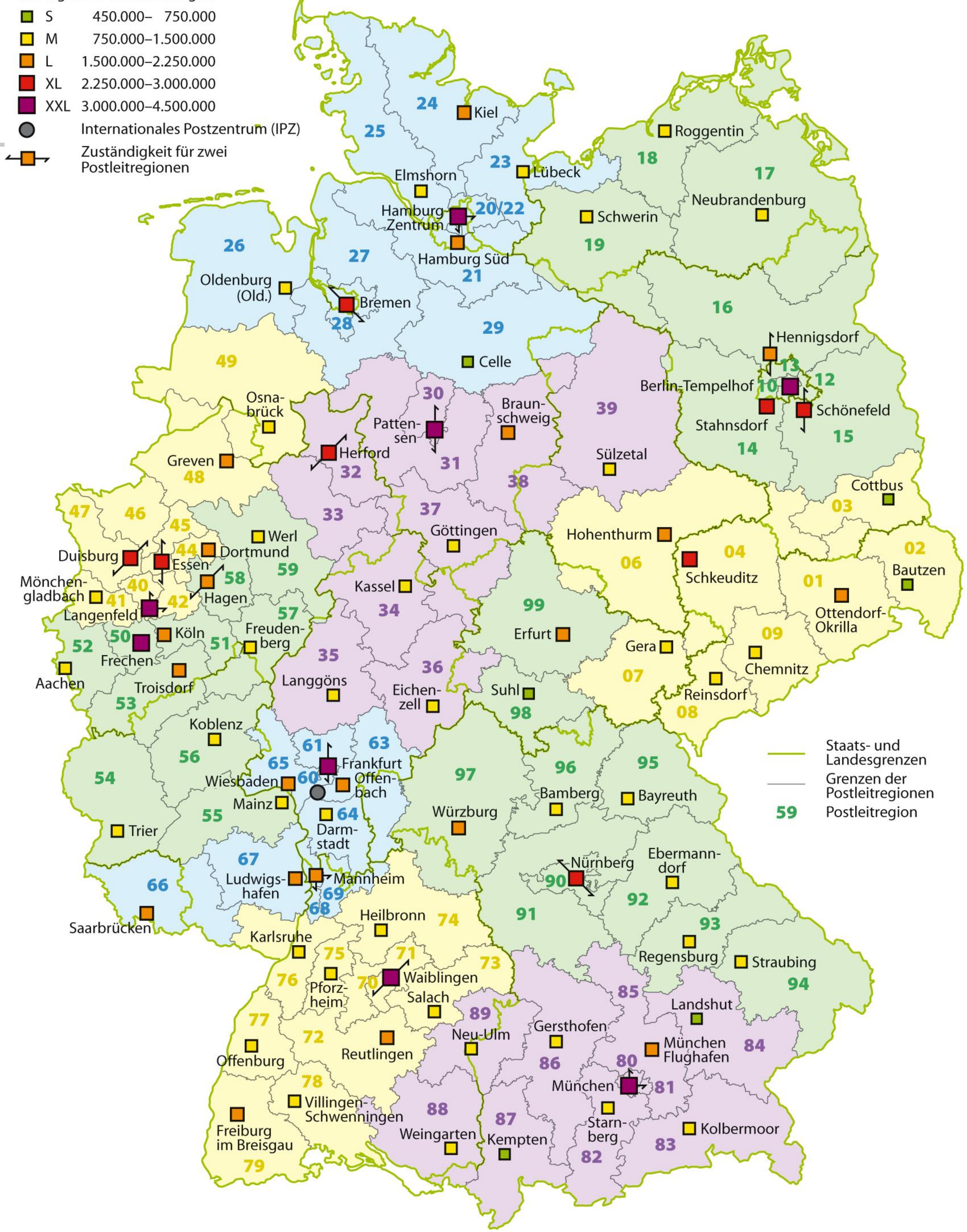
05 Pastry

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

Pastry

- Peter Druschel
  - Rice University, Houston, Texas
  - now head of Max-Planck-Institute for Computer Science, Saarbrücken/Kaiserslautern
- Antony Rowstron
  - Microsoft Research, Cambridge, GB
- Developed in Cambridge (Microsoft Research)
- Pastry
  - Scalable, decentralized object location and routing for large scale peer-to-peer-network
- PAST
  - A large-scale, persistent peer-to-peer storage utility
- Two names one P2P network
  - PAST is an application for Pastry enabling the full P2P data storage functionality
  - We concentrate on Pastry

Briefzentren der Deutschen Post AG



Pastry Overview

- Each peer has a 128-bit ID: nodeID
  - unique and uniformly distributed
  - e.g. use cryptographic function applied to IP-address
- Routing
  - Keys are matched to $\{0,1\}^{128}$
  - According to a metric messages are distributed to the neighbor next to the target
- Routing table has $O(2^b(\log n)/b) + \ell$ entries
  - n: number of peers
  - ℓ: configuration parameter
  - b: word length
    - typical: b= 4 (base 16), $\ell = 16$
    - message delivery is guaranteed as long as less than $\ell/2$ neighbored peers fail
- Inserting a peer and finding a key needs $O((\log n)/b)$ messages

