Peer-to-Peer-Netzwerke

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Christian Schindelhauer

Sommersemester 2006

12. Vorlesung

14.06.2006

schindel@informatik.uni-freiburg.de

Inhalte

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Kurze Geschichte der Peer-to-Peer-Netzwerke**
- **Das Internet: Unter dem Overlay**
- **Die ersten Peer-to-Peer-Netzwerke**
  - Napster
  - Gnutella
- **CAN**
- **Chord**
- **Pastry und Tapestry**
- **Gradoptimierte Netzwerke**
  - Viceroy
  - Distance-Halving
  - Koorde
- **Netzwerke mit Suchbäumen**
  - Skipnet und Skip-Graphs
  - P-Grid
- **Selbstorganisation**
  - Pareto-Netzwerke
  - Zufallsnetzwerke
  - Metrikbasierte Netzwerke
- **Sicherheit in Peer-to-Peer-Netzwerken**
- **Anonymität**
- **Datenzugriff: Der schnellere Download**
- **Peer-to-Peer-Netzwerke in der Praxis**
  - eDonkey
  - FastTrack
  - Bittorrent
- **Peer-to-Peer-Verkehr**
- **Juristische Situation**

Überblick

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Jeder Peer hat 128-bit ID: nodeID**
  - Eindeutig und gleichmäßig verteilt
  - z.B. durch Kryptographische Funktion angewendet auf IP-Adresse
- **Routing**
  - Die Schlüssel werden auf $\{0,1\}^{128}$ abgebildet
  - Gemäß einer Metrik werden Nachrichten zum nächstgelegenden Nachbar weitergereicht
- **Routing tabelle hat $O(2^b (\log n)/b) + \ell$ Einträge**
  - n: Anzahl Peers
  - b,ℓ: Konfigurationsparameter, b: Wortlänge
    - typisch: b= 4 (Basis 16), $\ell = 16$
    - Auslieferung ist garantiert, falls nicht mehr al $\ell/2$ Knoten ausfallen
- **Einfügen von Peers benötigt $O((\log n)/b)$ Nachrichten**

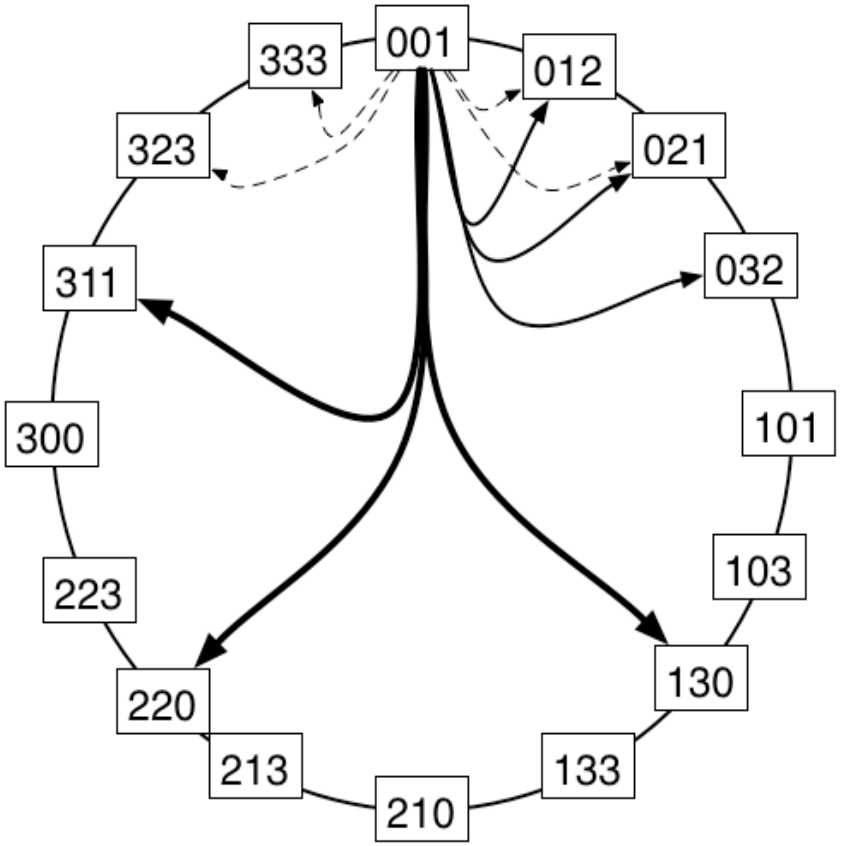
Routing-Tabelle

- **NodeId wird zur Basis 2^b dargestellt,**
  - z.B. NodeID: 65A0BA13
- **Für jeden Präfix p und Buchstaben x ∈ {0,..,2^b-1} von NodeID wird ein Repräsentant der Form px* eingetragen**
  - Beispiel b=4, damit ist 2^b=16
  - also 15 Einträge für 0*,1*, .. F*
  - 15 Einträge für 60*, 61*,... 6F*
  - ...
  - Existiert kein Repräsentant wird nichts eingetragen
- **Bezüglich einer Metrik wird immer der nächstgelegene Repräsentant gewählt**
  - Die Metrik resultiert auf den wechselseitigen Latenzzeiten zwischen den Knoten
- **Zusätzlich werden ℓ Nachbarn gemäß der NodeID gespeichert**
  - ℓ/2 mit nächst größerer ID und
  - ℓ/2 mit nächst kleinerer ID

0	1	2	3	4	5		7	8	9	a	b	c	d	e	f
x	x	x	x	x	x		x	x	x	x	x	x	x	x	x
6	6	6	6	6		6	6	6	6	6	6	6	6	6	6
0	1	2	3	4		6	7	8	9	a	b	c	d	e	f
x	x	x	x	x		x	x	x	x	x	x	x	x	x	x
6	6	6	6	6	6	6	6	6	6		6	6	6	6	6
5	5	5	5	5	5	5	5	5	5		5	5	5	5	5
0	1	2	3	4	5	6	7	8	9		b	c	d	e	f
x	x	x	x	x	x	x	x	x	x		x	x	x	x	x
6		6	6	6	6	6	6	6	6	6	6	6	6	6	6
5		5	5	5	5	5	5	5	5	5	5	5	5	5	5
a		a	a	a	a	a	a	a	a	a	a	a	a	a	a
0		2	3	4	5	6	7	8	9	a	b	c	d	e	f
x		x	x	x	x	x	x	x	x	x	x	x	x	x	x

