Willkommen bei Verteilte Systeme!

Von Datenbanken
über Webdienste
bis zu p2p und Sensornetzen.

Heute: **Peer-to-peer in der Praxis – wo und wie sich Verteilung lohnt(-e)**.

> *Wer nicht aus der Vergangenheit lernt, ist verdammt ihre Fehler wiederholen, mit weniger Zeit, denn „die Probleme sind ja schon gelöst".*

Darum praktisch erprobte p2p-Netze verstehen

> *after a few days (and especially nights) of nervous full-site tinkering, it turned a 40 minute deploy process into one that lasted just 12 seconds!*[1]

- Bittorrent-Deployment: https://vimeo.com/11280885

Spoiler: Cut-through routing.

[1] https://web.archive.org/web/20120807165933/http://engineering.twitter.com/2010/07/murder-fast-datacenter-code-deploys.html

Mein Ziel

Ich will, dass Sie die Fähigkeiten erwerben, unter denen zu sein, die die Deployment Zeit um Größenordnungen verringern, ohne dabei die Kosten dafür zu zahlen, Torrents als Blackbox zu sehen.

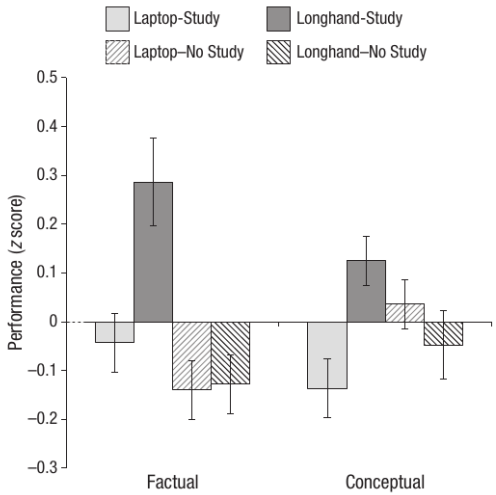
Torrent Bezeichnung für eine BitTorrent-Datei oder eine von BitTorrent verwaltete Datei.

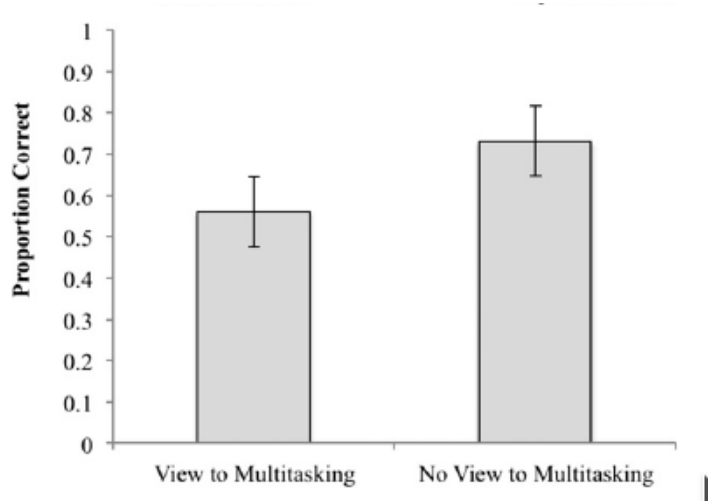
BitTorrent Ein p2p-System zum Verteilen großer Datenmengen, bei dem die Verwaltung auf zentralisierten Trackern läuft

Darum ich

- Seit 2004 in p2p-Entwicklung
- Seit 2013 mit Kompetenz :-)
- Aktuell Release-Manager des Freenet/Hyphanet Projektes

Laptops: Eigenverantwortlich





"even when laptops are used solely to take notes . . . their use results in shallower processing."
— Mueller and Oppenheimer (2014)

"Laptop multitasking hinders classroom learning for both users and nearby peers"
— Sana et al. (2013)

Laptops

- Möglichst zu

Projekte

- Notieren Sie bitte Ideen
- Modulplan: 39h Selbststudium
- Projekt interessanter und nützlicher als Klausur
- Projektideen sammeln.

Doing X with libp2p oder libresilient?

Auf einem der aktuellen NLnet-Projekte aufbauen?

Vorträge

- 5 Minuten pro Person, Gruppen möglich.
- Weitere Ideen: Übersicht über einen FOSDEM 2023-Vortrag.