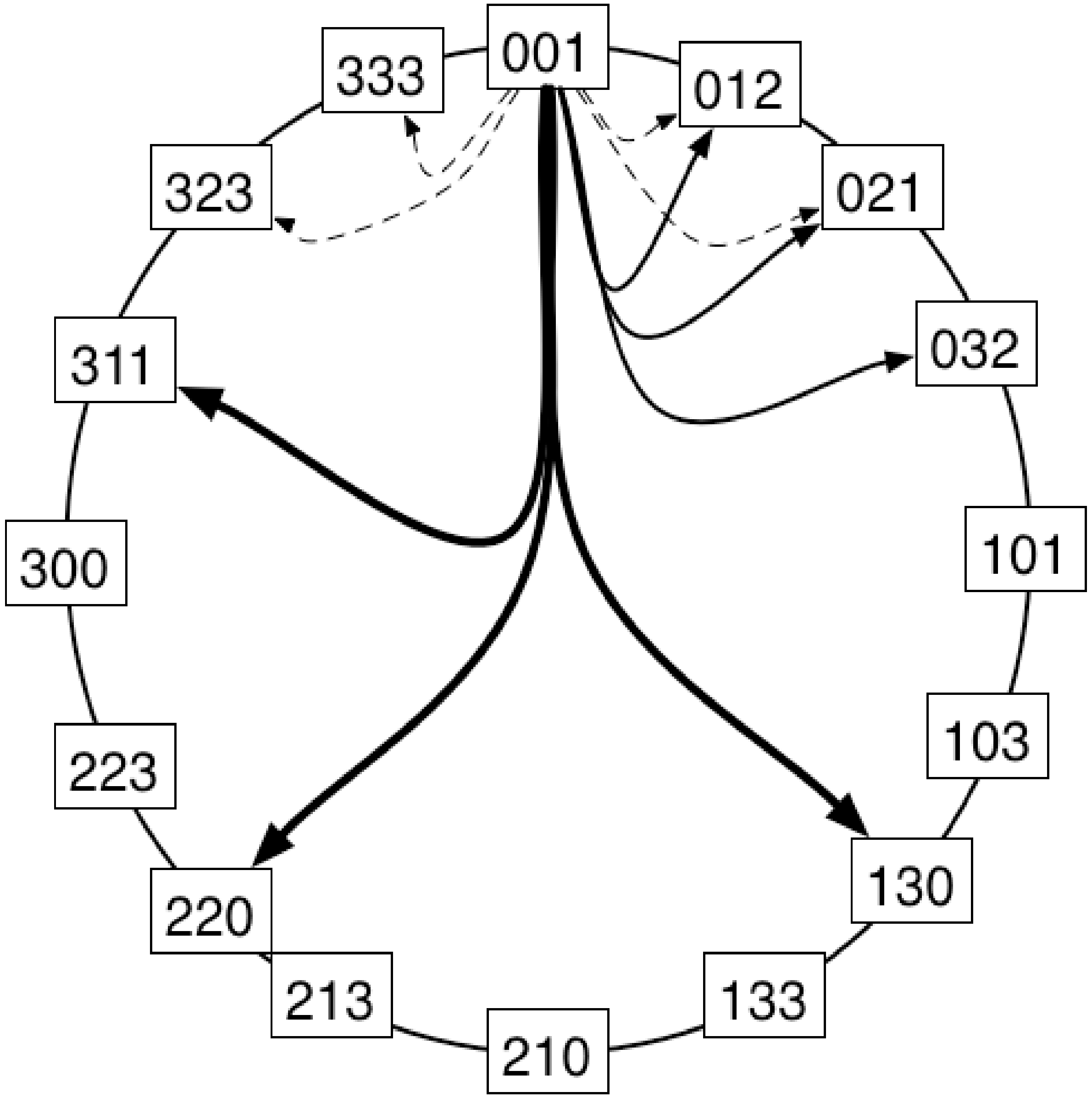
Routing Table

- NodeId presented in base 2^b
  - e.g. NodeID: 65A0BA13
- For each prefix p and letter x ∈ {0,..,2^b-1} add an peer of form px* to the routing table of NodeID, e.g.
  - b=4, 2^b=16
  - 15 entries for 0*,1*, .. F*
  - 15 entries for 60*, 61*,... 6F*
  - ...
  - if no peer of the form exists, then the entry remains empty
- Choose next neighbor according to a distance metric
  - metric results from the RTT (round trip time)
- In addition choose ℓ neighbors
  - ℓ/2 with next higher ID
  - ℓ/2 with next lower ID

0x	*1x*	*2x*	*3x*	*4x*	*5x*		*7x*	*8x*	*9x*	*ax*	*bx*	*cx*	*dx*	*ex*	*fx*
60x	*61x*	*62x*	*63x*	*64x*		*66x*	*67x*	*68x*	*69x*	*6ax*	*6bx*	*6cx*	*6dx*	*6ex*	*6fx*
650x	*651x*	*652x*	*653x*	*654x*	*655x*	*656x*	*657x*	*658x*	*659x*		*65bx*	*65cx*	*65dx*	*65ex*	*65fx*
65a0x		*65a2x*	*65a3x*	*65a4x*	*65a5x*	*65a6x*	*65a7x*	*65a8x*	*65a9x*	*65aax*	*65abx*	*65acx*	*65adx*	*65aex*	*65afx*

Routing Table

- Example b=2
- Routing Table
  - For each prefix p and letter x ∈ {0,..,2^b-1} add an peer of form px* to the routing table of NodeID
- In addition choose ℓ neighbors
  - ℓ/2 with next higher ID
  - ℓ/2 with next lower ID
- Observation
  - The leaf-set alone can be used to find a target
- Theorem
  - With high probability there are at most O(2^b (log n)/b) entries in each routing table



