Skalierbare zusammenfassungsbasierte Ähnlichkeitssuche in P2P-Netzen

Dr. Wolfgang Müller
Medieninformatik (LS Prof. Dr. Henrich)
Fakultät für Wirtschaftsinformatik und Angewandte Informatik
Otto-Friedrich-Universität Bamberg

Gliederung

1. Einführung P2P & Herausforderung
2. PlanetP: unsere Baseline
3. Rumorama: skalierbares PlanetP
4. Erweiterungen

Herausforderungen für P2P-Netze

- Quality of Service
- Skalierbarkeit
- Ausfallsicherheit
- Sicherheit gegen Angriffe

Herausforderungen für P2P-Netze

- Quality of Service
- Skalierbarkeit
- Ausfallsicherheit
- Sicherheit gegen Angriffe

Nutzungsmodell für Skalierbarkeit

- **1.000.000** Peers

- Texte: **20.000 Byte pro Zusammenfassung**
- Bilder: **400 Byte pro Zusammenfassung**

Nutzermodell für Ausfallsicherheit

- Erfahrungswert in Filesharing-Netzwerken:

- **Halbwertszeit ca. 1h**

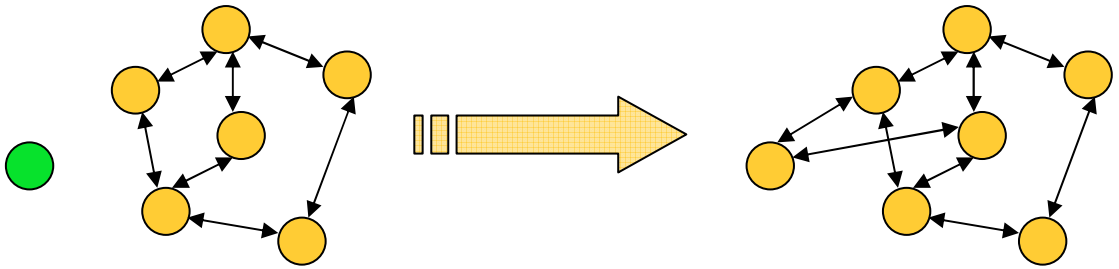
➔ **Modell: 5% Churn pro 5 Minuten**

Umwälzungsrate: Anteil der Peers die pro Zeiteinheit durch neue Peers ersetzt werden

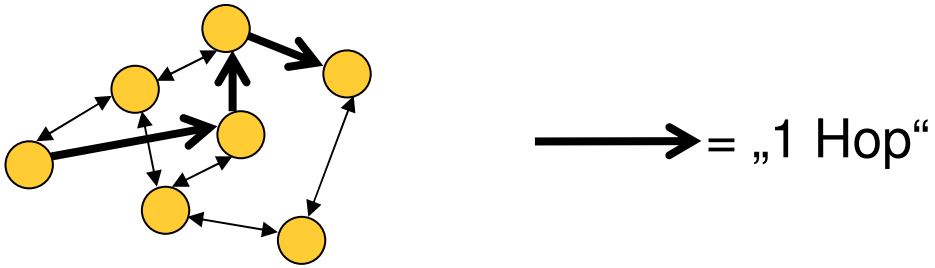
- Problem:
  Keine Erfahrungswerte mit sehr großen „ernsthaften“ Netzwerken

Lebenszyklus eines Peers

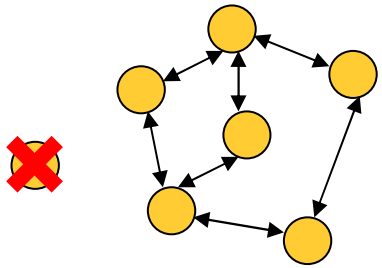
- Join



- Operation



- Leave



Ansätze zur P2P-Ähnlichkeitssuche

- Verteilte Indexstrukturen

- „Semantisches“ Routing:
  Peer-Zusammenfassungen als Routing-Information
  - Multi-Hop
  - Single Hop (Source Selection)

Ansätze zur P2P-Ähnlichkeitssuche

- Verteilte Indexstrukturen

- „Semantisches“ Routing: Peer-Zusammenfassungen als Routing-Information
  - Multi-Hop
  - Single Hop (Source Selection) ➔ PlanetP

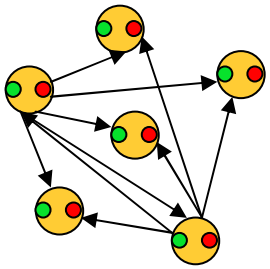
PlanetP

- Jeder Peer kennt alle Zusammenfassungen
- Zusammenfassungen mittels Rumor Spreading repliziert
- Anfragebearbeitung mittels *Source Selection*
  - Peer *g* erhält Anfrage *q*
  - *g* wählt anhand *q* und Zusammenfassungen Kandidaten aus
  - Gibt Anfrage an Kandidaten weiter
  - *g* Vereinigt Resultate der Kandidaten

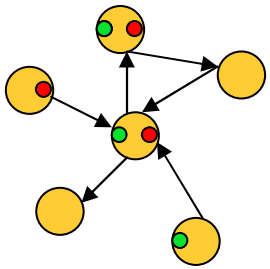
Peers, die wahrscheinlich gute Matches enthalten

Replikation mit Rumor Spreading (epidemische Algorithmen)

- Broadcast zu teuer:
  - viel Overhead
  - Hohe Maximalbelastung
  - Keine echte Replikation



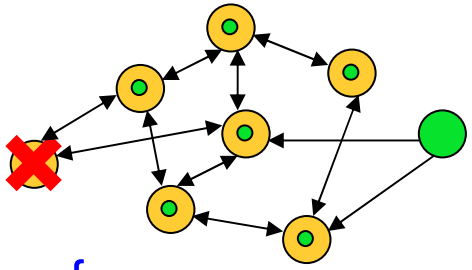
- Rumor Spreading billiger:
  - Zu festen Zeitpunkten neue Daten an *n* Nachbarn weitergeben
  - „Random Phone Call Model“
  - Langsamer, aber kostengünstiger



PlanetP, 1.000.000 Knoten, 5% Churn

Churn: Austausch eines Peers

- 1 Peer leave
- 1 Peer join



➔ Alle Peers erhalten eine neue Zusammenfassung

5% Churn in 5 Minuten:

➔ Alle Peers tauschen je 5 Minuten
50.000 Zusammenfassungen aus

➔ **26.600.000 Bit/s** **Traffic/Peer** (IR)

➔ **600.000 Bit/s** **Traffic/Peer** (CBIR)

Beurteilung PlanetP

- Peers wissen viel, nicht *zu* viel über Nachbarn
- Verspricht Effizienzgewinn gegenüber Broadcast
- Nutzbar für Ähnlichkeitsanfragen auf hochdimensionalen Vektoren

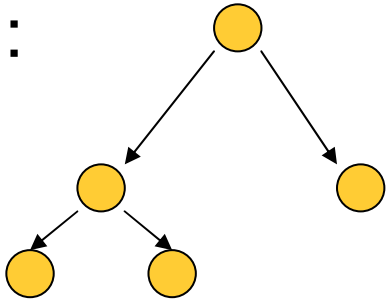
- Nicht skalierbar

Rumorama

- Idee: eine Hierarchie von PlanetP Netzen
  - „Blattnetze“ arbeiten wie PlanetP
    - verteilen Zusammenfassungen per Rumor Spreading
    - nutzen Zusammenfassungen zur Source Selection
  - die maximale Größe der Blattnetze kann definiert werden
  - eine Zugriffshierarchie zu den Blattnetzen wird überlagert
- Herausforderungen
  - Peers sind unzuverlässig
  - Peers sind unterschiedlich leistungsfähig

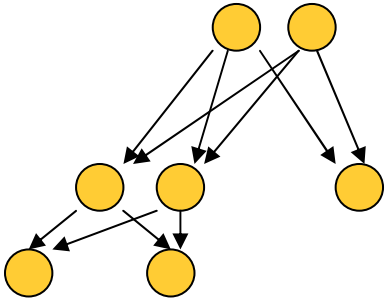
Hierarchien vs. P2P

- reine Hierarchien führen zu Problemen:
  - Wurzel ist „hot spot“
  - „Single point of failure“



- Lösung: „Multi-Hierarchie“
  - Jeder ist ein Blatt …
  - Jeder ist ein innerer Knoten …
  - Jeder ist ein Wurzelknoten …

  zumindest manchmal



Zum Begriff des Blattnetzes

- Information nicht instantan im Netz verteilt

➔ Peers haben unterschiedlichen Kenntnisstand über Netz

- Peers können Blattnetzgröße wählen

➔ Peers sehen unterschiedliche Blattnetze

➔ Es gibt nicht „das Blattnetz“

Die weiterentwickelte Idee

- jeder Peer hat zwei Arten von Nachbarn
  - „Nachbarn“ (lediglich als Kontaktinformation)
  - „Freunde“ (Kontaktinformation & Inhaltszusammenfassung)

- Daten eines Peers:
  - seine PeerId
  - FriendsMask
  - Liste der Nachbarn
  - Liste der Freunde & ihrer Inhaltszusammenfassungen

IDs, Masken & FriendsMasks

IDs haben die gleiche Struktur wie Masken!