Wiederholung 1

- Sammlung autonomer Knoten, die als ein kohärentes System erscheinen.
- **Ziele**: Resourcen, Verteilungstransparenz, Skalierbarkeit
- **Skalierung**: Größe, Geographie, Administration
- Latenz, Partitionierung, Replikation, Caching
- **Fallacies!**
- Cluster, Grid, Cloud, Ubiquitous, Mobile, DIS, Sensornetze

Wiederholung 2

- **Architektur**: Layered, Object, Resource, Event
- Schichten und Overlay Netze
- Prozesse sind isoliert, Threads teilen Speicher.
- **Middleware** als Schicht: Übernimmt Verteilung, gibt Garantien.
- **Messaging**: Request-Reply, Pub-Sub, Pipeline.
- **Overlay** metriken: Link Stress und Stretch

Fallacies of distributed Systems

1. The network is reliable
2. The network is secure
3. The network is homogeneous
4. Topology does not change
5. Latency is zero
6. Bandwidth is infinite
7. Transport cost is zero
8. There is one administrator

1. Hard disks don't fail
   Files stay intact
2. Power is stable
3. IPs are reachable
4. Constant factors are negligible
5. APIs stay compatible
6. Textfiles are simple

Ablauf heute

- Grundprobleme
- Gnutella (das Erste verbreitete, komplett verteilte p2p-Netz)

`--- PAUSE 14:30 ---`

- Kademlia (das am weitesten verbreitete DHT)
- BitTorrent
- Freenet/Hyphanet
- Weiteres (Aktuelles, WebRTC, . . .)

Was und warum?

Was?

peer-to-peer (p2p) peers (gleichberechtigte Partner) arbeiten zusammen, um sich gegenseitig einen Dienst zu erbringen.

Warum?

Sie haben ein unerwartet beliebtes Programm geschrieben. Jetzt wollen es 100 Millionen Leute herunterladen. Größe: 50GiB. Wie viel kostet die Verteilung?

Ziele heute

- Sie kennnen die zentralen Herausforderungen der Praxis:
  - Einstieg
  - Suche
  - Inhalte verbreiten
  - Kommunikation
- Sie können in Bezug auf zwei Herausforderungen die Eigenschaften von je zwei p2p-Systemen beschreiben, die sich in der echten Welt bewährt haben.
- Sie können einschätzen, ob ein bestimmtes Konzept diese Herausforderungen bestehen könnte.

Welche p2p-Netze kennen Sie

am FlipChart sammeln

Ziele für Grundprobleme

Sie können die Grundprobleme beschreiben, die Peer-to-Peer-Netze lösen müssen:

Fundamente

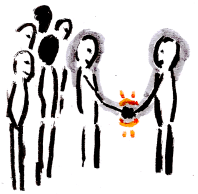
- Einstieg
- Suche
- Verbreitung

Aufbauend

- Kommunikation
- Störungsresistenz

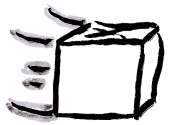
Grundprobleme in Peer-to-Peer-Netzen

- **Einstieg**: Wie finde ich meinen Platz im Netz?



- **Suche**: Wo gibt es, was ich brauche?

- **Verbreitung**: Wie vermeide ich Flaschenhälse?



- **Kommunikation**: Wie fließen Informationen durchs Netz?

- ***Störungsresistenz***: *Wie skaliert Gewünschtes besser als Unerwünschtes?*

Warum p2p?

Skalierbarkeit Ein einzelner Server bricht bei etwa 100k Anfragen pro Sekunde ein. *dwd bei Sturm Sabine 2020?*

Mit Nutzung wachsen Ähnliche Infrastruktur für 1000 Leute oder 10 Millionen Leute

Infrastrukturkosten 100k€ pro Jahr = Entwickler oder Entwicklerin

Warum nicht p2p?

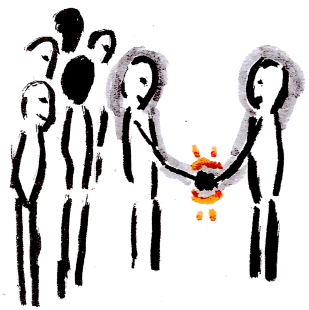
- Gestiegende Leistung von Servern. *Sturm: dwd[2] hielt größtenteils Stand (durch vereinfachte Seite[3])*
- Handies sind durch Batterie und Netz begrenzt $\rightarrow$ keine kontinuierliche Leistung. (Nachts möglich?)
- Viele der einfachen Lösungen unmöglich, z.B. Geld auf Probleme werfen.