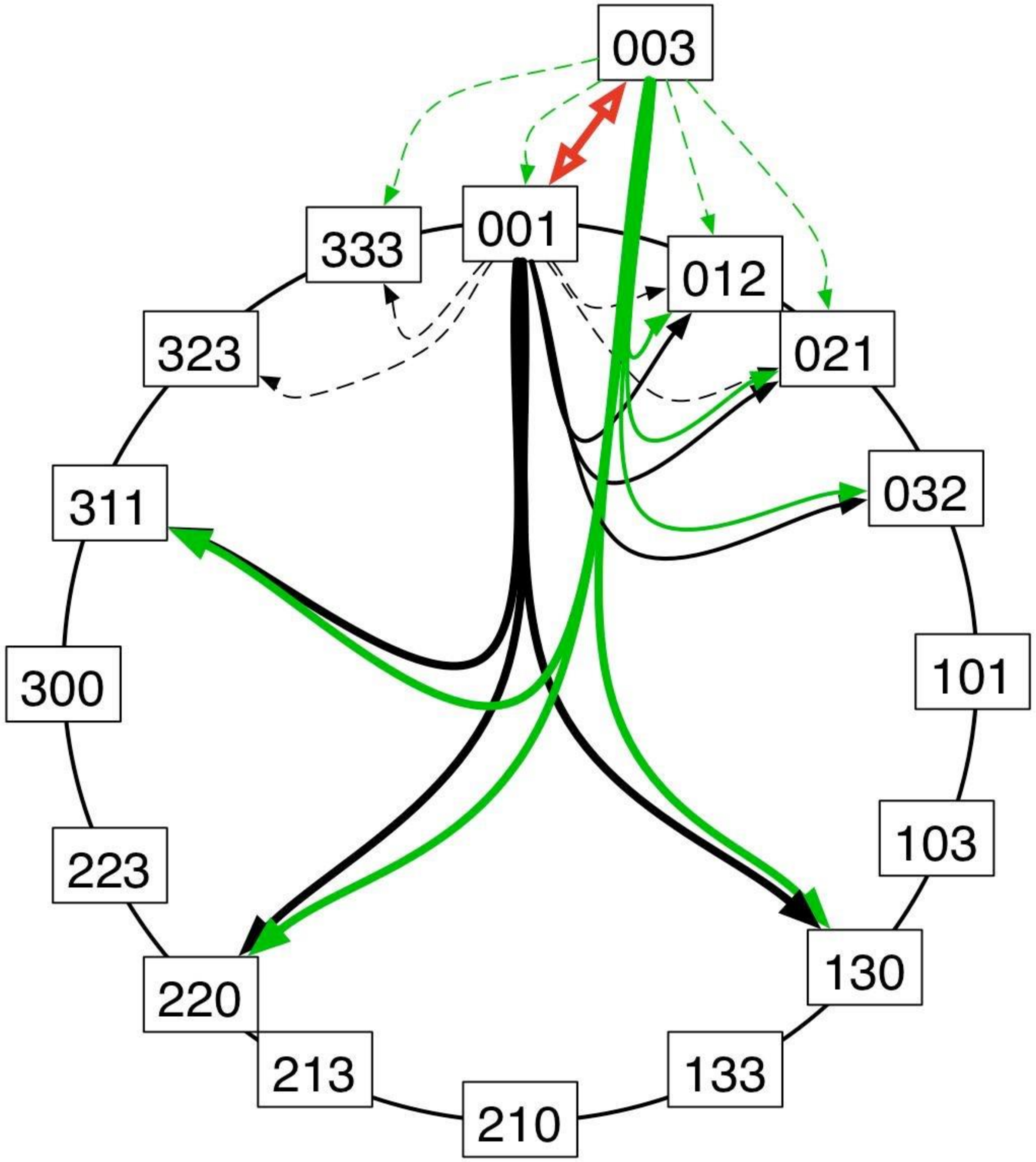
Routing Table

- Theorem
  - With high probability there are at most $O(2^b (\log n)/b)$ entries in each routing table
- Proof
  - The probability that a peer gets the same m-digit prefix is $$2^{-bm}$$
  - The probability that a m-digit prefix is unused is $$(1-2^{-bm})^n \leq e^{-n/2^{bm}}$$
  - For m=c (log n)/b we get $$e^{-n/2^{bm}} \leq e^{-n/2^{c \log n}} \leq e^{-n/n^c} \leq e^{-n^{c-1}}$$
  - With (extremely) high probability there is no peer with the same prefix of length (1+ε)(log n)/b
  - Hence we have (1+ε)(log n)/b rows with 2^b-1 entries each

0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
650x	651x	652x	653x	654x	655x	656x	657x	658x	659x		65bx	65cx	65dx	65ex	65fx
65a0x		65a2x	65a3x	65a4x	65a5x	65a6x	65a7x	65a8x	65a9x	65aax	65abx	65acx	65adx	65aex	65afx