- PeerID
  - ID interpretiert als 160bit String
  - z.B. [01010001010101…010110]

```
matches([01],[0])
nicht matches([01],[1])
```

- Masken:
  - Bitstring beliebiger Länge [1011]
  - eine Maske *m* „matcht“ eine Maske *m‘* ⇔ *m* ist Präfix von *m‘* oder umgekehrt

- Freunde eines Peers = Peers, deren PeerID seine FriendsMasks matchen!
- Jeder Peer ist sein eigener Freund (FriendsMasks ist Präfix seiner PeerID)
- längere FriendsMask ➔ weniger Peers matchen ➔ weniger Freunde
  - Effekt: Anzahl der Freunde kann für jeden Peer individuell geregelt werden

Nachbarn

seine FriendsMask

- Peer verwaltet Referenzen auf Nachbarn
- ID [10111]11000110101010…
- Je einige Nachbarn (zufällige Auswahl) für
  - 0… und 1…
  - 10… und 11…
  - 100… und 101…
  - 1010… und 1011…
  - 10110… und 10111…

längere Bitgruppen möglich

Ablauf einer Anfrage

- Teilschritte:
  1. Verteilung der Anfrage in alle Blattnetze
  2. Bearbeitung der Anfrage in den Blattnetzen
  3. Kombination der Ergebnisse auf dem „Rückweg“

- Verteilung: über Nachbarn
- Bearbeitung: unter Freunden

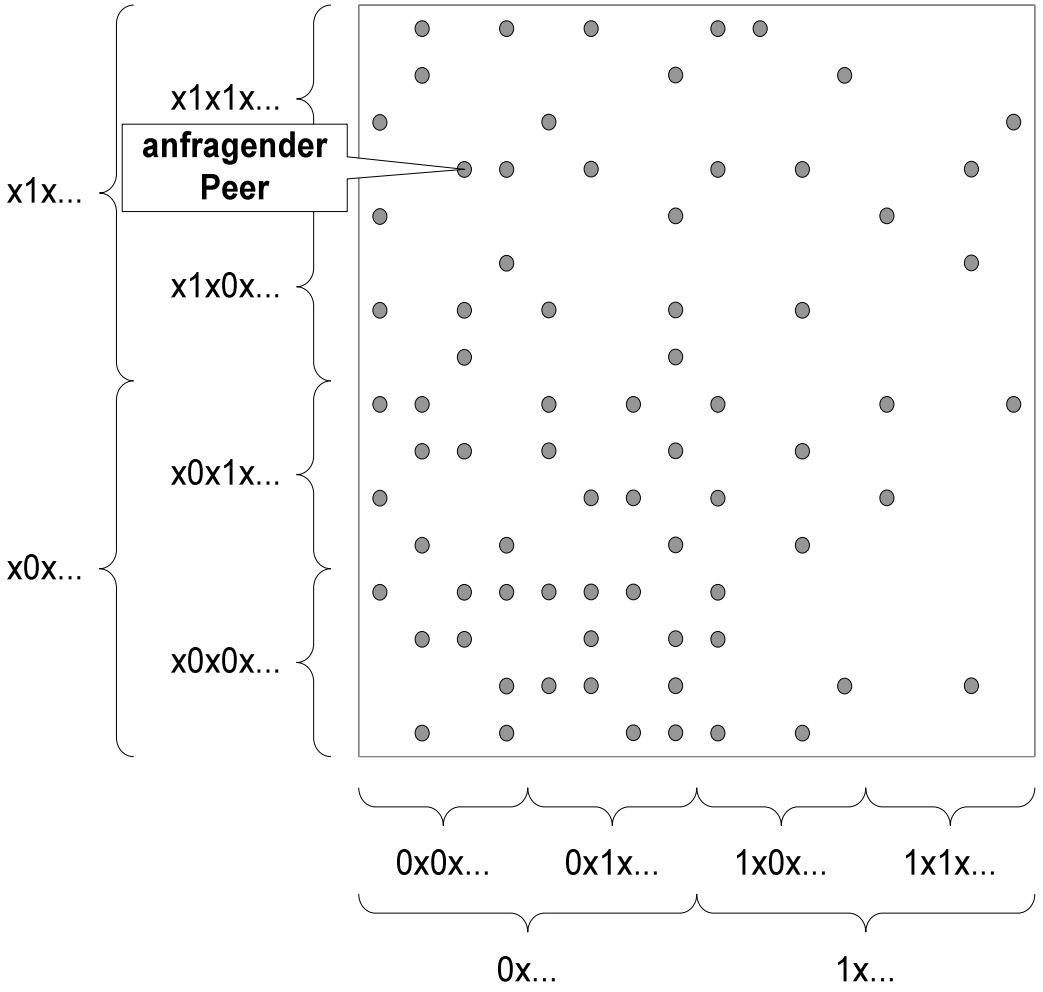
Anfragen

Maske der Anfrage (initial leer)

gewünschter Umfang des Ergebnisses

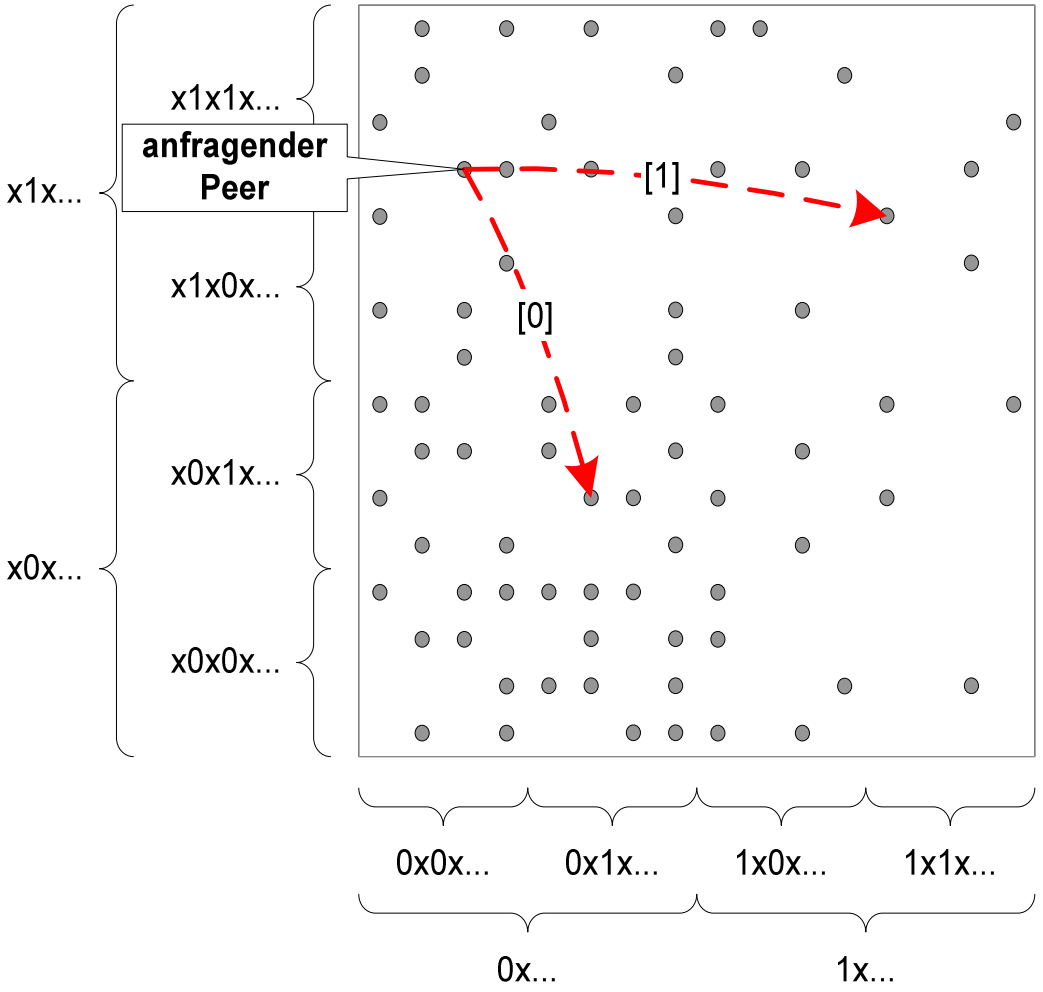
- Peer erhält Nachricht
  `somePeer.query(mask,requestedSize,q)`
- Falls |mask| ≥ |FriendsMask|:
  - der Peer kennt alle Zusammenfassungen von Peers mit matchender PeerID
  - Folge: er kann unter seinen Freunden eine „herkömmliche“ PlanetP Anfrage bearbeiten
- Falls |mask| < |FriendsMask|:
  - Peer kennt nur einige Peers, die `mask` matchen
  - er leitet die Anfrage daher weiter an
    - `neighbor[mask+"1"].query(mask+"1",requestedSize,q)`
    - `neighbor[mask+"0"].query(mask+"0",requestedSize,q)`