Routing-Tabelle

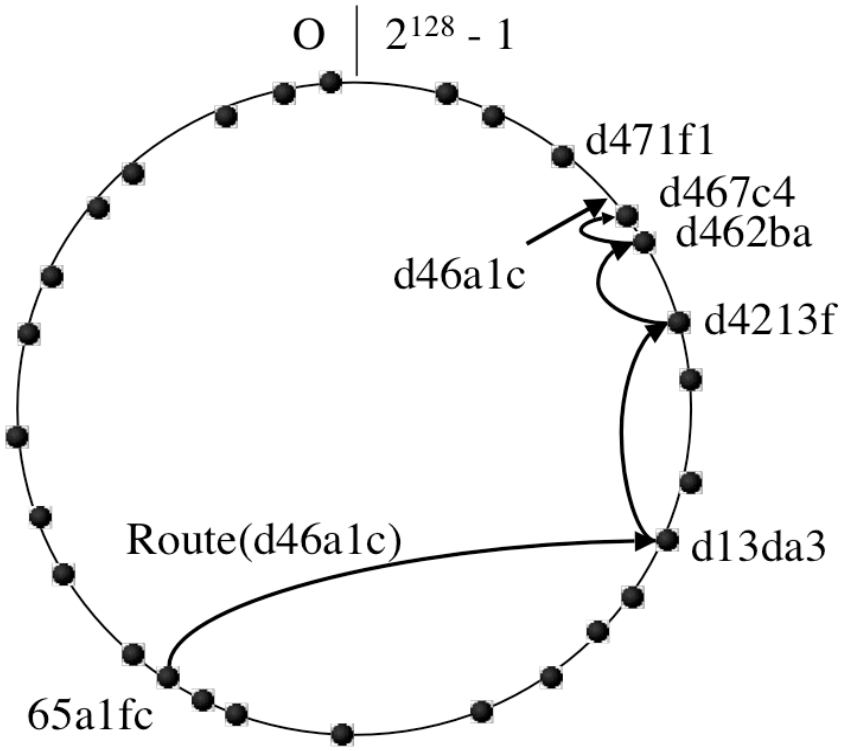
- **Beispiel: b=2**
- ***routing table*:**
  Für jeden Präfix p und Buchstaben x ∈ {0,..,2b-1} von NodeID wird ein Repräsentant der Form px* eingetragen
- ***leaf set*:**
  Zusätzlich werden ℓ Nachbarn gemäß der NodeID gespeichert
  - $\ell/2$ mit nächst größerer ID und
  - $\ell/2$ mit nächst kleinerer ID
- **Beobachtung:**
  - Allein durch die Leaf-Set werden die Zielknoten immer gefunden
- **Lemma**
  - Mit hoher Wahrscheinlichkeit sind höchstens $O(2^b (\log n)/b)$ Einträge in jeder Routing-Table



Routing - im Prinzip

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Zuerst wird durch Hash-Funktion die Zieladresse hergestellt**
- **Befindet sich die Zieladresse innerhalb der ℓ-Nachbarschaft,**
  - wird direkt dorthin ausgeliefert
  - oder das Fehlen des wird Ziel-Peers festgestellt
- **Ansonsten wird in der Routingtabelle die Adresse mit gemeinsame Präfix gesucht und**
- **Dorthin wird die Nachricht weitergeleitet**
- **Zusätzlich wird eine Menge M von nahen Knoten unterhalten, diese werden hinsichtlich der Latenzzeit im Internet optimiert**



Routing im Detail

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **L:** ℓ-Nachbarschaft
- **R:** Routing-Table
- **M:** Knoten in der Nähe von D (gemäß Latenzzeit)
- **D:** Schlüssel
- **A:** Node-ID des aktuellen Peers
- **R^i_ℓ:** ℓ-te Zeile und i-te Spalte der Routing-Table
- **L_i:** Nummerierung der ℓ-Nachbarschaft
- **D_ℓ:** ℓ-te Ziffer des Schlüssels D
- **shl(A):** Länge des gemeinsamen Präfix von A und D in Ziffern

```
(1)  if (L_{-⌊|L|/2⌋} ≤ D ≤ L_{⌊|L|/2⌋}) {
(2)      // D is within range of our leaf set
(3)      forward to L_i, s.th. |D − L_i| is minimal;
(4)  } else {
(5)      // use the routing table
(6)      Let l = shl(D, A);
(7)      if (R_l^{D_l} ≠ null) {
(8)          forward to R_l^{D_l};
(9)      }
(10)     else {
(11)         // rare case
(12)         forward to T ∈ L ∪ R ∪ M, s.th.
(13)             shl(T, D) ≥ l,
(14)             |T − D| < |A − D|
(15)     }
(16) }
```

Routing - Diskussion

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Falls Routing-Table korrekt ist,**
  - benötigt Routing O((log n)/b) Nachrichten
- **Solange Leaf-Set korrekt ist,**
  - benötigt Routing O(n/ℓ) Nachrichten
  - Tatsächlich ist es aber wesentlich schneller, da selbst defekte Routing-Table erhebliche Beschleunigung bringt
- **Routing berücksichtigt nicht die wahren Abstände**
  - M wird nur benutzt, falls Fehler in der Routing-Tabelle vorliegen
  - Durch Ausnutzung der Lokalität, Verbesserungen möglich
- **Daher verwendet Pastry Heuristiken zur Verbesserung der Latenzzeit**
  - Diese werden in den besonders teuren letzten Hops angewendet

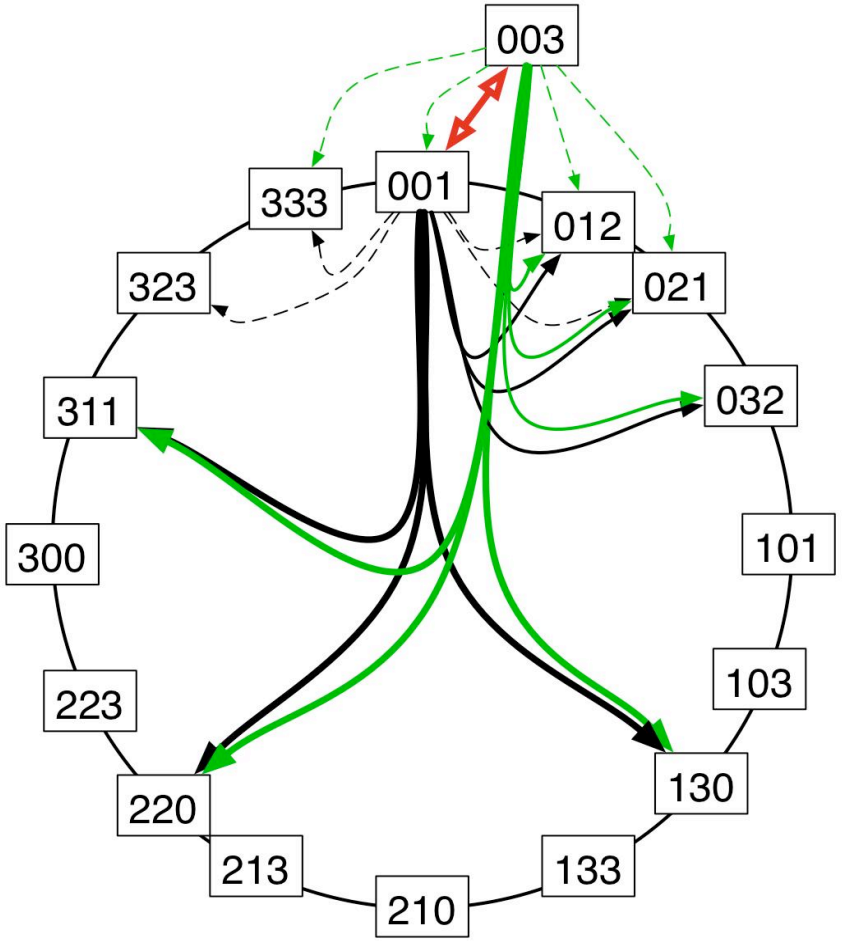
Einfügen Peers

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Neuer Knoten X sendet Nachricht Knoten Z mit längsten gemeinsamen Präfix p**
- **X erhält**
  - Routingtabelle von Z
  - Nachbarschaftsmenge von Z
- **Z aktualisiert Nachbarschaftsmenge**
- **X informiert ℓ-Nachbarschaft**
- **X informiert Peers in Routing-Table**
  - mit gleichen Präfix p außerhalb der ℓ-Nachbarschaft (falls $\ell/2 < 2^b$)

Aufwand für Einfüge-Operation:

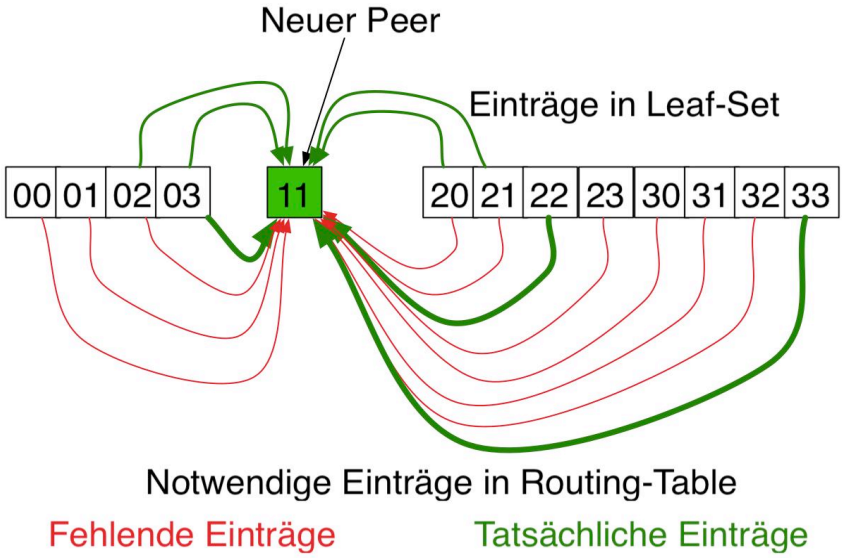
- ℓ Nachrichten an Nachbarschaft
- Erwartet (2^b - ℓ /2) Nachrichten an Knoten mit gemeinsamen Präfix
- Eine Nachricht an Knoten Z mit Antwort



Wenn das Einfügen versagt

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Die Übernahme der Nachbarschaftsmenge von nächstgelegenen Peer reicht im allgemeinen nicht**
- **Beispiel:**
  - Falls kein Peer mit 1* vorhanden ist, müssen alle anderen Peers auf den neuen Knoten zeigen
  - Einfügen von 11:
  - 03 kennt aus Routing-Table
    - 22,33
    - 00,01,02
  - 02 kennt aus Leaf-Set
    - 01,02,20,21
- **11 kann nicht alle notwendigen Links veranlassen**



Fehlender Eintrag in Routing-Table

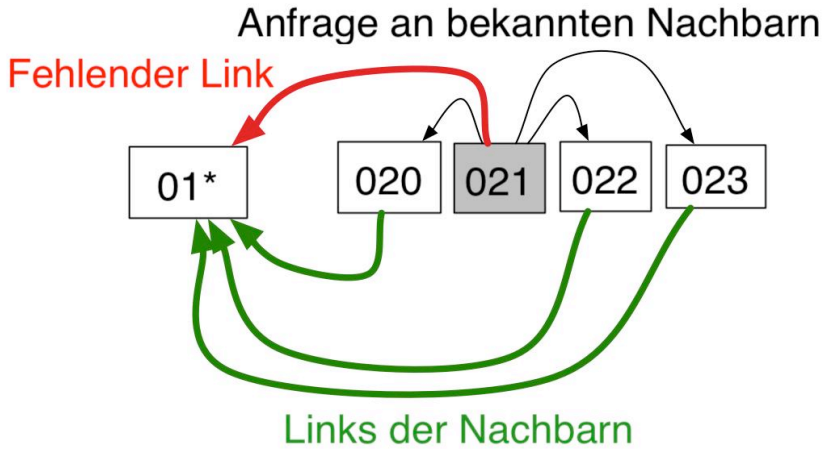
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Annahme:

- **Tabelleneintrag R^i_ℓ fehlt in Peer D**
  - ℓ-te Zeile und i-te Spalte der Routing-Table
- **Wird festgestellt sobald eine Nachricht von einem Peer mit diesem Präfix ankommt**
- **Kommt auch vor, falls ein Peer das Netzwerk verlässt**

- **Kontaktiere Peers in der selben Zeile**
- **Falls Peer bekannt ist, wird Adresse kopiert**

- **Falls dies scheitert, führe Routing zu dieser Adresse durch**



Lokalisierung der k nächsten Knoten

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Direkte Nachbarn im Peer-to-Peer-Netzwerk sind nicht Knoten in der Nähe**
  - z.B. Link von Neuseeland nach Südbaden
  - von Indien nach Kalifornien
- **Das TCP-Protokoll des Internets misst Latenzzeiten**
  - Damit kann eine Metrik definiert werden
  - Diese ist die Grundlage zur Suche nach den nächsten Peers
- **Damit können die Einträge in der Routing-Table optimiert werden**

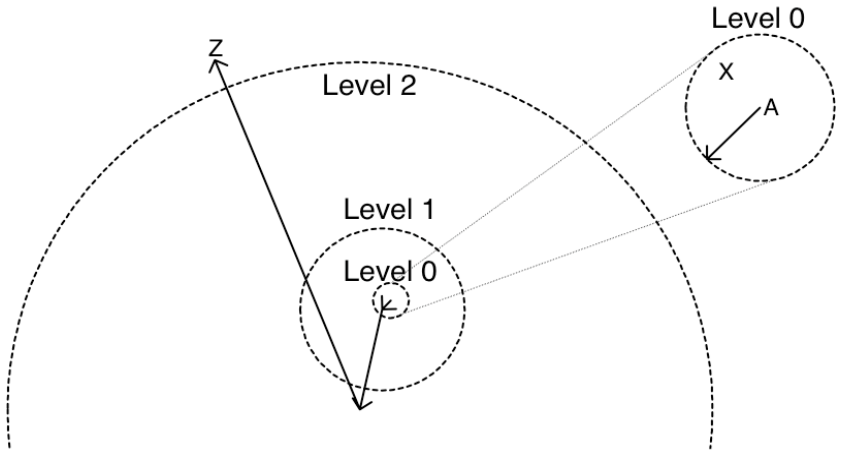
- **Sämtliche Verfahren in Pastry beruhen hier auf Heuristiken**
  - D.h. es gibt keinen mathematischen Nachweis der Güte der Verfahren

- **Annahme: Metrik ist Euklidisch**

Lokalität in Routing-Table

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

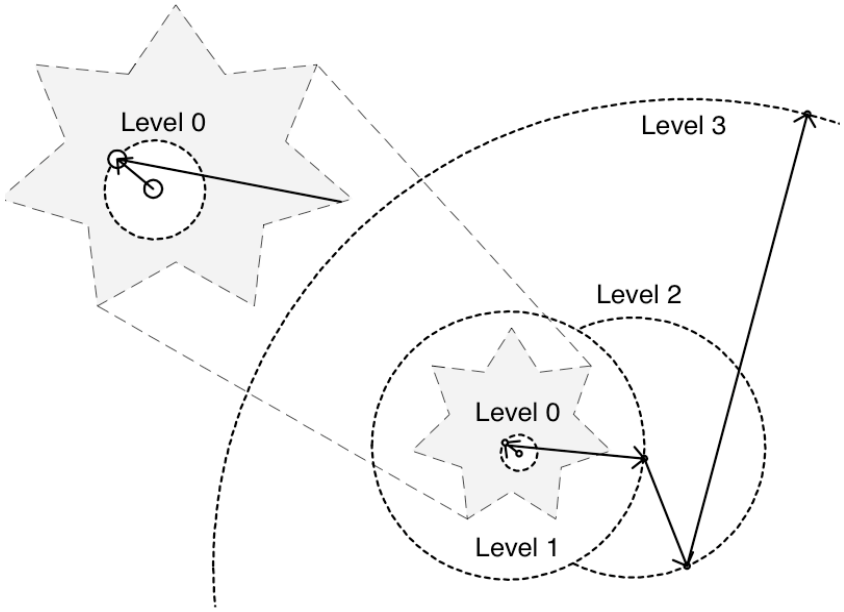
- **Annahme:**
  - Beim Einfügen eines Peers A in Pastry kontaktiert der Knoten zuerst einen nahen Knoten P
  - Alle Knoten haben schon ihre Routing-Table optimiert
- **Aber:**
  - Der zuerst kontaktierte Peer P ist nicht bezüglich der NodeID nahe
- **1. Schritt**
  - Kopiere Einträge der ersten Zeile der Routing-Table von P
    - wegen der Dreiecksungleichung sind diese Einträge recht gut
- **2. Schritt**
  - Kontaktiere passenden Peer P‘ von P mit gleichen ersten Buchstaben
  - Wiederum sind diese Einträge relativ nah
- **Wiederhole diese Schritte bis alle Tabelleneinträge gefüllt sind**