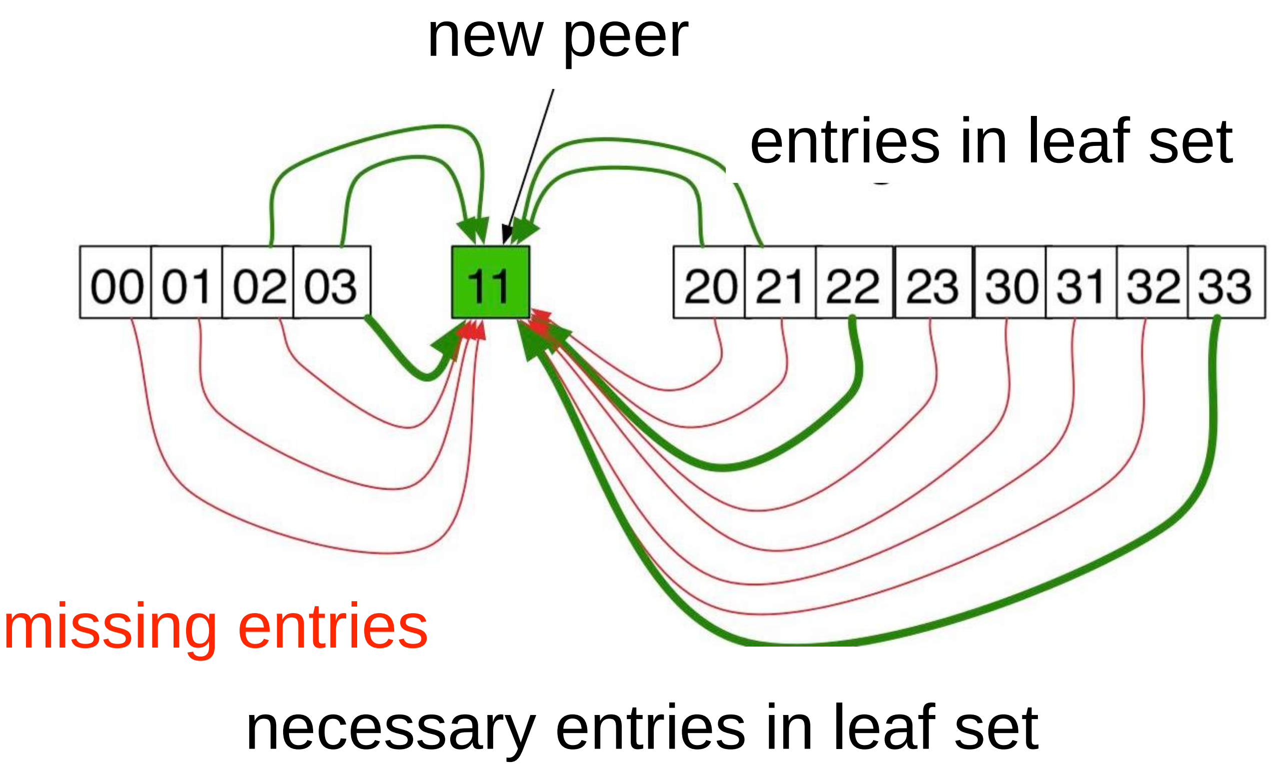
A Peer Enters

- New node x sends message to the node z with the longest common prefix p
- x receives
  - routing table of z
  - leaf set of z
- z updates leaf-set
- x informs informiert ℓ-leaf set
- x informs peers in routing table
  - with same prefix p (if $\ell/2 < 2^b$)
- Numbor of messages for adding a peer
  - ℓ messages to the leaf-set
  - expected (2^b - $\ell/2$) messages to node with common prefix
  - one message to z with answer



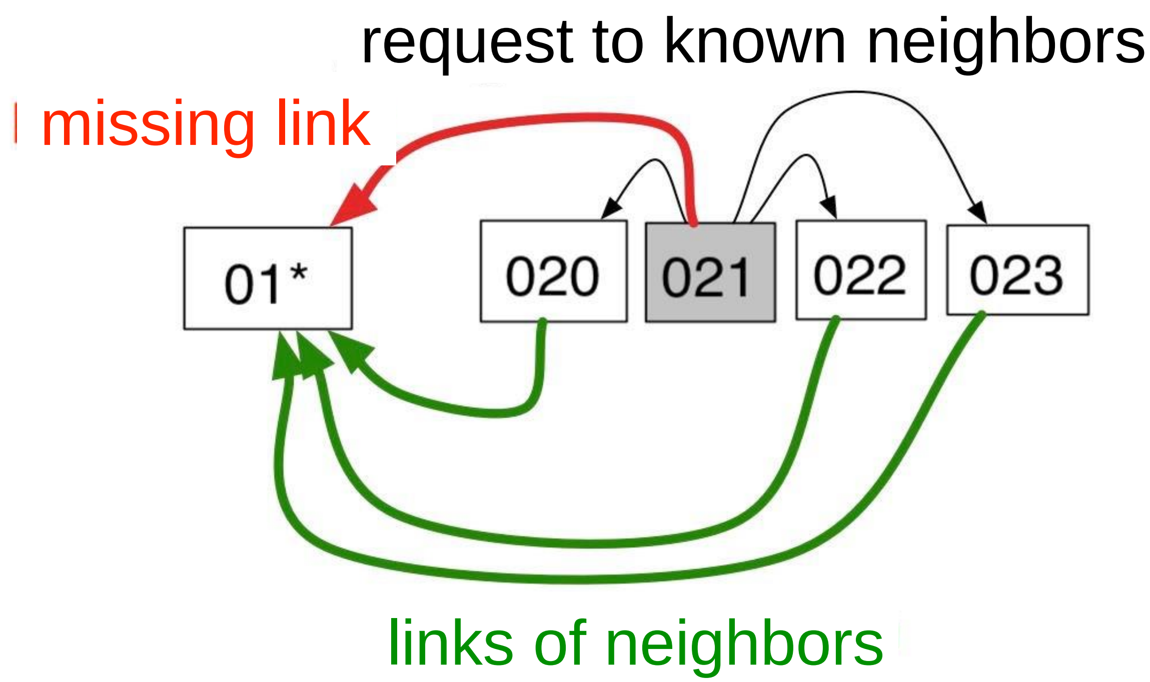
When the Entry-Operation Errs

- Inheriting the next neighbor routing table does not allows work perfectly
- Example
  - If no peer with 1* exists then all other peers have to point to the new node
  - Inserting 11
  - 03 knows from its routing table
    - 22,33
    - 00,01,02
  - 02 knows from the leaf-set
    - 01,02,20,21
- 11 cannot add all necessary links to the routing tables