Ablauf einer Anfrage in Rumorama



1. Schritt der Verteilung

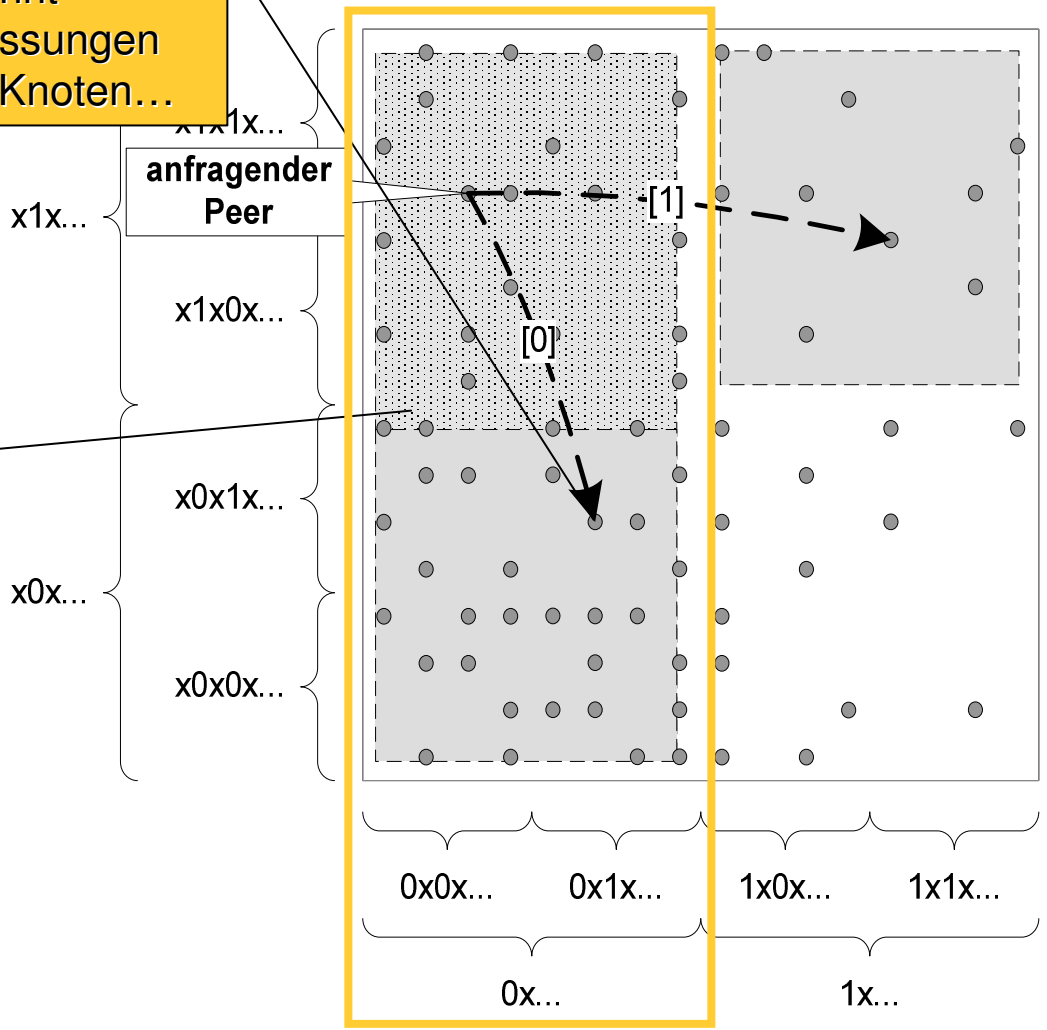
es sind noch keine Blattnetze erreicht



1. Schritt der Verteilung (genauer)

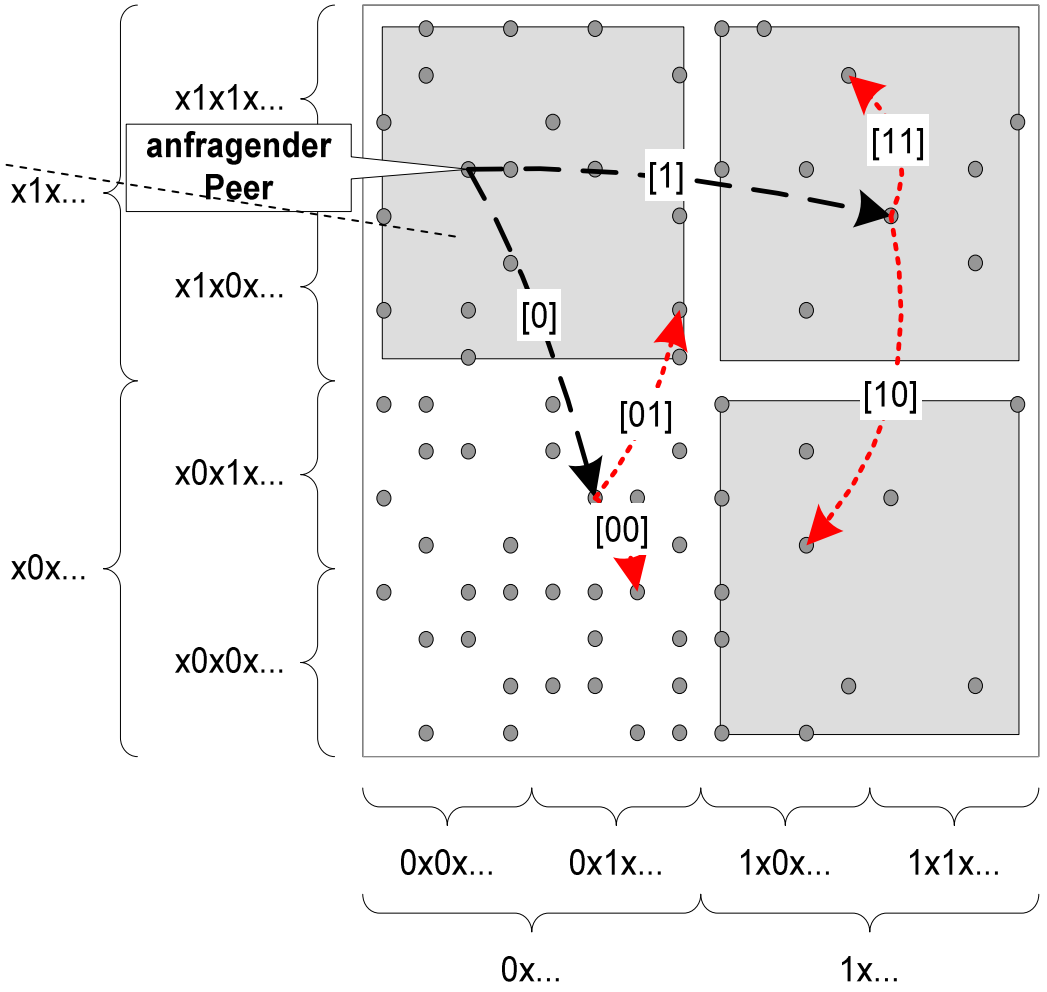
Peer kennt Zusammenfassungen ausgegrauter Knoten…