Lokalität im Routing

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Für jeden Eintrag in der Routing-Table wird der nächste Knoten gemäß der Latenzzeit gewählt**
  - Routing wählt nicht unbedingt den kürzesten Weg (bezüglich der Latenzmetrik)
  - Dennoch Hoffnung auf kurze Wege
- **Warum?**
  - Mit der Länge des gemeinsamen Präfix steigt die erwartete Latenzmetrik exponentiell
  - Damit sind die letzten Sprünge am teuersten
  - Hier greifen aber die Leaf-Set-Einträge



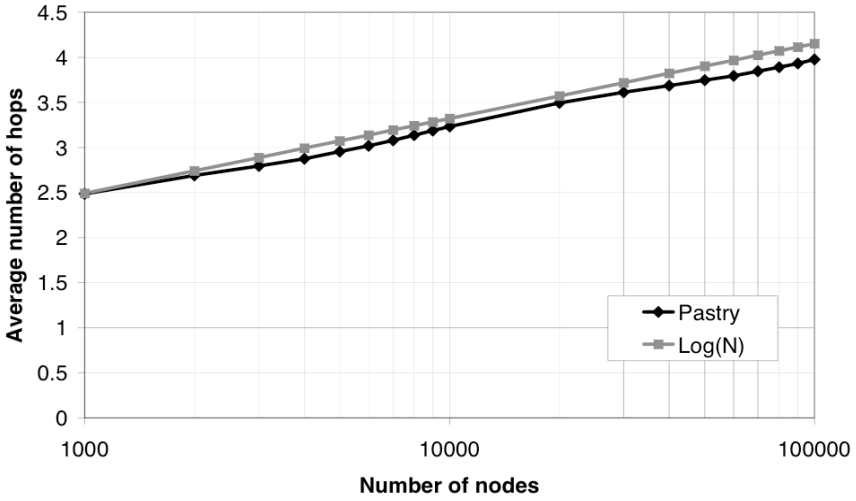
Lokalisierung (latenz-) naher Knoten

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Die Metrik Node-ID und die Latenzmetrik sind völlig unvergleichbar**
- **Bei replizierten Daten auf k Knoten, können nahe Peers mit ähnlicher Node-ID übersehen werden**
- **Hier findet eine Heuristik Anwendung**
- **Experimentelle Untersuchungen scheinen die Güte dieses Ansatzes zu bestätigen**

Experimentelle Resultate Skalierung

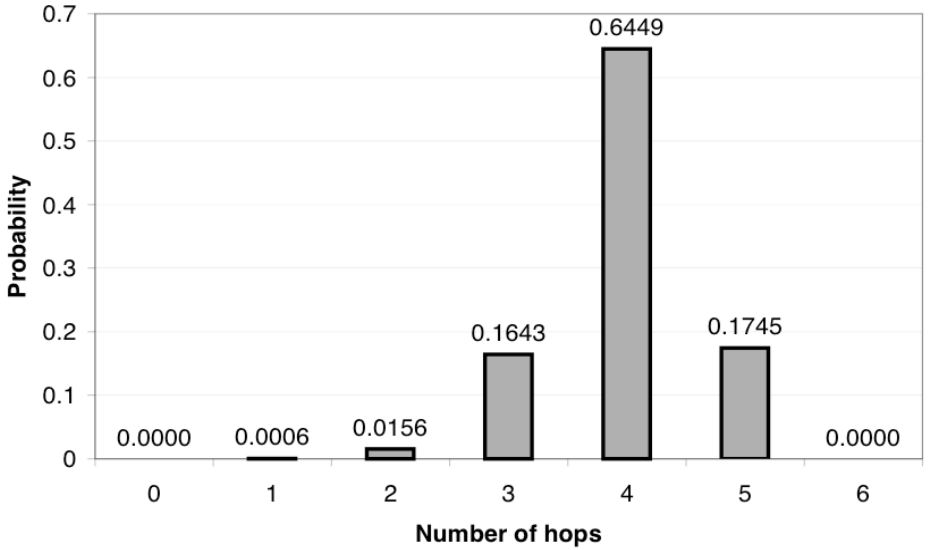
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



- **Parameter b=4, ℓ=16, M=32**
- **In diesem Experiment steigt die durchschnittlich Hop-Distanz logarithmisch mit der Knotenanzahl an**
- **Analyse sagt hier 4 log n voraus**
- **Gute Übereinstimmung**

Experimentelle Resultate Verteilung der Routing-Hops

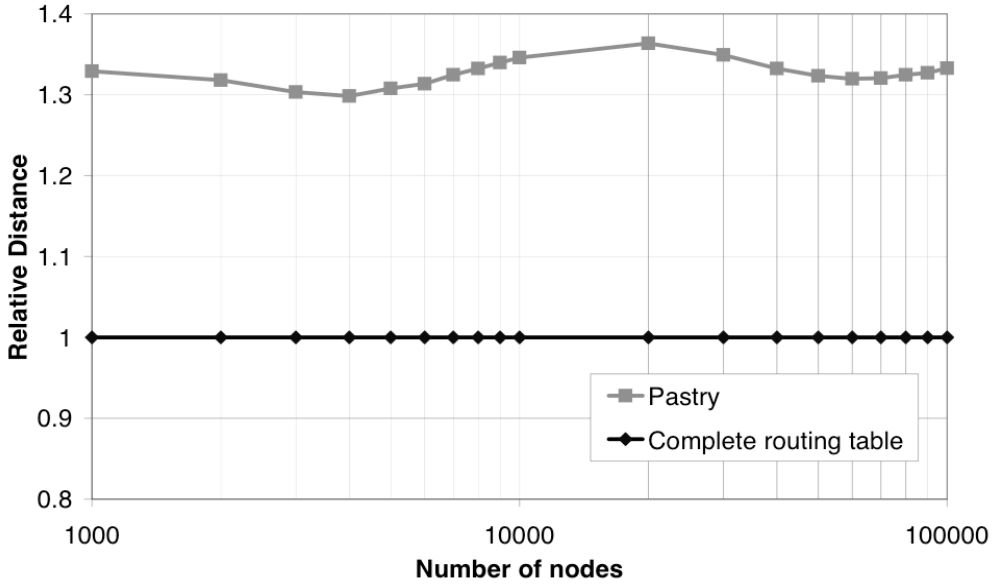
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



- **Parameter b=4, ℓ=16, M=32, n = 100 000**
- **Ergebnis:**
  - –Die Abweichung von der erwarteten Hop-Distanz ist extrem gering
- **Analyse sagt Abweichung mit extrem kleiner Wahrscheinlichkeit voraus**
  - –Gute Übereinstimmung