[2] dwd: Deutscher Wetterdienst.

[3] $\Rightarrow$ gibt es eine einfachere Lösung?

Einstieg: Wie finde ich meinen Platz im Netz?

- **Erste Adressen**: Wie finde ich Adressen anderer Knoten?
- **Wahl der Verbindungen**: Mit wem sollte ich mich verbinden?
- **Routing-Informationen**: Welche Daten brauchen die Knoten?



Strukturiert vs. Unstrukturiert

Strukturiert

- **Erste Adressen**: Braucht Topologie[4]
- **Wahl der Verbindungen**: Nur bestimmte sinnvoll
- **Routing-Informationen**: Durch Auswahl der Partner (Peers)

Unstrukturiert

- **Erste Adressen**: Einfache Liste
- **Wahl der Verbindungen**: Beliebige Andere
- **Routing-Informationen**: Explizit austauschen

Kann ich alle direkt erreichen?

[4] Topologie: Struktur des Netzes.

Suche: Wonach suchen?

- **Schlüsselwort**: Gnutella, Skype (vor MS)
- **Inhalts-Hash**: Kademlia, BitTorrent VHT, Freenet
- **Öffentlicher Schlüssel**: Freenet

Glossar:

BitTorrent VHT Verteilte Hashtabelle, ein DHT

DHT Distributed Hashtable

Öffentlicher Schlüssel public key, das Gegenstück zum privaten Schlüssel in asymmetrischer Verschlüsselung.

Suche: Wo gibt es, was ich brauche?

Zwei Konzepte:

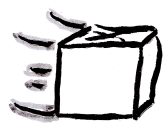
- Pfade zu exisiterenden Daten finden: Gnutella
- Daten an die richtigen Orte bringen: Kademlia, BitTorrent VHT, Freenet

Schlüssel zum Licht

Verbreitung: Wie vermeide ich Flaschenhälse?

- Zentralisiert: Streaming im Provider-Netz via Multicast
- Swarming: Nutzer übernehmen einen Teil der Verteilung
  - Koordiniert von zentraler Stelle: BitTorrent (Tracker)
  - Koordiniert durch die Nutzer: Gnutella (Download-Mesh)
  - Unabhängig verteilte Fragmente: Freenet[5]

Glossar:



Download-Mesh Name des Protokolls

Tracker Ein Server der den BitTorrent-Schwarm Koordiniert

[5] Reduziert Swarming auf Download einzelner Dateien, braucht aber caching: Zeitlich begrenzte Zwischenspeicherung.

Kommunikation

- Vier-Augen Gespräch (PM/DM/msg/Anruf/...)
- Unterhaltung in Gruppen (Chat, Forum, Videokonferenz, ...)
- Öffentliche Unterhaltung
- Von neuen Inhalten erfahren
- Informationen über Inhalte (Kommentare, Bewertung, ...)

Störungsresistenz: Wie skaliert Gewünschtes besser als Unerwünschtes?

Störung

"Disruption", alles, was den die Qualität des Dienstes für die Nutzenden verringert

Störungsresistenz: Wie skaliert Gewünschtes besser als Unerwünschtes?

Störung

"Disruption", alles, was den die Qualität des Dienstes für die Nutzenden verringert

In Gruppen sammeln

- **Knotenauswahl**:
- **Suche**:
- **Verbreitung**:
- **Kommunikation**:

Störungsresistenz: Wie skaliert Gewünschtes besser als Unerwünschtes?

Störung

"Disruption", alles, was die Qualität des Dienstes für die Nutzer verringert

Auf jeder Ebene nötig

- **Knotenauswahl**: Verbindung mit Angreifern
- **Suche**: Spam, Falschinformationen
- **Verbreitung**: Dateien korrumpieren
- **Kommunikation**: Spam, Belästigung und Zensur[6]

[6] „Das Web betrachtet Zensur als Störung und lenkt Anfragen darum herum."
— The Internet treats censorship as a malfunction and routes around it. – John Perry Barlow

Störquellen

Sammeln am Flipchart

[7] Werbung ist Spam durch die genutzte Plattform.

Störquellen

Sammeln am Flipchart

Quellen

- **Parasiten**: Bessere Leistung auf Kosten Anderer (leecher).
- **Trolle**: Kein Finanzinteresse, minimale Resourcen, nutzen jegliche Lücke.
- **Spammer**: Erfolg durch Verbreitung eigener Inhalte.[7]
- **Konkurrenten**: Erfolg durch verringerte Qualität des Systems.
- **Angreifer**: Erfolg durch Schädigung von Nutzern.