…aber nicht Knoten in Schraffiertem Bereich

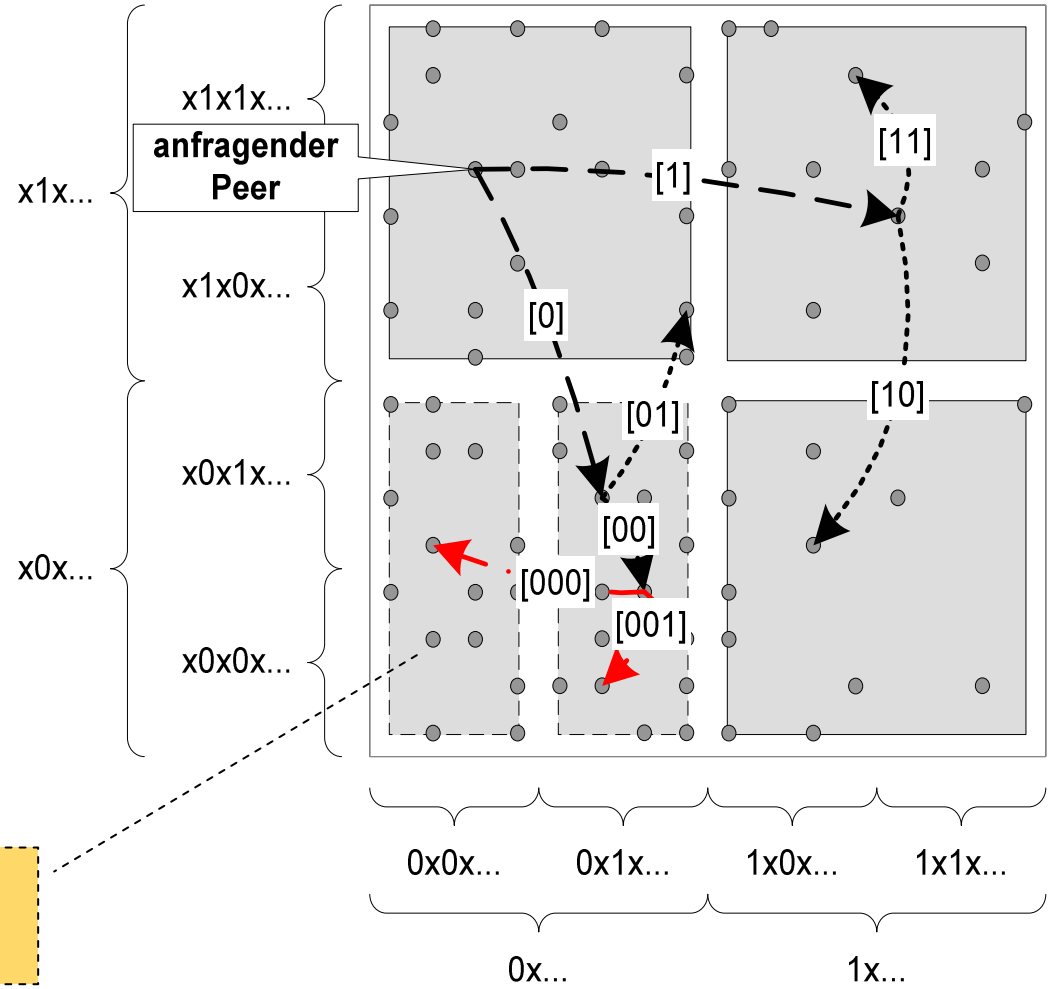


2. Schritt der Verteilung (parallel)

nun sind bei 3 Peers Blattnetze erreicht; hier kann bereits mit der Anfrage-bearbeitung in den Blattnetzen begonnen werden



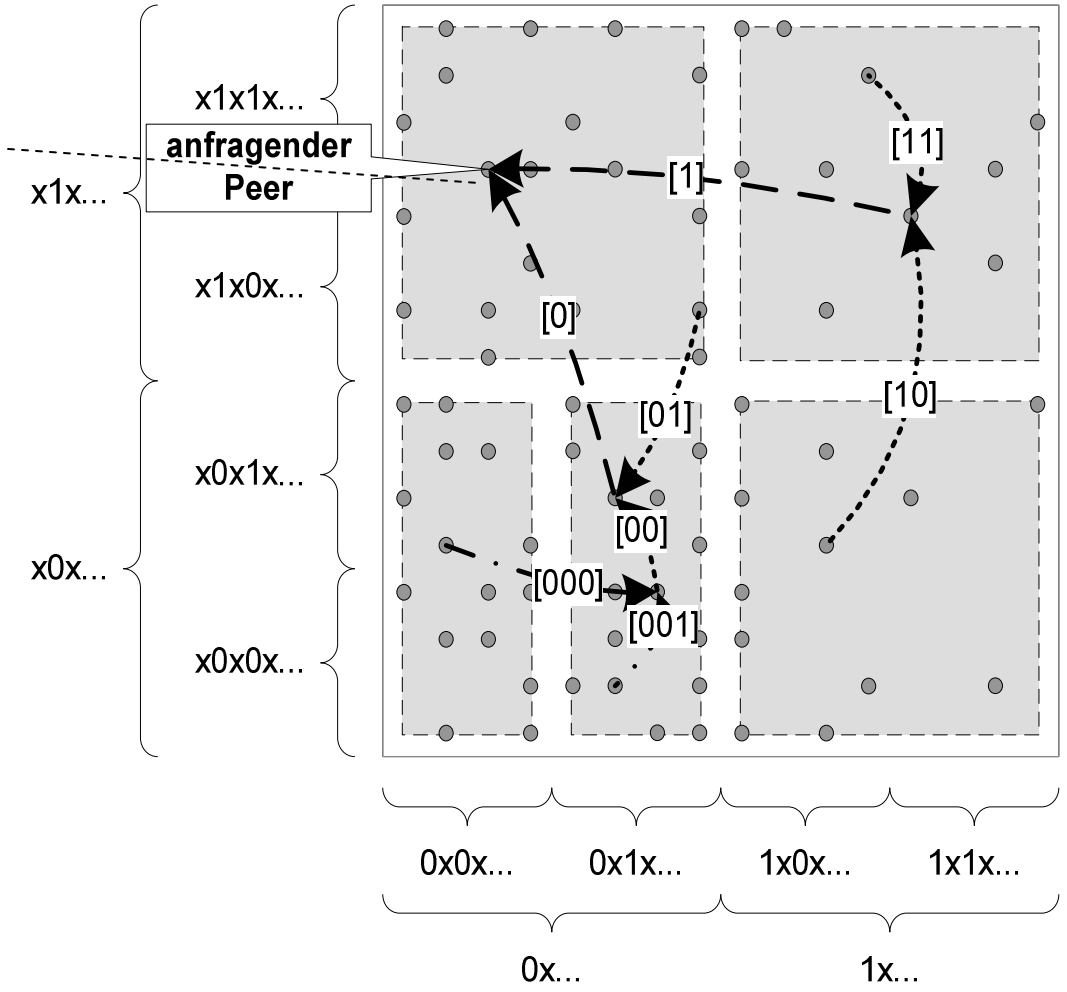
3. Schritt der Verteilung (parallel)



nun sind überall Blattnetze erreicht

Zusammenfassung der Ergebnisse

die Ergebnisse werden nun zurückgegeben



Stärken von Rumorama

- Effizient
  - Kommunikationsaufwand einer Anfrage ~ PlanetP
  - Anfrage benötigt log(*N*) Zeitaufwand (durch Parallelität)
  - Koexistenz verschiedener FriendsMask-Längen / Blattnetzgrößen
- Robustheit durch Multi-Hierarchie