Experimentelle Resultate Latenzzeit

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



- **Parameter b=4, ℓ=16, M=3**
- **Im Vergleich zum kürzesten Weg ist erstaunlich gering**
  - scheint zu stagnieren

Kritische Betrachtung der Experimente

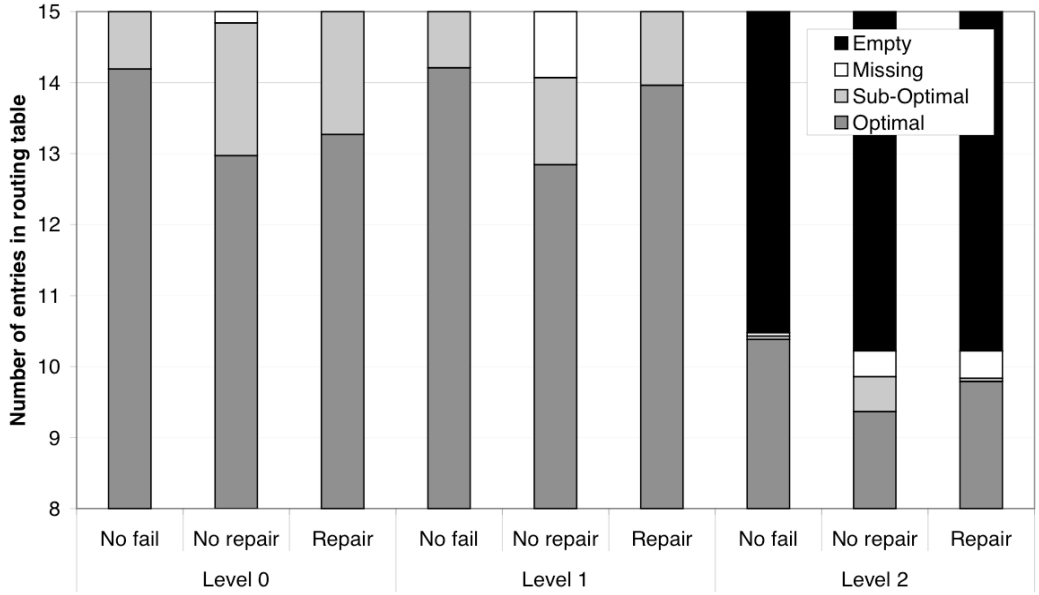
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Experimente geschahen in einer (gutartigen) Simulationsumgebung**
- **Mit b=4, L=16 wird die Anzahl der Links pro Knoten relativ groß**
  - Damit kann der Faktor $2^b/b = 4$ das Ergebnis erheblich beeinflussen
  - Beispiel n= 100 000
    - $2^b/b \log n = 4 \log n > 60$ Links in Routing Table
    - Hinzu kommen noch 16 Links in Leaf-Set und 32 in M
- **Dadurch ist der Grad im Verhältnis zu anderen Protokollen wie CHORD enorm groß**

- **Annahme Euklidischer Latenzzeiten aus der Luft gegriffen**

Experimentelle Untersuchungen Knotenausfälle

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



- **Parameter b=4, ℓ=16, M=32, n = 5 000**

- **No fail: vor Ausfall**
- **No repair: 500 von 5000 Peers fallen aus**
- **Repair: Nach Reparatur der Routing-Tables**

Tapestry

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

von Zhao, Kubiatowicz und Joseph (2001)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Grundlagen und Routing

Tapestry

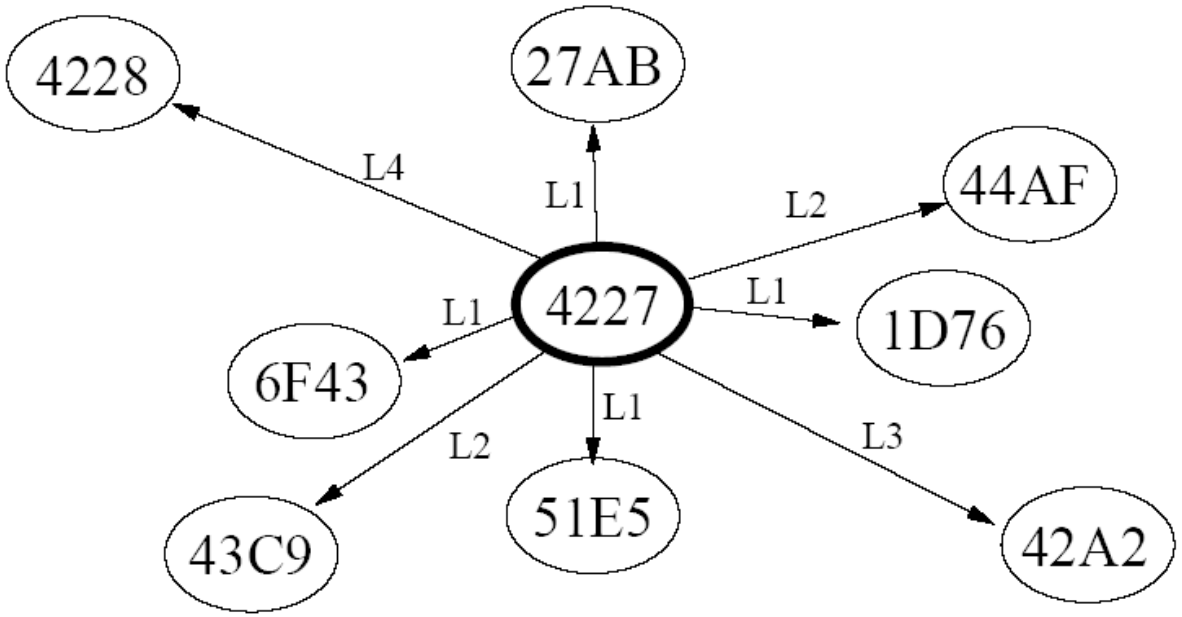
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Objekte und Peers im Netzwerk werden durch eindeutige Objekt-IDs (Globally Unique Identifiers GUIDs) bzw. Peer-IDs identifiziert**
- **IDs werden (wie schon bei CAN und CHORD) durch Hashfunktion berechnet**
- **IDs sind Zeichenketten zur Basis b (Im Fall b=16 also Zahlen im Hexadezimalsystem)**

Nachbarschaftsmenge eines Knotens (1)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Jeder Knoten A unterhält Links zu anderen Knoten, die einen Prefix x seiner Peer-ID mit ihm teilen**
  - Peer mit ID B=xoy hat einen Nachbar A, falls xoy´=A für beliebige y, y´
- **Links sind in Level unterteilt (abhängig von der Länge des gemeinsamen Prefix)**
  - Level L = |x|+1



Nachbarschaftsmengen eines Knotens (2)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Links mit gleichem Prefix werden weiter unterteilt nach dem ersten Buchstaben j, in dem sie sich unterscheiden**