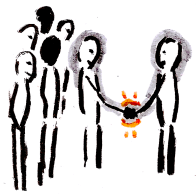
[7] Werbung ist Spam durch die genutzte Plattform.

Weitere Eigenschaft: Grad der Verteilung

Serverkoordinierte Teilgruppen bis vollständig dezentrale Interaktion.

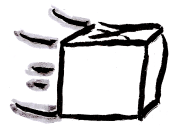
Zusammenfassung

- **Einstieg**: Erste Adressen und Routing-Info



- **Suche**: Schlüsselwort, Inhalt, Public Key

- **Verbreitung**: Tracker, Download-Mesh, Fragmente mit Caching



- **Kommunikation**: Privatnachricht, Forum, Neuigkeiten, Kommentare

- ***Störung***: *Parasiten, Trolle, Spammer, Konkurrenten und Angreifer*

Gnutella

Erinnert mich bitte an die Pause

> *On March 14th, 2000, . . . an early version . . . with a note: "Justin and Tom work for Nullsoft, makers of Winamp and Shoutcast. See? AOL can bring you good things!" . . . AOL ordered him to take the program down immediately . . . calling Gnutella an "unauthorized freelance project." . . . hackers had gone . . . to reverse-engineer it . . . into the hands of the open-source community . . .*
>
> — *The World's Most Dangerous Geek; Interviewed by David Kushner; RollingStone.com; January 13, 2004.*

Ursprung der ersten Tauschbörse.[8] *Sie verschwand nach verlorenen Urheberrechtsklagen der Entwicklungsfirmen in der Obskurität und seine technischen Errungenschaften gerieten in Vergessenheit.*

[8] Tauschbörse: Ein Dienst, in dem Nutzerinnen und Nutzer Inhalte anbieten

Ziele

Sie verstehen die grundlegende Funktionsweise von Gnutella als Beispiel einer effizienten, dezentralen Schlüsselwort-Suche.

Sie erkennen, wo die für Gnutella entwickelten Techniken sinnvoll genutzt werden können.

Sie wissen, welche Probleme ungelöst blieben.