Details

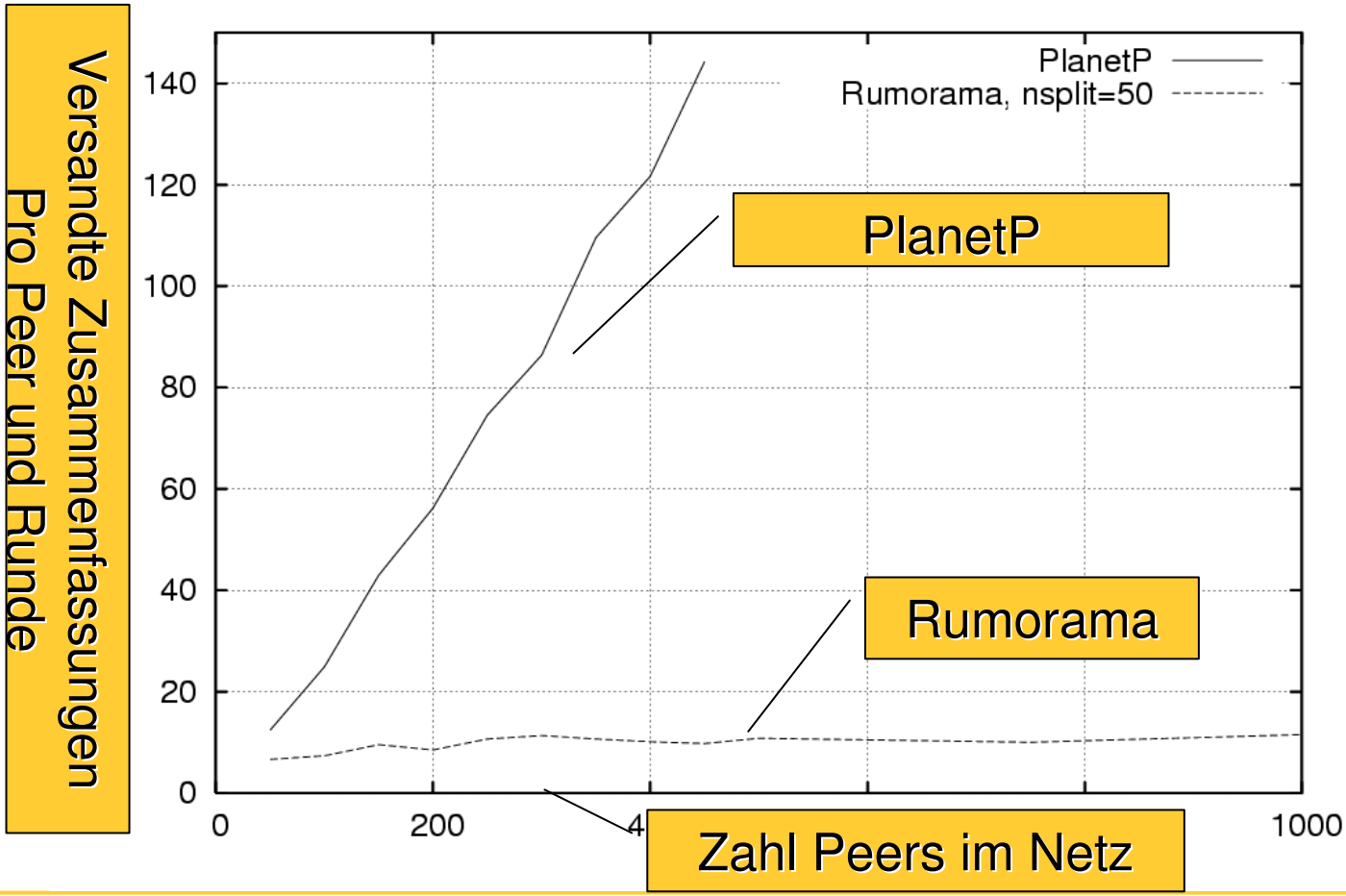
Drei parallele Rumor-Spreading-Prozesse

1. Friend Spreading (Verteilung von Freund-Adressen)
2. Summary Spreading (Verteilung von Zusammenfassungen)
3. Neighbor Spreading (Auffrischen der Nachbarnlisten)

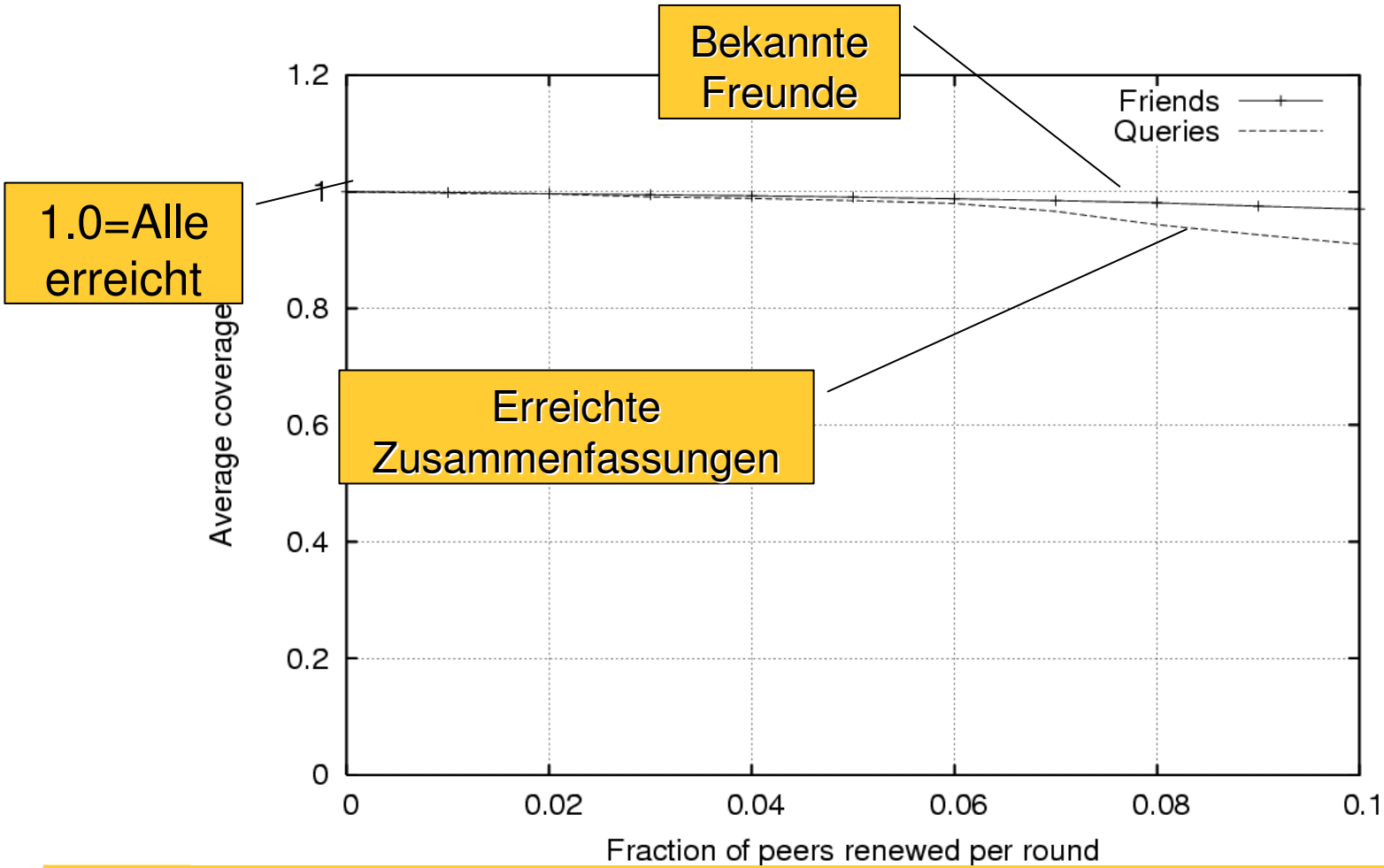
Experimente

- Aufbau
  - Discrete-Event Simulation
  - Simulation von Ausfällen von Peers (0..10% Churn)
  - Verschiedene Größen des Netzwerks
- Messungen
  - Anzahl der verschickten Zusammenfassungen
  - Tiefe des Multicast-Baums (Hops)
  - Anteil der erreichten Zusammenfassungen/Churn