- **Das Peer mit NodeID A besitzt b|A| Nachbarschaftsmengen**

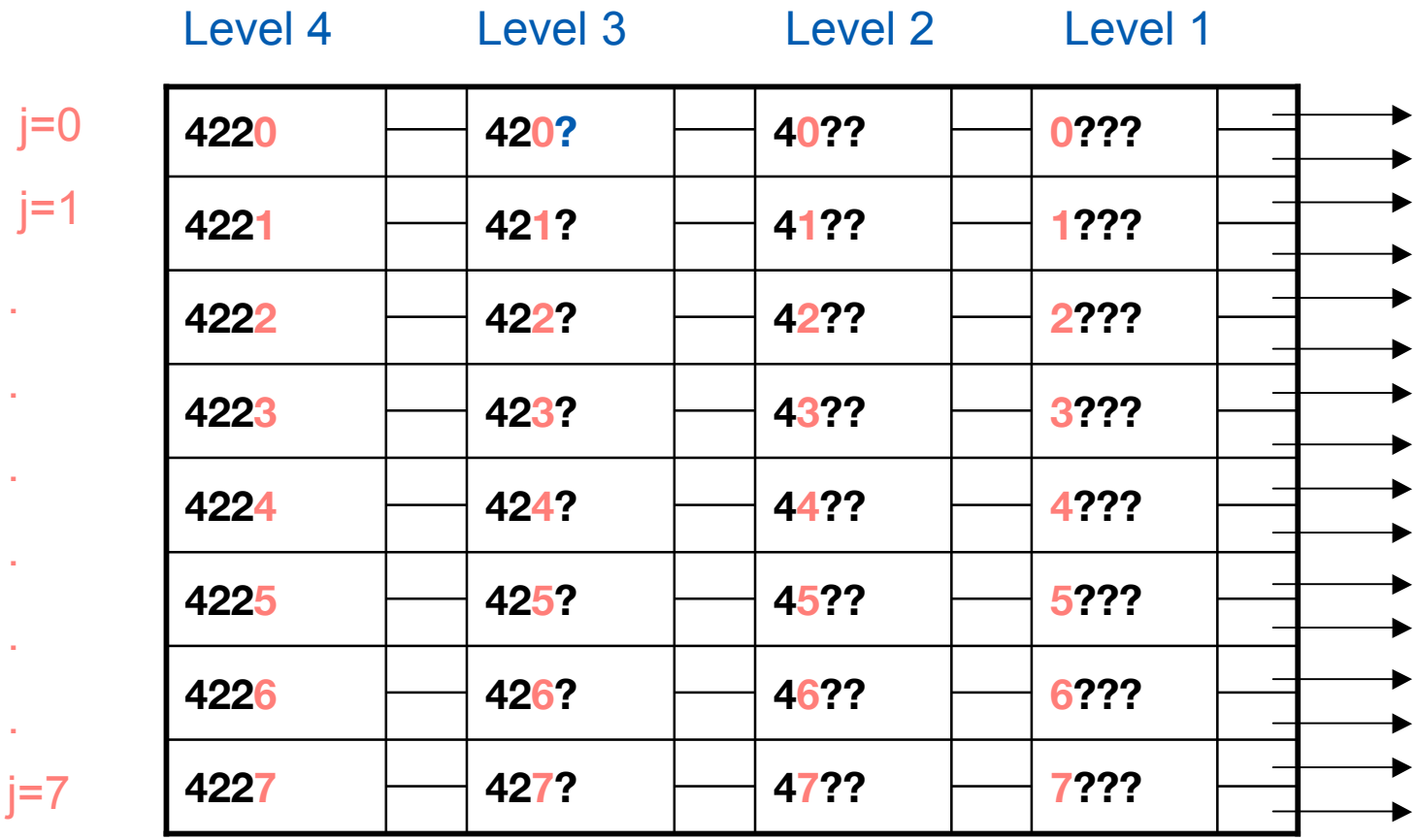
$$N_{x,j}^{A}$$

  - die alle mit xoj beginnenden NodeIDs enthalten
    - Wobei: A = Peer-ID
      - x = Prefix von A
      - j = erster Buchstabe, in dem sich der Nachbar von A unterscheidet
- **Knoten dieser Mengen werden (x,j) Knoten genannt**

Beispiel einer Nachbarschaftsmenge

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Nachbarschaftsmengen des Knotens 4221



Nachbarschaftsmengen (3)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Es werden jedoch nur *k* Links in jeder Nachbarschaftsmenge verwaltet:**

$$k \geq 1 : \left| N_{x,j}^{A} \right| \leq k$$

  - (sonst vollständiger Graph!)

- **Nachbarschaftsmengen können (und werden) leer sein!**

Eigenschaften von Nachbarschaftsmengen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **1. Konsistenz:**

  Falls $N_{x,j}^{A} = \emptyset$ für beliebiges *A*, dann existieren keine (*x*,*j*) Peers im Netzwerk.

  Wir bezeichnen dies als Loch in der Routing-Tabelle bei Level |*x*|+1, Buchstabe *j*.

- Hieraus folgt, dass das Netzwerk zusammenhängend ist

Routing kann geschehen, indem Schritt für Schritt ein Buchstabe des Startknotens *A* dem Zielknoten *B* mit NodeID $b_1 o b_2 o \ldots o b_n$ angepasst wird

So werden sukzessive Knoten der folgenden Mengen gewählt:

N_{ϕ,b_1}^{A} (1. Hop, zu Knoten A_1)

$N_{b_1,b_2}^{A_1}$ (2. Hop, zu Knoten A_2)

$N_{b_1 o b_2,b_3}^{A_2}$ (3. Hop, zu Knoten A_3)

Eigenschaften von Nachbarschaftsmengen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **2. Lokalität**

  Jede Menge $N^A_{x,j}$ enthält die bezüglich eines gegebenen metrischen Raumes nächsten (*x*,*j*) Nachbarn. Der nächste Knoten mit Prefix *xoj* wird ***primärer Nachbar*** genannt, alle anderen ***sekundäre Nachbarn***

Wichtige Eigenschaft, die einen geringen Stretch gewährleistet!

(Verhältnis zwischen der vom Routing gefunden Distanz und der kürzesten Distanz)

Beobachtung:

Jeder Knoten findet seinen nächsten Nachbarn in der Menge

$$\bigcup_{j\in[0,b-1]} N^A_{\phi,j}$$

Root-Sets

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- Objekte mit ID *Y* werden von Root-Peers R_Y verwaltet.
- Zuständige Peers werden durch Funktion *MapRoots*(*Y*) berechnet.
- Menge R_Y wird Root-Set genannt und es gilt:

Eigenschaft 3:

- **Eindeutiges Root-Set**

  Root-Set R_Y für Objekt *Y* muss eindeutig sein, egal wo im Netzwerk *MapRoots*(*Y*) berechnet wird

(Einfach zu gewährleisten, falls Hashfunktionen benutzt werden)

Problem: von *MapRoots*(*Y*) gelieferte Knoten müssen existieren!!