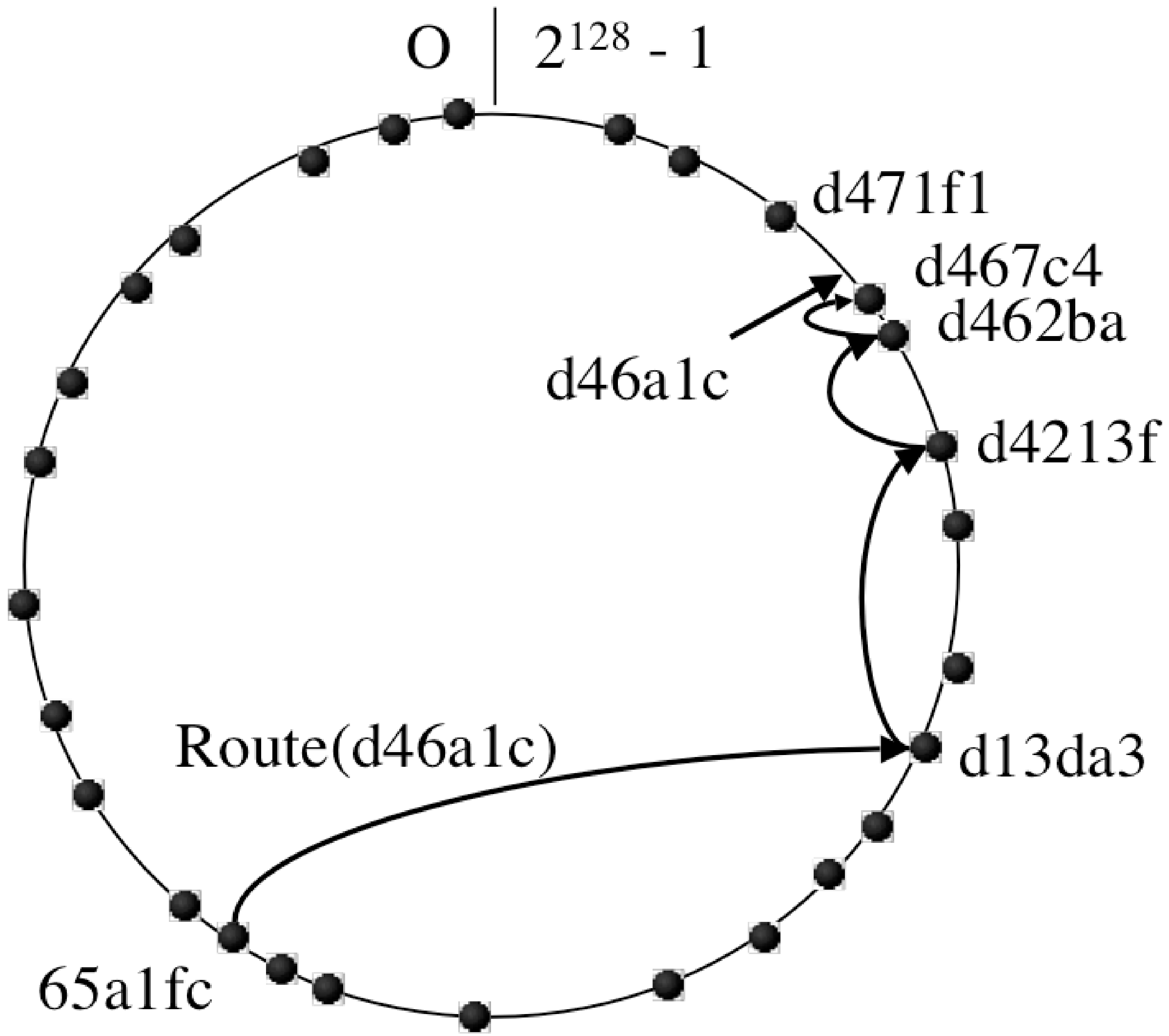
Missing Entries in the Routing Table

- Assume the entry R_i^j is missing at peer D
  - j-th row and i-th column of the routing table
- This is noticed if message of a peer with such a prefix is received
- This may also happen if a peer leaves the network
- Contact peers in the same row
  - if they know a peer this address is copied
- If this fails then perform routing to the missing link



Lookup

- Compute the target ID using the hash function
- If the address is within the ℓ-leaf set
  - the message is sent directly
  - or it discovers that the target is missing
- Else use the address in the routing table to forward the mesage
- If this fails take best fit from all addresses



Lookup in Detail

- L: ℓ-leafset
- R: routing table
- M: nodes in the vicinity of D (according to RTT)
- D: key
- A: nodeID of current peer
- R^i_l: j-th row and i-th column of the routing table
- L_i: numbering of the leaf set
- D_i: i-th digit of key D
- shl(A): length of the larges common prefix of A and D (shared header length)

```
if (L_{-⌊|L|/2⌋} ≤ D ≤ L_{⌊|L|/2⌋}) {
    // D is within range of our leaf set
    forward to L_i, s.th. |D − L_i| is minimal;
} else {
    // use the routing table
    Let l = shl(D, A);
    if (R_l^{D_l} ≠ null) {
        forward to R_l^{D_l};
    }
    else {
        // rare case
        forward to T ∈ L ∪ R ∪ M, s.th.
            shl(T, D) ≥ l,
            |T − D| < |A − D|
    }
}
```

Routing — Discussion

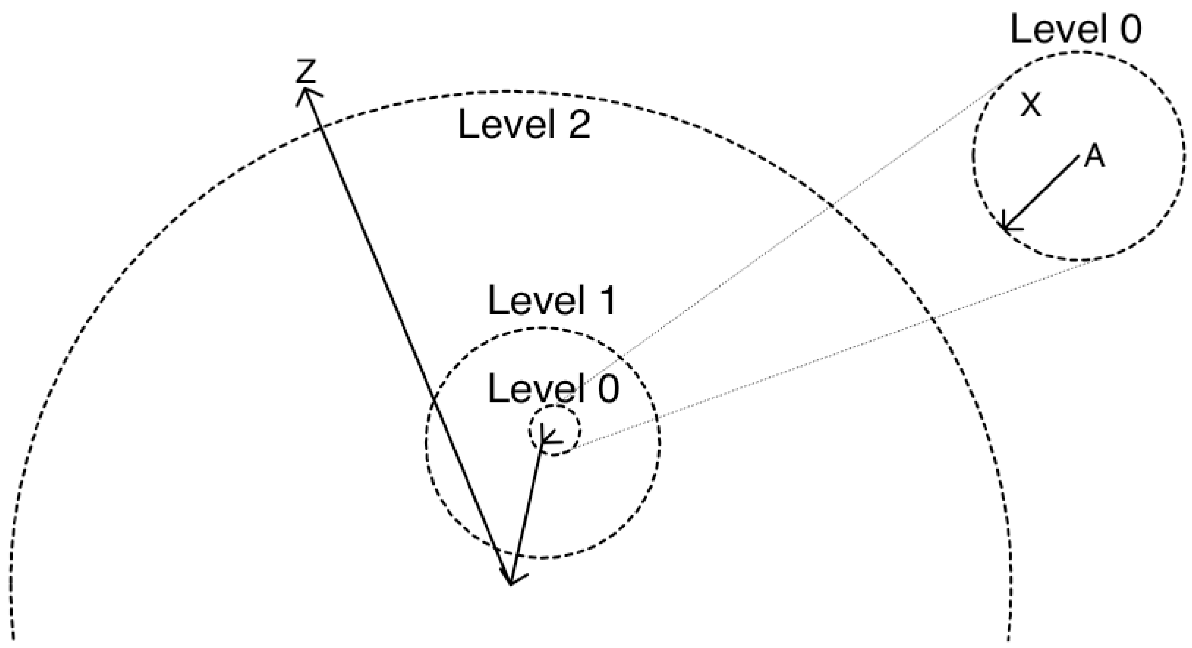
- If the Routing-Table is correct
  - routing needs O((log n)/b) messages
- As long as the leaf-set is correct
  - routing needs O(n/l) messages
  - unrealistic worst case since even damaged routing tables allow dramatic speedup
- Routing does not use the real distances
  - M is used only if errors in the routing table occur
  - using locality improvements are possible
- Thus, Pastry uses heuristics for improving the lookup time
  - these are applied to the last, most expensive, hops