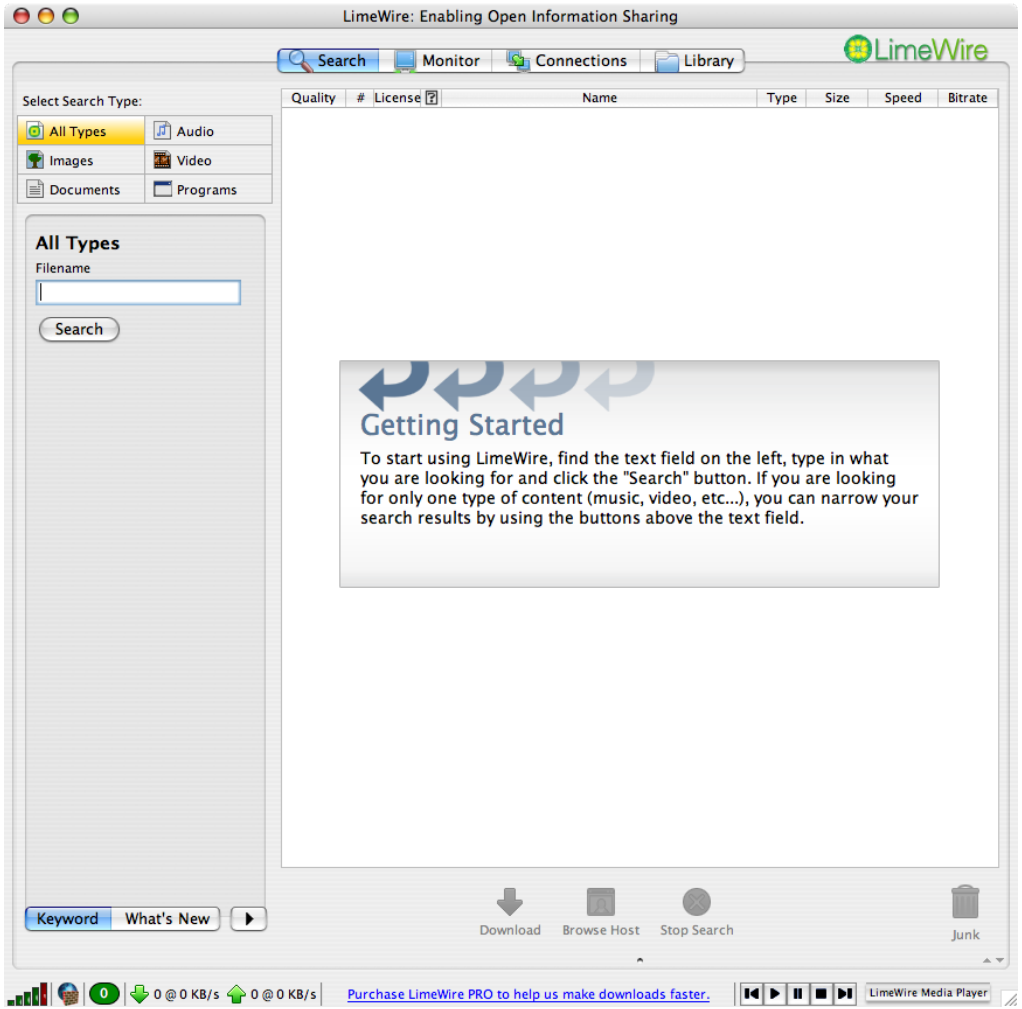
Inhalt

- **Nutzersicht**: Das war Gnutella
- **Einstieg**: GWebCaches
- **Suche**: Slow-Start + Keyword-Multicast
- **Verteilung**: Download-Mesh
- **Kommunikation**: Neues und Sammlung zeigen
- **Störungsresistenz**: Heuristik oder Inhalts-Matrizen

Sicht der Nutzer/-innen

- 50 Millionen Knoten
- Globale Suche nach Dateinamen und ID3-Tags
- Filter für Creative-Commons-Lizenzen
- Suche nach den neusten Dateien (What's New?)
- Downloads von vielen Quellen ohne zentrale Koordination
- Audio-Streaming um 2004 („Dateivorschau“)
- *LimeWire, Bearshare, Shareaza, Phex, gtk-gnutella, . . .*

Praktisch


LimeWire: Enabling Open Information Sharing
LimeWire
Search
Monitor
Connections
Library
Select Search Type:
All Types
Audio
Images
Video
Documents
Programs
All Types
Filename
Search
Quality
#
License
Name
Type
Size
Speed
Bitrate
Getting Started
To start using LimeWire, find the text field on the left, type in what you are looking for and click the "Search" button. If you are looking for only one type of content (music, video, etc...), you can narrow your search results by using the buttons above the text field.
Keyword
What's New
Download
Browse Host
Stop Search
Junk
0 @ 0 KB/s
0 @ 0 KB/s
Purchase LimeWire PRO to help us make downloads faster.
LimeWire Media Player

Implementierung (Grundlagen)

- **Adressen**: Webcaches
- **Verbindung**: HTTP-Handshake, dann Binär über TCP socket + out of band Antworten via UDP
- **Verbreitung**: HTTP + swarming
- **Kommunikation**: Via Suche

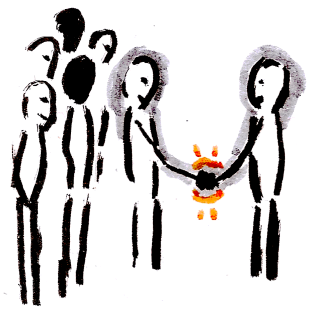
Einstieg: Webcaches

Ursprünglich

- Liste langlebiger Hosts einkompiliert

Final

- Liste der letzten guten Knoten
- UDP Host-Caches: Minimalserver, die IP-Listen sammelten und die neusten weitergaben
- Beispiel: GhostWhiteCrab[9]



[9] gwc resource: https://github.com/gtk-gnutella/gwc

Weitere Knoten finden: X-Try

Beim Handshake (wie HTTP):

When rejecting a connection, a servent MUST, if possible, provide the remote host with a list of other Gnutella hosts, so it can try connecting to them. This SHOULD be done using the X-Try header.

An X-Try header can look like:

```
X-Try:1.2.3.4:1234,3.4.5.6:3456
```

Weitere Knoten finden: Pong

Pong messages contains information about a Gnutella host. The message has the following fields

Bytes:	Description:
0-1	Port number. The port number on which the responding host can accept incoming connections.
2-5	IP Address. The IP address of the responding host. Note: This field is in big-endian format.

...