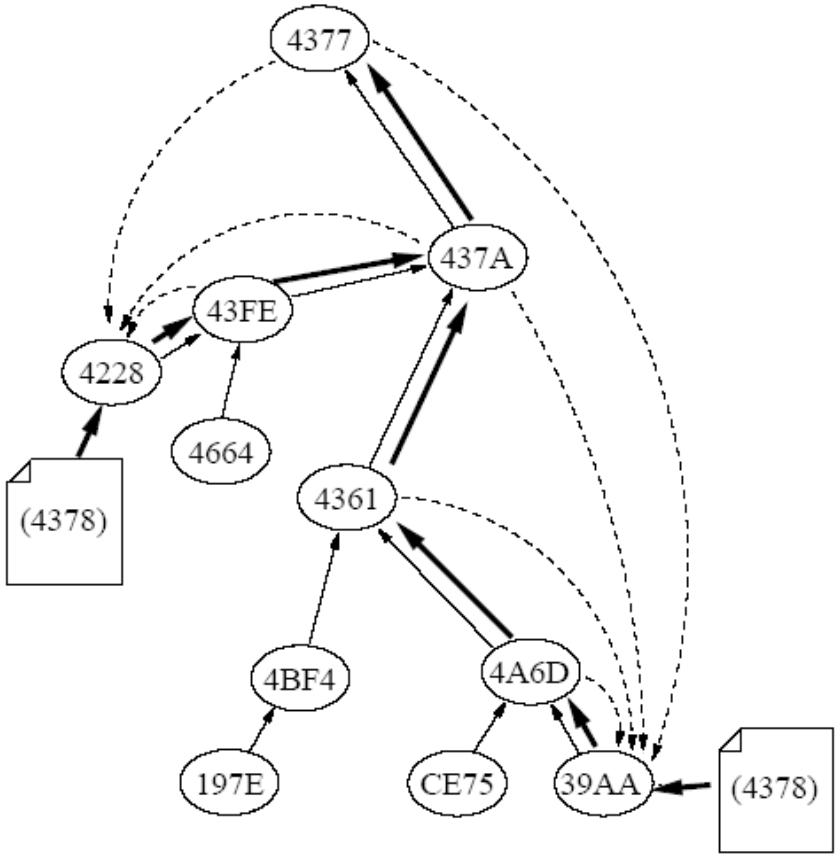
Publikation von Objekten

- Peers die ein Objekt *Y* bereitstellen (***Storage Server***), müssen dies den zuständigen Root-Peers mitteilen:

Storage Server berechnet $MapRoots(Y)=R_Y$

Und schickt Nachricht an **alle** Roots in R_Y
(entlang primärer Nachbarn)

Jeder Peer, der die Nachricht weiterleitet,
speichert Objekt-Zeiger auf den Storage-Server



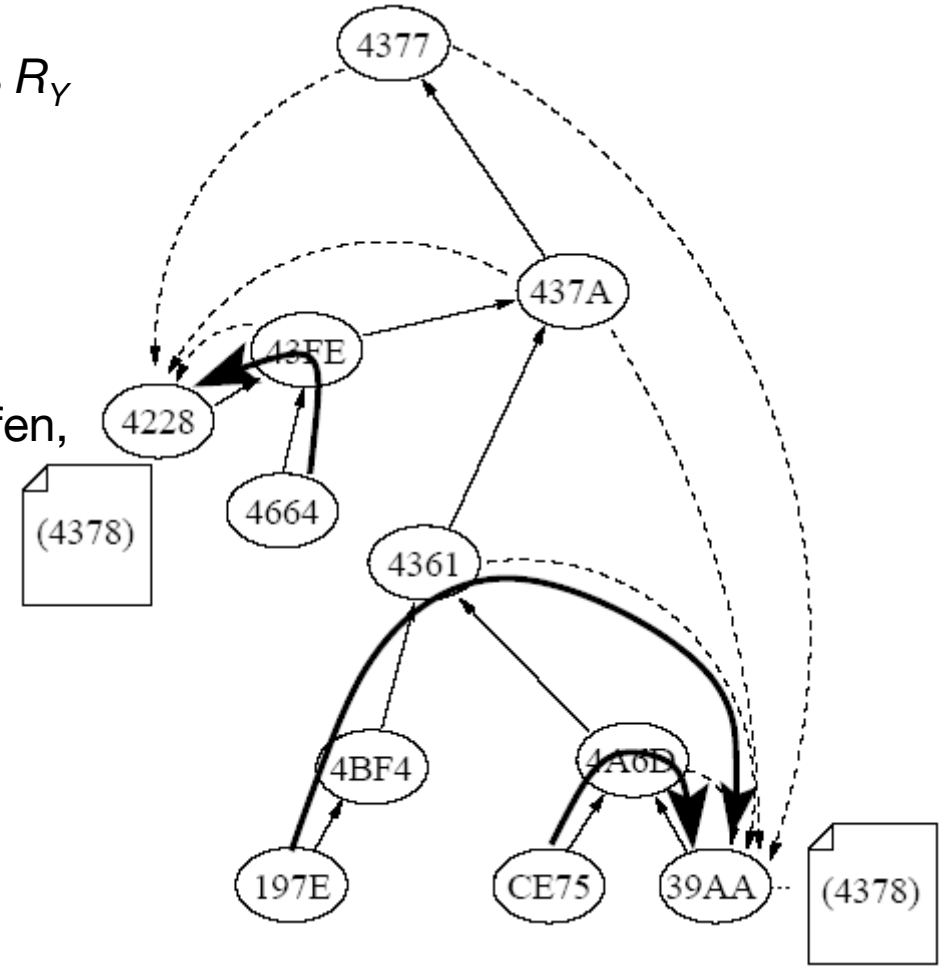
Anfragen

Sucht ein Peer Objekt *Y*, so berechnet er ebenfalls *MapRoots*(*Y*)

wählt einen zufälligen Root-Server aus R_Y

Routet zu diesem Root-Server
(entlang primärer Nachbarn)

Wird dabei ein Objekt-Zeiger angetroffen,
so wird direkt zum Storage-Server
gesprungen



Fehlertoleranz

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Beobachtung 1:

Wenn $|R_Y| > 1$ und die Root-Server in R_Y unabhängig voneinander sind (uniformes Hashing), können fehlgeschlagene Suchen mit einem anderen Root-Server wiederholt werden.

⇒**Kleine** temporäre Fehler in den Routing Tabellen können toleriert werden

Fehlertoleranz durch Soft-State Pointer:

Links auf Objekte werden nach bestimmter Zeit verworfen

=>Storage Server müssen ihre Objekte regelmässig neu publizieren

- Vermeidet tote Links auf bereits ausgeschiedene Storage Server
- Neu eingetroffene Peers werden mit Links versorgt

Surrogate Routing

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Bislang vernachlässigt:

- *MapRoots*(*Y*) kann auf Server verweisen, die nicht existieren!
  (sogar mit hoher Wahrscheinlichkeit! Warum??)
- Was geschieht bei Löchern in den Routing-Tabellen?

Lösung:

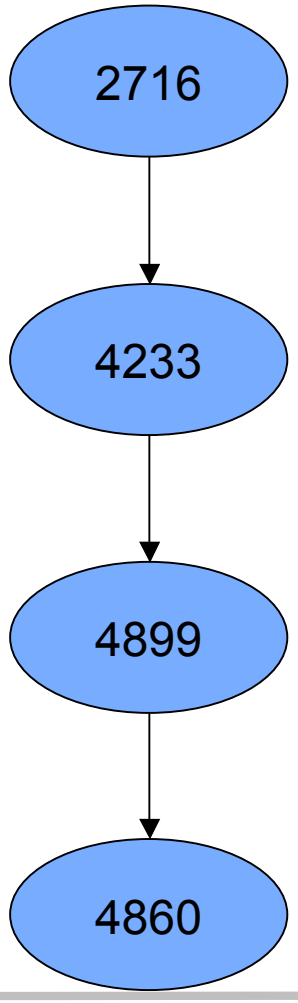
Surrogate Routing

- Versuche schrittweise den berechneten Root-Server zu erreichen
- Wird dabei auf ein Loch an Stelle (x,j) getroffen, dann wähle den nächsten Eintrag (x,j+1) (existiert dieser nicht, dann (x,j+2),…)
- Fahre so lange fort, bis der aktuelle Peer der letzte Peer auf oder oberhalb des aktuellen Routing Levels ist.