Localization of the k Nearest Peers

- Leaf-set peers are not near, e.g.
  - New Zealand, California, India, ...
- TCP protocol measures latency
  - latencies (RTT) can define a metric
  - this forms the foundation for finding the nearest peers
- All methods of Pastry are based on heuristics
  - i.e. no rigorous (mathematical) proof of efficiency
- Assumption: metric is Euclidean

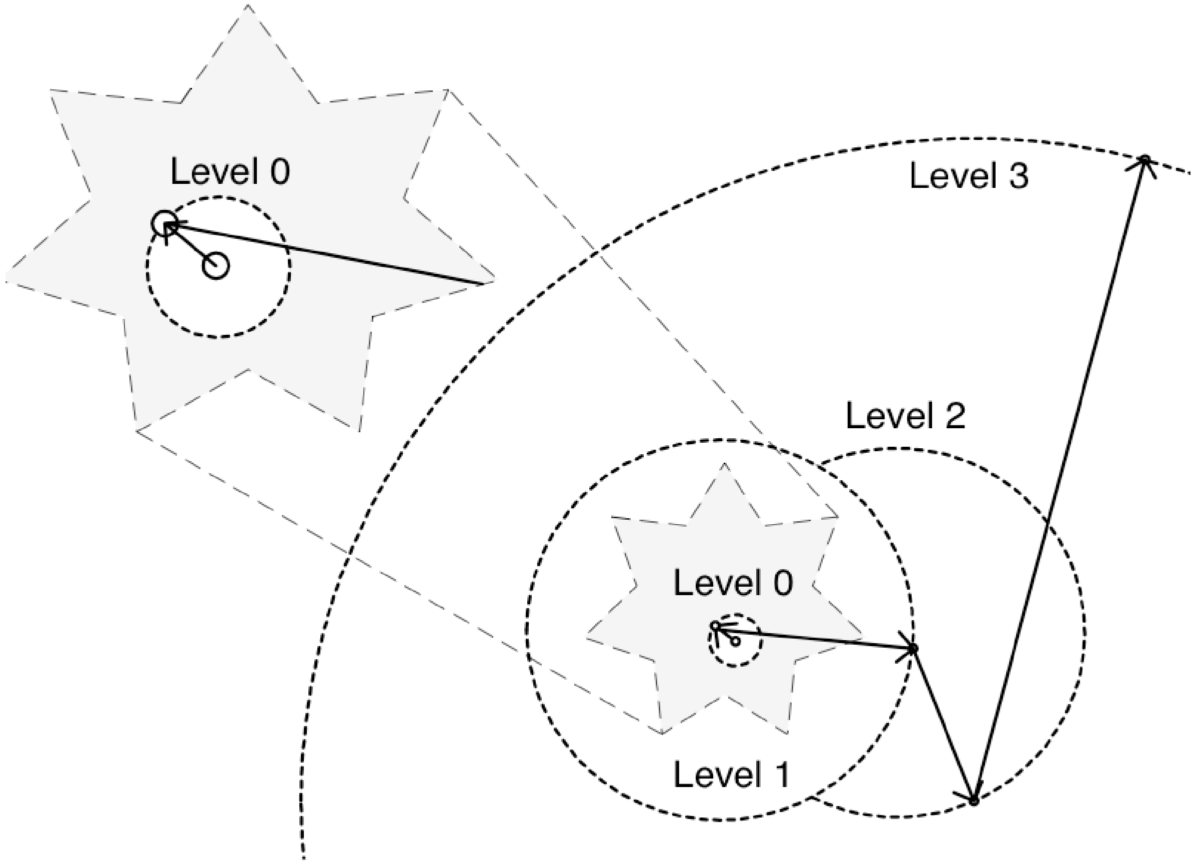
Locality in the Routing Table

- Assumption
  - When a peer is inserted the peers contacts a near peer
  - All peers have optimized routing tables
- But:
  - The first contact is not necessary near according to the node-ID
- 1st step
  - Copy entries of the first row of the routing table of P
    - good approximation because of the triangle inequality (metric)
- 2nd step
  - Contact fitting peer p‘ of p with the same first letter
  - Again the entries are relatively close
- Repeat these steps until all entries are updated



Locality in the Routing Table

- In the best case
  - each entry in the routing table is optimal w.r.t. distance metric
  - this does not lead to the shortest path
- There is hope for short lookup times
  - with the length of the common prefix the latency metric grows exponentially
  - the last hops are the most expensive ones
  - here the leaf-set entries help