* When a Ping message is received (TTL>1 and it was at least one second since another Ping was received on that connection), a servent MUST, if possible, respond with a number of Pong Messages. These pongs MUST have the same message ID as the incoming ping, and a TTL no lower than the hops value of the ping.

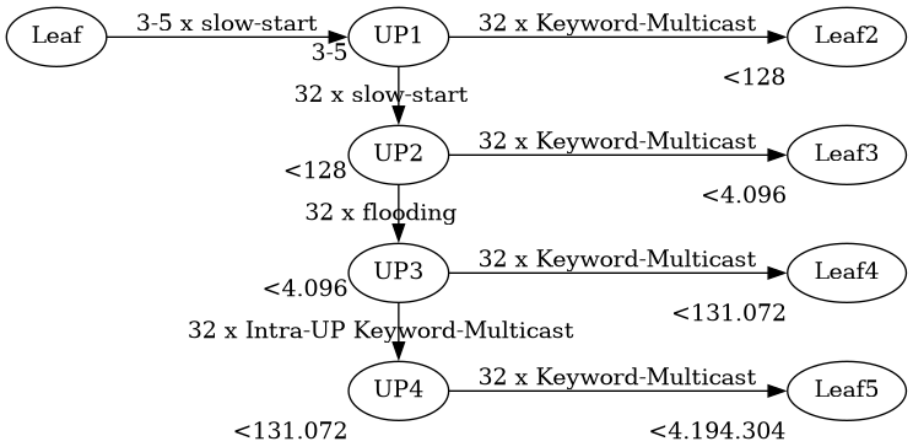
→ http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html

Suche abschicken

```
<15 bytes GUID>0x00
0x80 ; message type: Query
0x07 ; TTL: 7
0x00 ; Hops 0
0x00,0x00,0x09 ; payload length, max: 4kiB
0x00,0x00 ; min speed
test foo ; payload: search criteria
0x00 ; null-terminator, begins extensions
```

GUID Globally Unique ID. Zufällig erstellt, um Schleifen zu vermeiden.

Suche im Netz



Nicht existente Datei durchschnittlich: 4 x 32 x 32 = 4.096 Knoten.

Last (empirisch): <1kiB/s Leaf, <14kiB/s Ultrapeer

Ultrapeer (UP) Ein Hub, über den Kommunikation läuft.

Leaf Ein Randknoten, sucht über Ultrapeers.

Knoten Ein Peer oder Ultrapeer.

Suche 1: Slow-Start

„Dynamic Querying“ (DQ)

- Leaf fragt einen UP nach dem anderen. Stoppt nach „genug“ Ergebnissen (um die 100).
- UP fragt Leafs und andere UPs. Stoppt nach „genug“ Ergebnissen.

Suche 2: Keyword-Multicast

Query Routing Protocol (QRP)

- Suchwörter normalisiert:[10] lowercase, keine Akzente, ...
- Query Routing Table (QRT): Set mit schwachen Hashes von normalisierten Suchwörtern
- Automatisch hochskaliert für gewünschten Füllgrad

Intra-Ultrapeer-QRP:

- Vereinigung der Tabellen

Ähnlich: Bloom-Filter

[10] ungelöst: Japanische oder Chinesische Zeichen.

Größe der Query Routing Tabellen in Gnutella

- Hashes: Normalisierte Suchwörter in der Suchanfrage oder im Dateinamen
- Größe: Variabel, Default in LimeWire 128kiB, interpolation auf größere und kleinere Tabellen möglich.
- Aktuell verfügbare Quelle: BitSetQRTTableStorage.java
- Hash-Funktion pro Suchwort: HashFunction.java

Suche 4: Dateien nach Hash finden

- Zugriff auf Magnet-Links[11] brauchte exakte Dateisuche.[12]
- Angepasstes Kademlia ⇒ im Abschnitt zu Kademlia.

[11] Magnet-Links liefern Infos für Downloads in leicht kopierbarem Link.

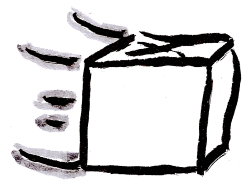
[12] kt=. . . : Suchanfrage, wurde kaum genutzt. Weiteres:
https://en.wikipedia.org/wiki/Magnet_URI_scheme#Design

Verteilung in Gnutella: Out-of-Band

Ursprünglich auf dem Suchpfad zurückgereicht, aber:

- 5 Schritte
- Durchschnittliche Lebensdauer eines Knotens:[13] 2h
- => Abbruch nach durchschnittlich 24 Minuten

Daher: Download-Mesh, unabhängig von der Suche



[13] 2h als Lebensdauer sind erstaunlich persistent. Aktuelles bei Freenet.