Der so gefundene Peer wird als Surrogate-Server bezeichnet

Beispiel Surrogate Routing

- Suche nach 4666 durch Peer 2716:



Level 1, j=4

Level 2, j=6 existiert nicht, nächster Link: j=8

Level 3, j=6

Knoten 4860 hat keine Level 4 Nachbarn => fertig

Eindeutigkeit Surrogate Routing

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Theorem:**
  - Wenn Eigenschaft 1 gilt (Konsistenz), dann wird Surrogate Routing einen eindeutigen Root-Server bestimmen

Surrogate Routing

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Beobachtung 2:**
  - Surrogate Routing lässt sich auf Gebrauch mit mehreren Root-Peers verallgemeinern.
  - Es muss lediglich zum Surrogate-Peer des ausgewählten Root-Peers geroutet werden.

- **Theorem**
  - Routing in Tapestry benötigt O(log n) Hops mit hoher Wahrscheinlichkeit

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Einfügen neuer Peers

Einfügen neuer Peers

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Wird ein Peer in das Netzwerk eingefügt, so soll das entstehende Netzwerk das gleiche sein, als wenn das Netzwerk von Grund auf mit ihm konstruiert worden wäre**

- **Dafür muss Folgendes gewährleistet sein:**

- **Eigenschaft 4:**
  - Liegt Knoten A auf dem Pfad zwischen dem publizierenden Knoten von Objekt Y und dessen Root-Server, dann besitzt A einen Link auf Y

Algorithmus zum Einfügen eines Peers

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Der Algorithmus lässt sich in die folgenden Schritte gliedern

- Suche im Netzwerk nach eigener ID, kopiere Nachbarschaftstabellen von so gefundenem Surrogate-Peer
- Kontaktiere die Peers, die Löcher in ihren Routing Tabellen haben, die durch den neuen Peer gefüllt werden können
  Bsp: 1234 ist neuer Peer und es existierten keine 123? Peers, so müssen alle 12?? Peers benachrichtigt werden
- Aufbauen der eigenen Routing Tabelle

Wir betrachten zunächst Schritt 2 …

Acknowledged Multicast Algorithmus

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- Multicast Nachricht besteht aus Prefix X und einer Funktion

Ablauf:

- Der neue Peer sendet Multicast Nachricht an seinen Surrogate-Peer $N^A_{x,*}$ (bestehend aus gemeinsamen Prefix und Funktion)
- Empfängt ein Knoten A die Nachricht, sendet er sie weiter an seine Nachbarn (mit Prefix x*)
- Empfängt ein Knoten eine Nachricht, die nicht mehr weitergeleitet werden kann, wendet er die mitgesendete Funktion an.
- Die Funktion transferiert Objekte zum neuen Peer und löscht nicht mehr benötigte Verweise
- Jeder Knoten, der Nachrichten verschickt hat, erwart eine Rückmeldung von allen Empfängern. Sind diese eingetroffen, benachrichtigt er den „über“ ihm liegenden Peer

Aufwand Acknowledge Multicast Algorithmus

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Vernachlässigt man die Nachrichten, die ein Peer an sich selbst sendet, so entsteht ein Spannbaum**
- **Werden k Peers erreicht, so hat der Baum k Knoten und k-1 Kanten**
- **O(k) Nachrichten**

Aufbauen der Nachbarschaftsmengen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Wir wollen die Nachbarschaftsmengen $N^A_{x,j}$ für einen neuen Peer A bestimmen

- Diese müssen Eigenschaften 1 (*Konsistenz*) und 2 (*Lokalität*) erfüllen!

  Dies kommt dem Lösen des Nächste-Nachbarn Problems für viele verschiedene Prefixe gleich

- Die Nachbarschaftsmengen werden nun Level-weise aufgebaut.

- Aus dem vorigen Schritt (Multicast) kennen wir bereits alle Knoten, die das Prefix x, $|x|=i$, mit A teilen

- Nun werden schrittweise die Level i Nachbarschaftsmengen aus den Level $i+1$ Nachbarschaftsmengen berechnet

Schichtweise Erweiterung der N.-Mengen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Algorithmen Beschreibung

- **Fordere von allen Level *i* Nachbarn Listen mit ihren Links auf Level *i*-1 an**
- **Kontaktiere alle diese Peers und berechne Entfernung zu Ihnen (dabei prüfen kontaktierte Peers, ob sie den neuen Knoten selbst noch aufnehmen müssen)**
- **Speichere die *k* nächsten Peers in der eigenen Nachbarschaftsmenge (dies gewährleistet die Lokalität)**
- **Fahre fort, bis Level 0 erreicht ist**

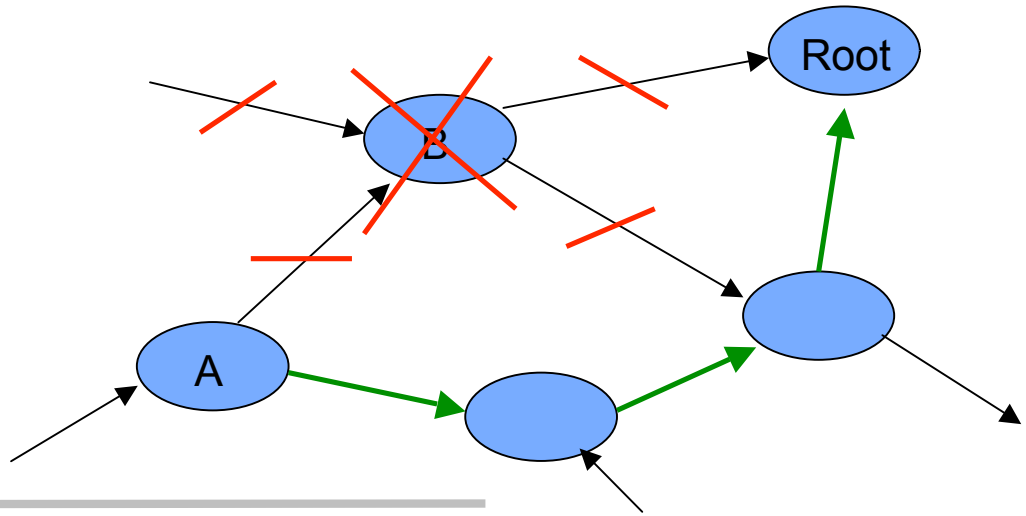
➢ Algorithmus hat mit hoher Wahrscheinlichkeit Laufzeit $O(log^2\ n)$

Entfernen von Knoten

Entfernen von Knoten

- **Stellt Knoten A fest, dass Nachbar B tot ist, dann:**

- **Entferne B aus Routing Tabelle**
  - Entstehen hierbei Löcher, müssen diese gefüllt werden, oder es muss sichergestellt sein, dass das Loch nicht gefüllt werden kann
    - Lösung: Acknowledged Multicast
- **Re-Publiziere alle Objekte, deren nächster Hop zum Root-Peer B ist (grüne Kanten)**



Pastry versus Tapestry

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Beide beruhen auf dem selben Routing-Prinzip von**
  - Plaxton, Rajamaran und Richa
  - Verallgemeinerung von Routing auf dem Hyperwürfel
- **Tapestry**
  - ist nicht vollständig selbstorganisierend
  - achtet stark auf die Konsistenz der Routing-Tabelle
  - ist analytisch verstanden und hat nachweisbare Performanz
- **Pastry**
  - Analytische Methodik wird ersetzt durch Heuristiken
  - Algorithmische Beschreibung ungenau
  - Experimentelle Verifikation ersetzt analytische Untersuchungen
  - Praktisch besser umsetzbar
  - Durch Leaf-Sets sehr robust

Ende der 12. Vorlesung

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Peer-to-Peer-Netzwerke

Christian Schindelhauer
schindel@informatik.uni-freiburg.de