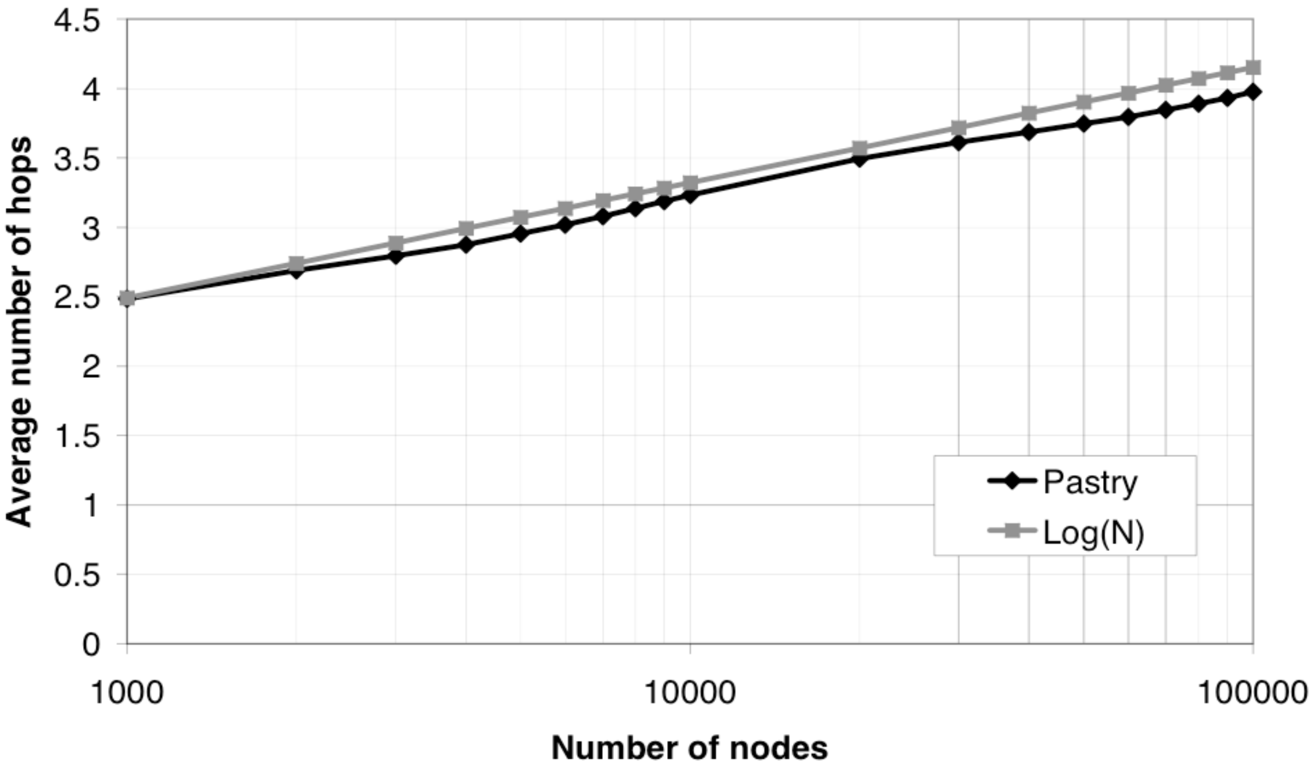


Localization of Near Nodes

- Node-ID metric and latency metric are not compatible
- If data is replicated on k peers then peers with similar Node-ID might be missed
- Here, a heuristic is used
- Experiments validate this approach