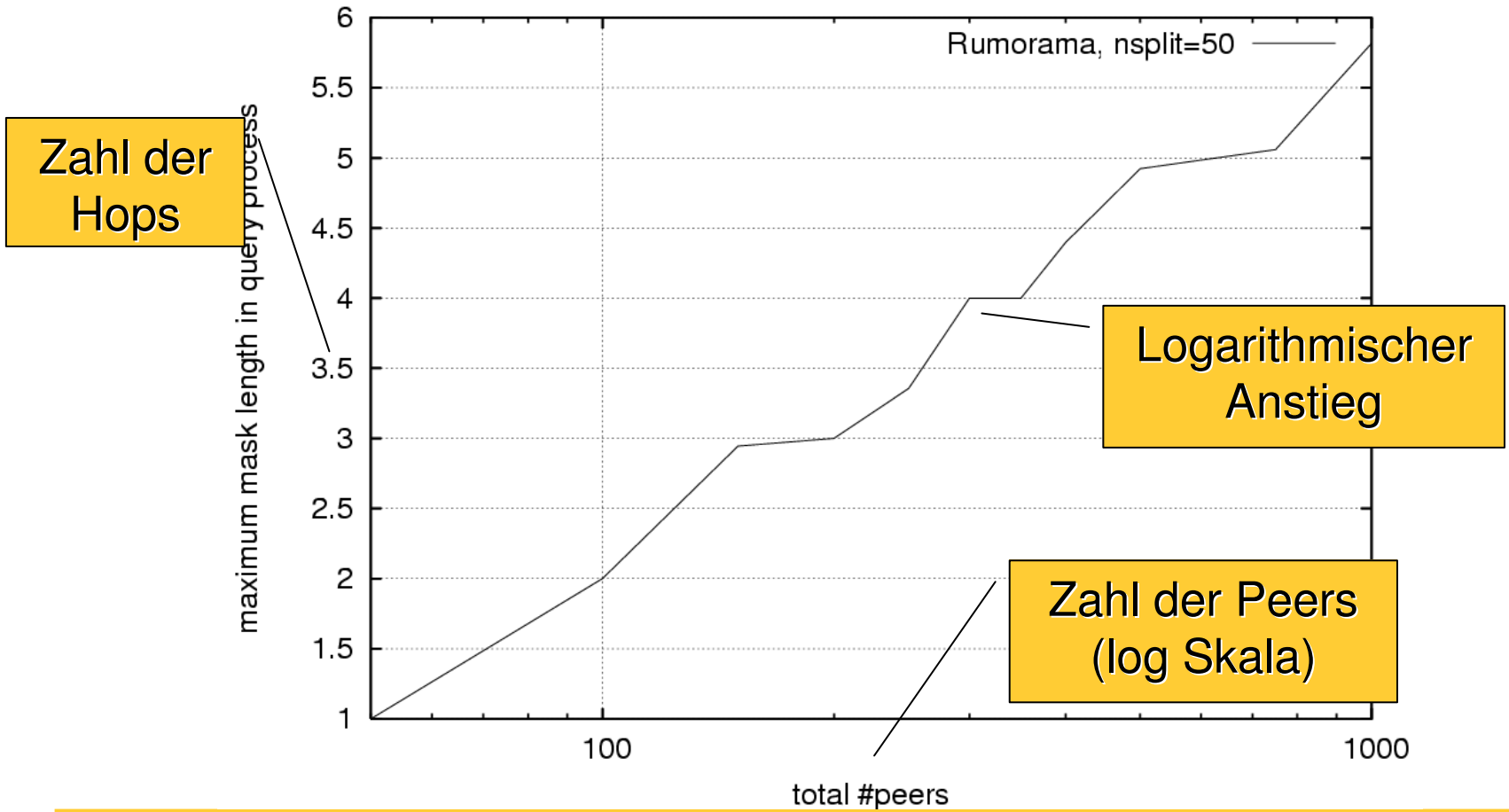
Kommunikationsaufwand (Versandte Zusammenfassungen)



Erreichte Zusammenfassungen/Churn



Erreichte Zusammenfassungen/Churn



Resultatqualität

- Siehe voriger Vortrag

Rechnung: Gesamt-Kosten für Anfrage

In Image Retrieval realistisch

> Annahme:
5% der Peers werden kontaktiert

➔ 1.000.000 * 0.05 = 50.000 Knoten

➔ 50.000 * 700 Byte = 35.000.000 Byte

Kosten für einzelnen Peer

- Innerer Knoten:
  - Anfrage wird an 2 Knoten weitergegeben
  - ➔ 2 * 700 = **1.400 Bytes** versandt
- Knoten im Blattnetz:
  - Anfrage wird an 5% der Knoten weitergegeben
  - Je nach Blattnetzgröße
    1.750 Byte (5% * 50 Peers)
    bis
    35.000 Byte (5% * 1.000 Peers)

➔ **Sehr gute Lastverteilung** zwischen Knoten

Anstehende Verbesserungen

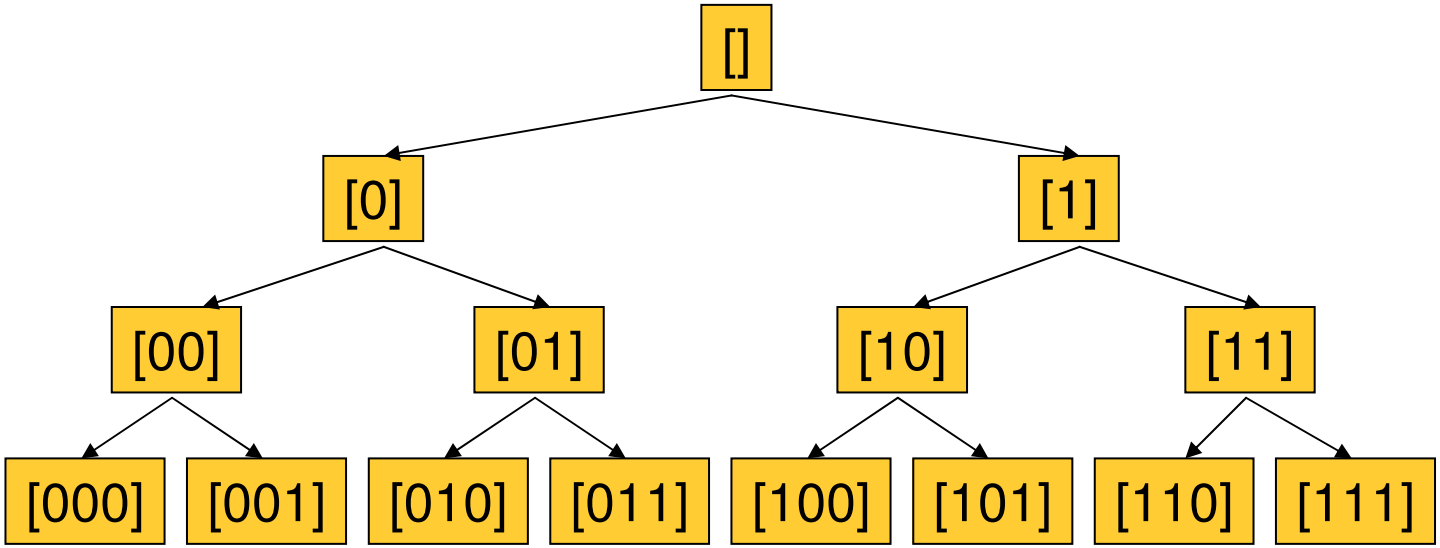
- Verbessern der Netzwerkaspekte
  - Inhaltlich ähnliche Knoten zu Freunden machen
  - Technisch nahe Knoten (z.B. gleiches Subnetz) zu Freunden machen
- Verbesserungen der Selektivität
  - Hybrider Ansatz (EMH2005, wie vorgetragen)
  - Aggregate von Zusammenfassungen
  - Hierarchische Ansätze:
    auch Blattnetze auswählen