Verteilung: Download-Mesh

- Standard HTTP Range-Requests
- Content-Addressed: HOST/uri-res/raw/urn:sha1:HASH[14]
- 5 zusätzliche Header:[15]

X-Alt Bestätigte Quelle für die Datei, IP/Port

X-NAlt Unerreichbare Quelle oder Quelle mit Korrumpierten Daten. IP/Port

X-Gnutella-Content-URN Merkle-Tree Root-Hash

X-Thex-URI /uri-res/N2X?urn:sha1:HASH;MERKLE_TREE_ROOT

X-Available-Ranges bytes 0-10,20-30 (Beispiel)

[14] https://www.ietf.org/rfc/rfc2169.txt und http://www.nuke24.net/docs/2015/HashURNs.html

[15] http://rfc-gnutella.sourceforge.net/src/Partial_File_Sharing_Protocol_1.0.txt

Kommunikation: Schwachstelle

- Chat nie wirklich verlässlich
- Kein bleibender Kontakt zu anderen
- Funktionierend:
  - Was gibts Neues? (via LimeWire: Neueste Dateien sehen)
  - Sammlung durchsuchen (Alle freigegebenen Dateien sehen)

Störungsresistenz: Heuristiken als Spam-Filter

Ähnlich wie E-Mail-Spamfilter.

Hat Spam auf 10-20% der Ergebnisse reduziert.[16]

[16] Ginge so ein Webshop? Wer braucht welche Garantien?

Störungsresistenz: Objektvertrauen via Credence

- Jede korrekt bezeichnete Datei: 1.0
- Jede inkorrekt bezeichnete Datei: -1.0
- Wertungen anderer mit Korrelation der gemeinsamen Wertungen multipliziert.

→ `http://credence-p2p.org`

Wurde nie in ein Mainstream-Prgogramm übernommen.

Verbleibende Schwächen 2008

- 10-20% Spam-Ergebnisse trotz 50 Millionen Nutzer.
  - Credence nie weitverbreitet.
- Ein Schritt Flooding: Windows begrenzte Verbindungszahl.
- Parameter-Anpassungen beim Wachstum.
- Keine Kommentare, Peer-Chat wurde nie gut.

Das Ende von Gnutella

2010: Die Zeit von Gnutella endete nicht durch technische Grenzen, sondern durch Klagen der Musikindustrie. Der Besitzer von LimeWire haftete mit seinem Privatvermögen. LimeWire bleibt freie Software, aber ohne große Verbreitung.[17]

Die entwickelte Technologie gerät in Vergessenheit.[18]

[17] Die Geschichte von LimeWire: https://melmagazine.com/en-us/story/an-oral-history-of-limewire-the-little-app-that-changed-the-music-indu

[18] Teile der Spezifikation: http://rfc-gnutella.sourceforge.net/rfc-gnutella.zip und https://web.archive.org/web/20070429042042/http://www.the-gdf.org/index.php?title=Main_Page

Gnutella Routing Experiment

- Peers: Tisch + davor + dahinter
- Letzte 2 Hops
- Suche nach Namen
- Hash = 1. Buchstabe
- QRT[19]: Hash der Namen der Peers
- Intra-UP QRT: QRTs der Peers, zusammengefasst

Was müsst ihr vorher austauschen?

[19] QRT: Query Routing Table.

Zusammenfassung Gnutella

- Effiziente Suche nach Schlüsselworten
- TCP-basiertes Binärprotokoll, 50 mio Nutzer, 1kiB/s Leaf, 14kiB/s Ultrapeer
- Einstieg: WebCache-Server + Austausch QRT (wie Bloom-Filter)[20]
- Suche: Slow-Start + QRT Routing
- Verteilung: Download-Mesh
- Störungsresistenz: Heuristik oder Objektbasiert

[20] Set von schwachen Hashes der Suchwörter, Anzahl keys dynamisch skaliert und interpoliert

Projektideen

- Download-Mesh implementieren
  - Nur Range-Requests + magnet für Quellen
  - Quellen-Gossip via XAlt[21]
  - Mit Merkle-Tree oder hashliste für chunks und mit XNalt
- Suche über WebRTC in Javascript
  - flooding über vereinfachtes Binärprotokoll
  - QRP / QRT
  - Sharing als Upload in local storage
  - GGEP: Generic Gnutella Extension Protocol; Binarprotokoll für beliebige Daten.