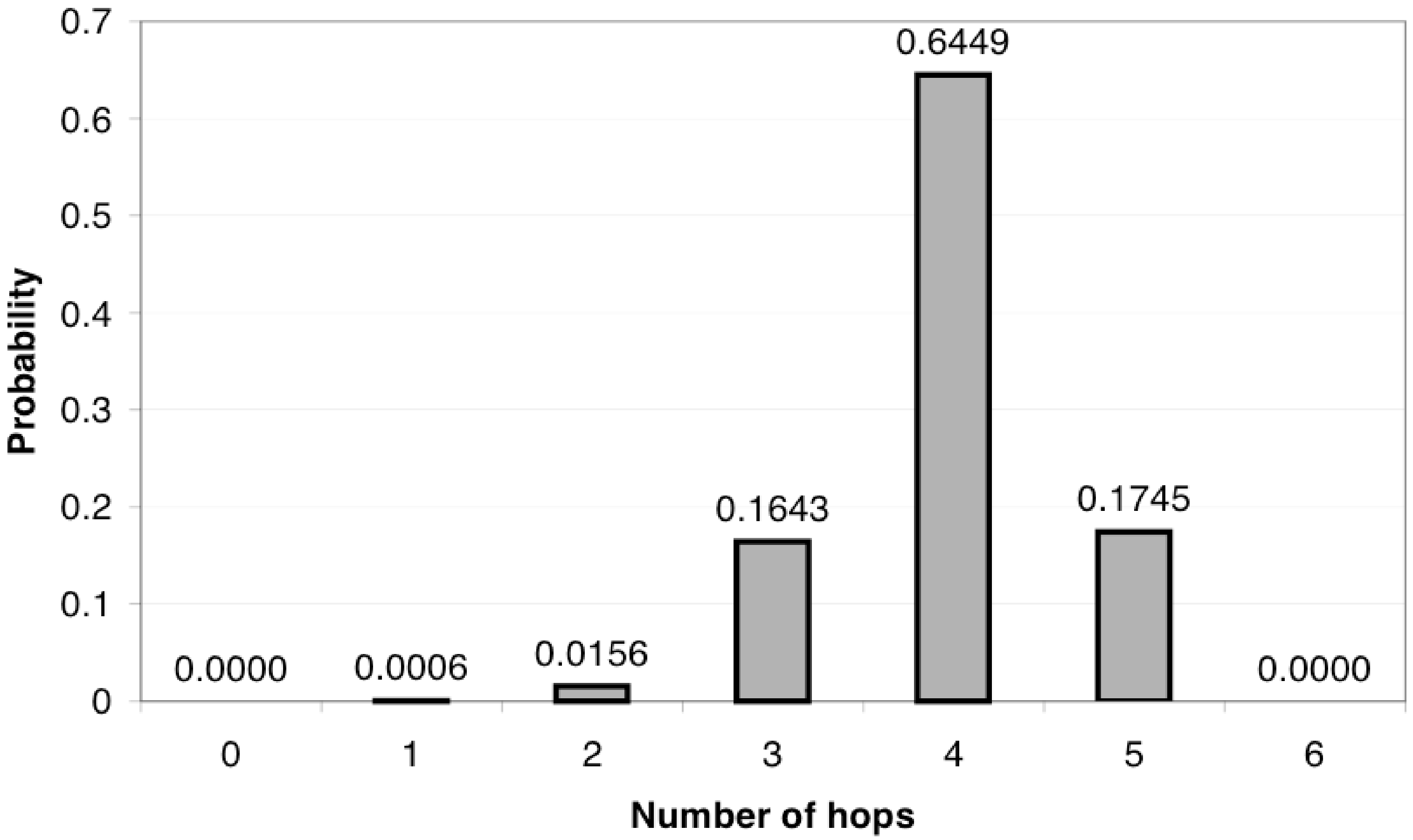
Experimental Results — Scalability

- Parameter b=4, l=16, M=32
- In this experiment the hop distance grows logarithmically with the number of nodes
- The analysis predicts O(log n)
- Fits well



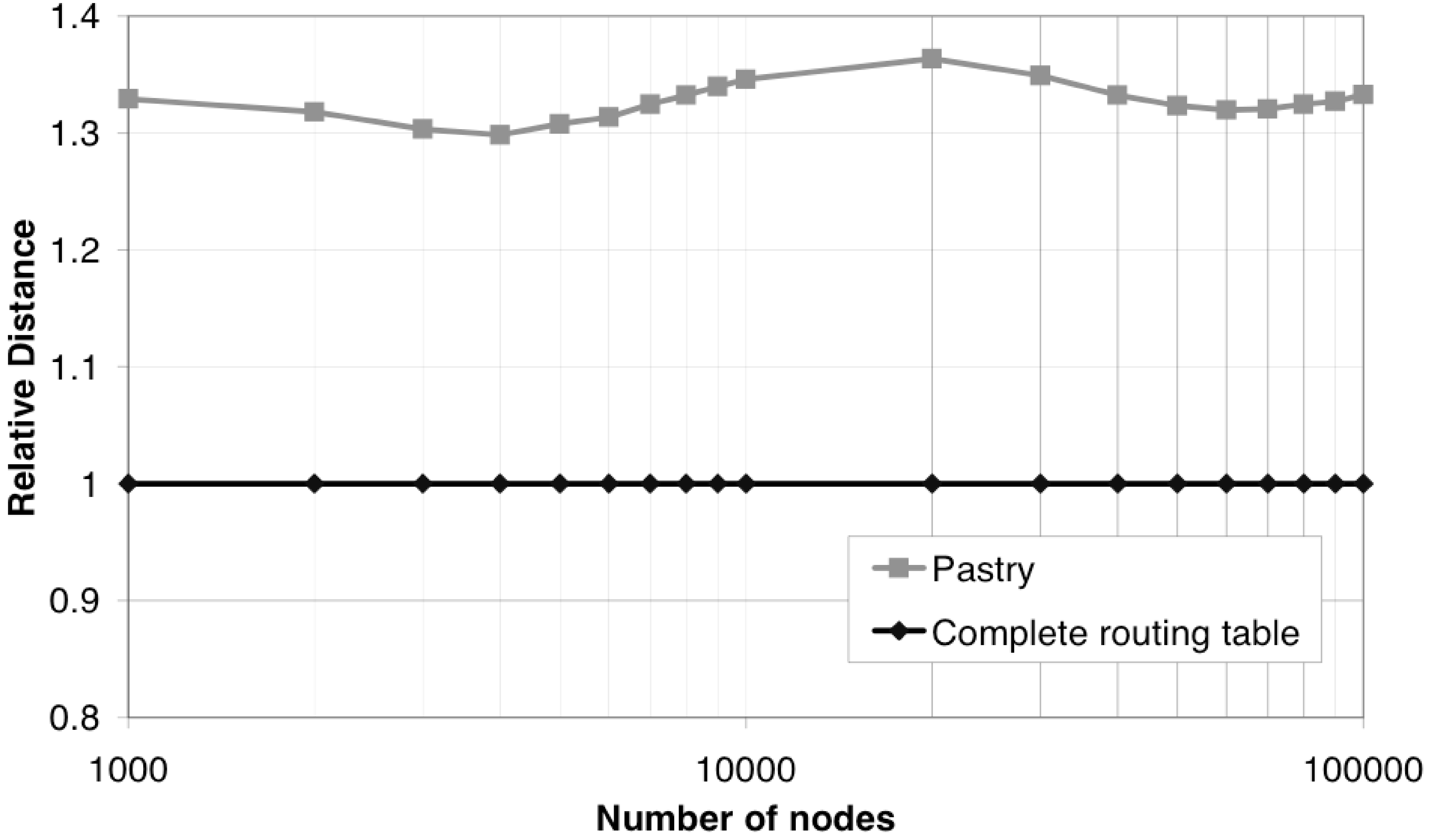
Experimental Results
Distribution of Hops

- Parameter b=4, l=16, M=32, n = 100,000
- Result
  - deviation from the expected hop distance is extremely small
- Analysis predicts difference with extremely small probability
  - fits well



Experimental Results — Latency

- Parameter b=4, l=16, M=3
- Compared to the shortest path astonishingly small
  - seems to be constant



Interpreting the Experiments

- Experiments were performed in a well-behaving simulation environment
- With b=4, L=16 the number of links is quite large
  - The factor $2^b/b = 4$ influences the experiment
  - Example n= 100 000
    - $2^b/b \log n = 4 \log n > 60$ links in routing table
    - In addition we have 16 links in the leaf-set and 32 in M
- Compared to other protocols like Chord the degree is rather large
- Assumption of Euclidean metric is rather arbitrary

Peer-to-Peer Networks

05 Pastry

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