Aggregate

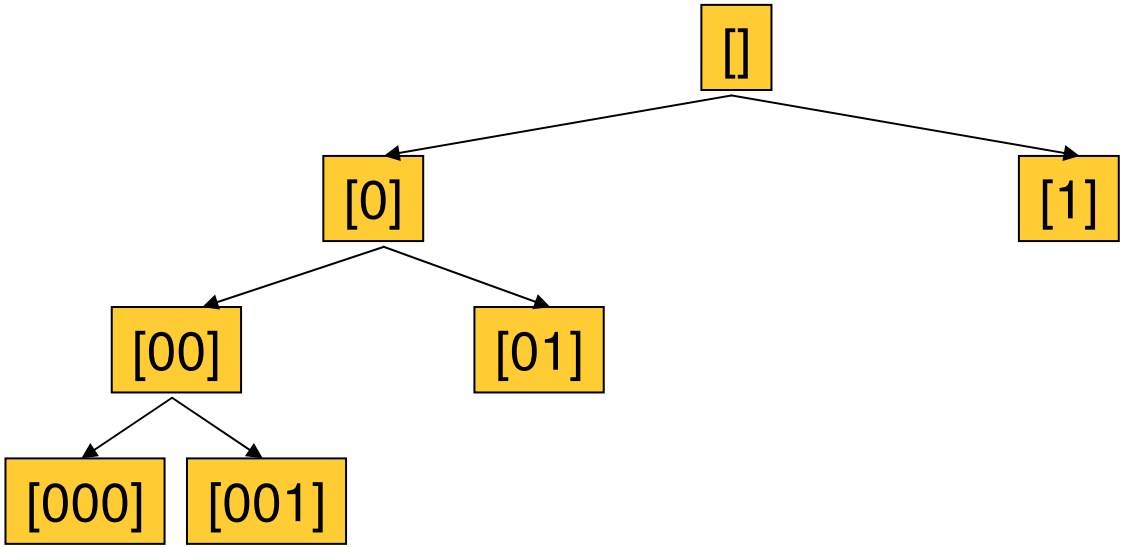
- Betrachten Hierarchie von Netzen zu Masken [],[0],[1] etc…
- Bauen Zusammenfassungs-Aggregate rekursiv von unten her auf

Aggregate

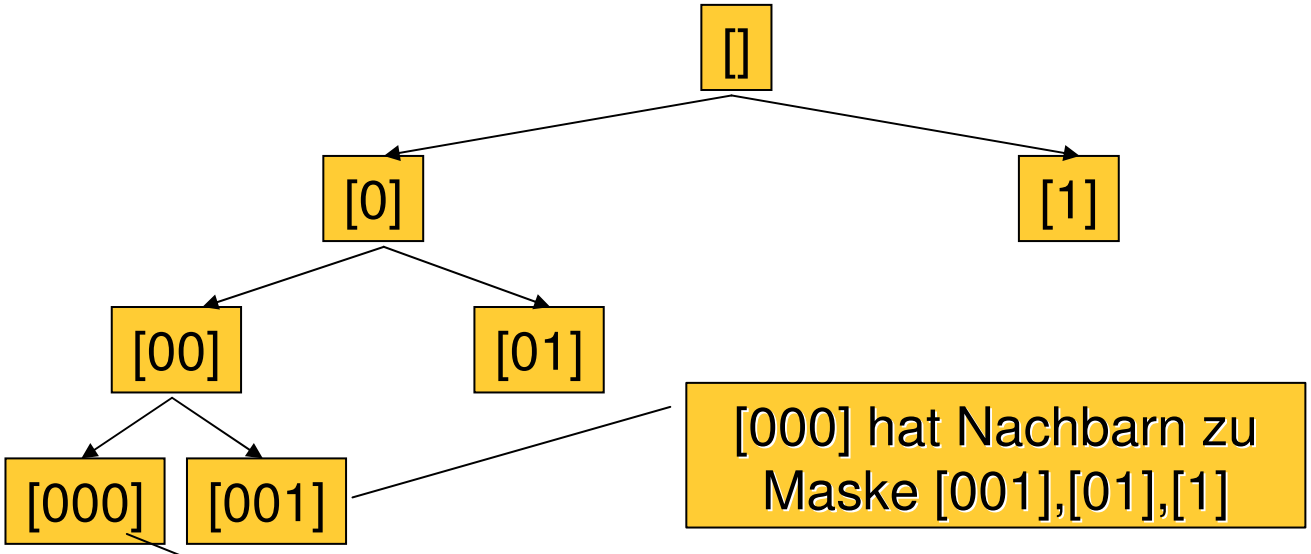
- Betrachten Hierarchie von Netzen zu Masken [],[0],[1] etc…:



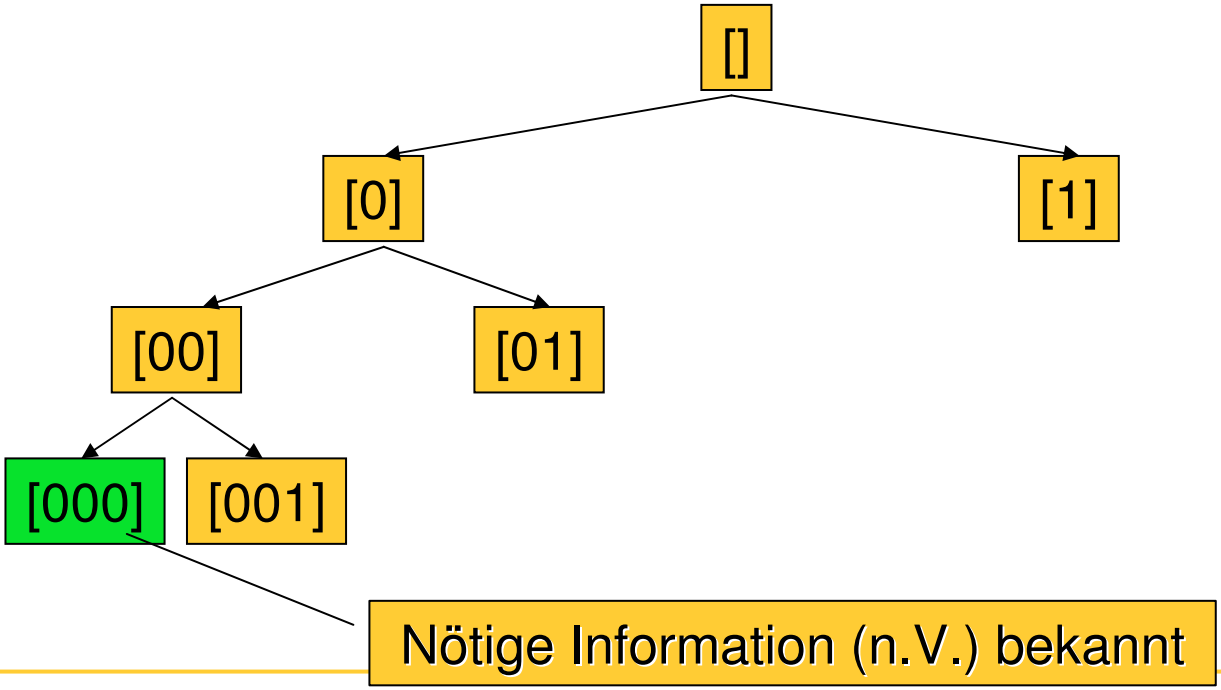
Aggregate: Nachbarnstruktur von [000]



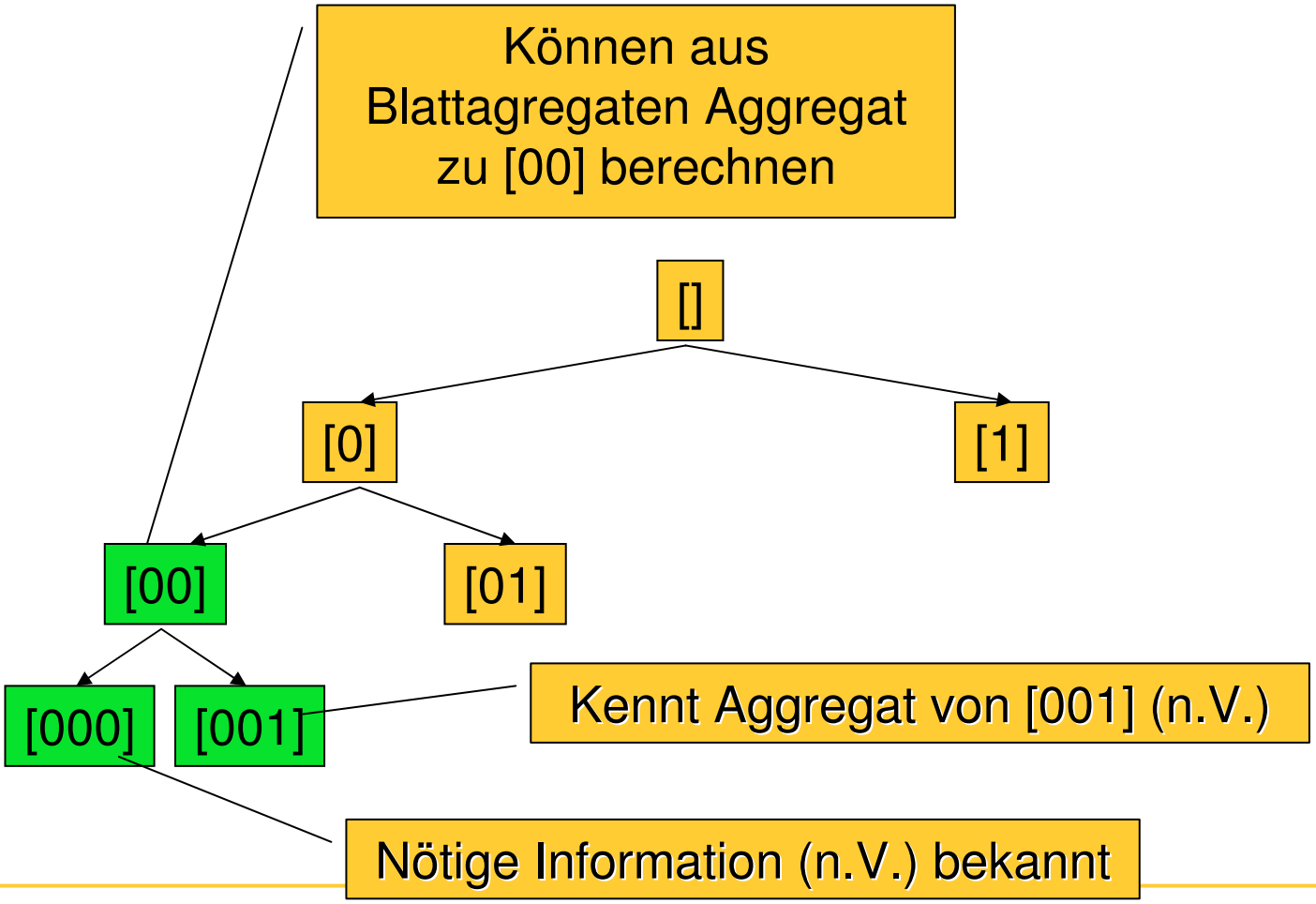
Aggregate: Nachbarnstruktur von [000]



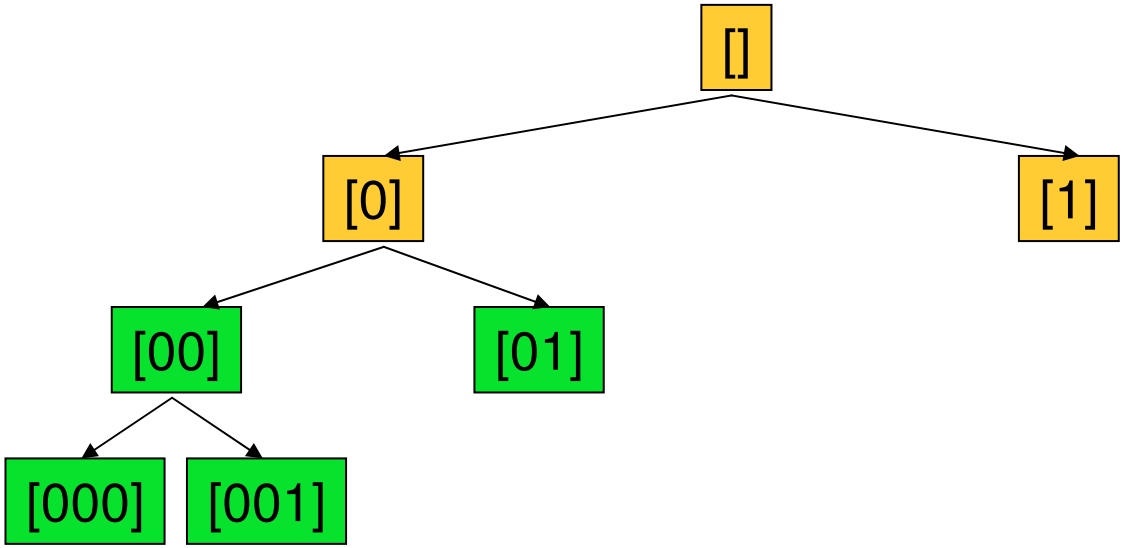
Aggregat zu [000]



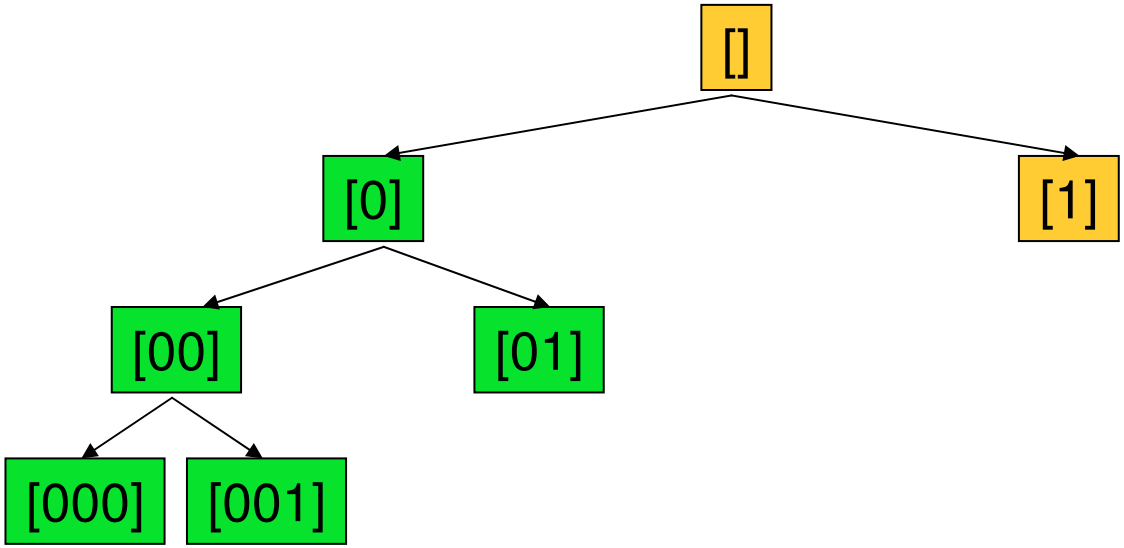
Aggregat zu [00]



Aggregat zu [01]: Analog



Aggregat zu [0]: Analog



Details

- Aggregate jeweils als Teil des Neighbor Spreadings verteilen
- Herausforderungen:
  - Zahl der übertragenden Aggregate minimieren
  - Qualität der Aggregate maximieren
- Stand: Experimente laufen

Zusammenfassung

- Rumorama skalierbare Variante von PlanetP
- Zusammenfassungsbasiertes Retrieval
- Skalierbarkeit
- Load Balancing
- Interessante neue Forschungsprobleme

Vielen Dank für Ihre Aufmerksamkeit.