[21] XAlt/XNalt: Header, der gute / kaputte Quellen beschreibt.

PAUSE

--- PAUSE ---

Kademlia

Lookup in einer Verteilten Hash-Tabelle (DHT) mit xor-Metrik.

- Nutzersicht
- Suche
- Einstieg *(nutzt die Suche)*

Ziele

Sie verstehen die grundlegende Funktionsweise von Kademlia als Beispiel einer effizienten, dezentralen Hash-Suche.

Sie erkennen, wo die in Kademlia entwickelten Techniken sinnvoll genutzt werden können.

Sicht der Nutzer/-innen

Werkzeuge

Ursprünglich Tauschbörsen: Kad in aMule, VHT in Torrent clients
Amazon Dynamo verwendet das sehr ähnliche Chord.

Anwendung

- Suche nach exakten Dateien
- Löst Magnet-links auf
- Server-Auswahl zum Schreiben; eventual consistency

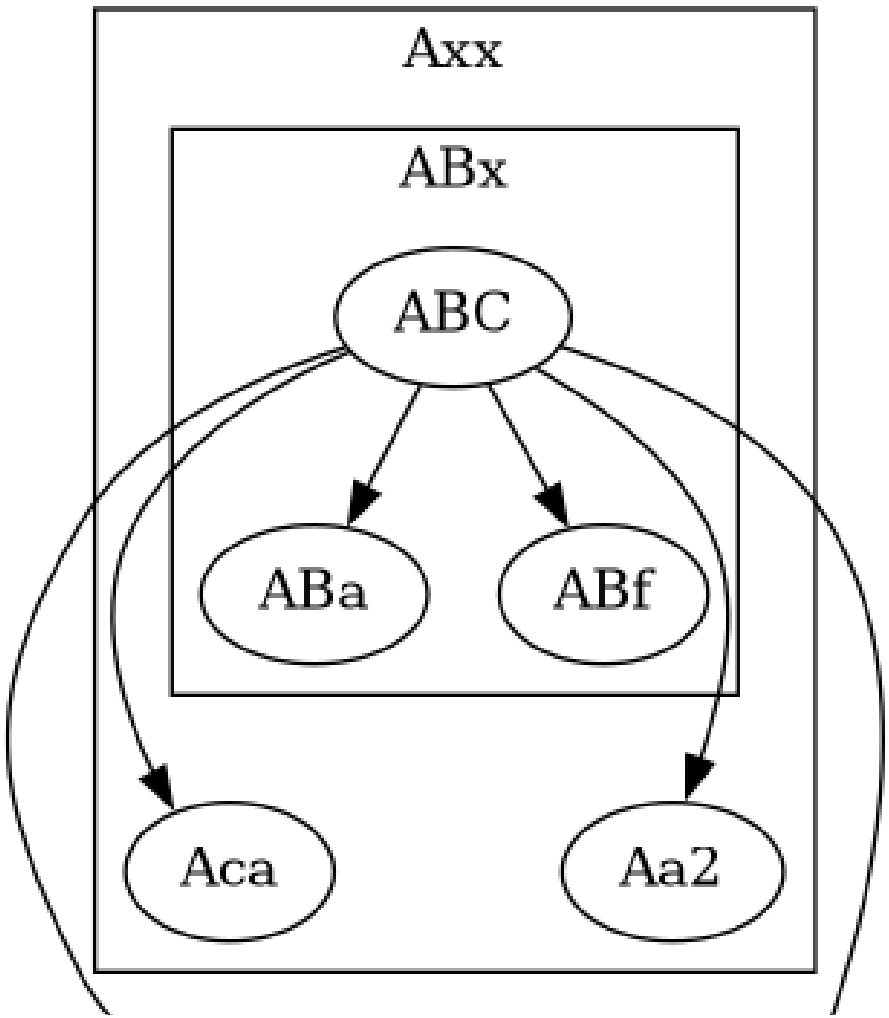
Suche in Kademlia

- Jeder Knoten hat eine zufällige ID
- Suche nach Hash $\rightarrow$ Distributed Hash Table
- Distanz zwischen Hash und ID via **xor-Metrik**[22]
- Schritt für Schritt in O(log(N)) zum richtigen Server

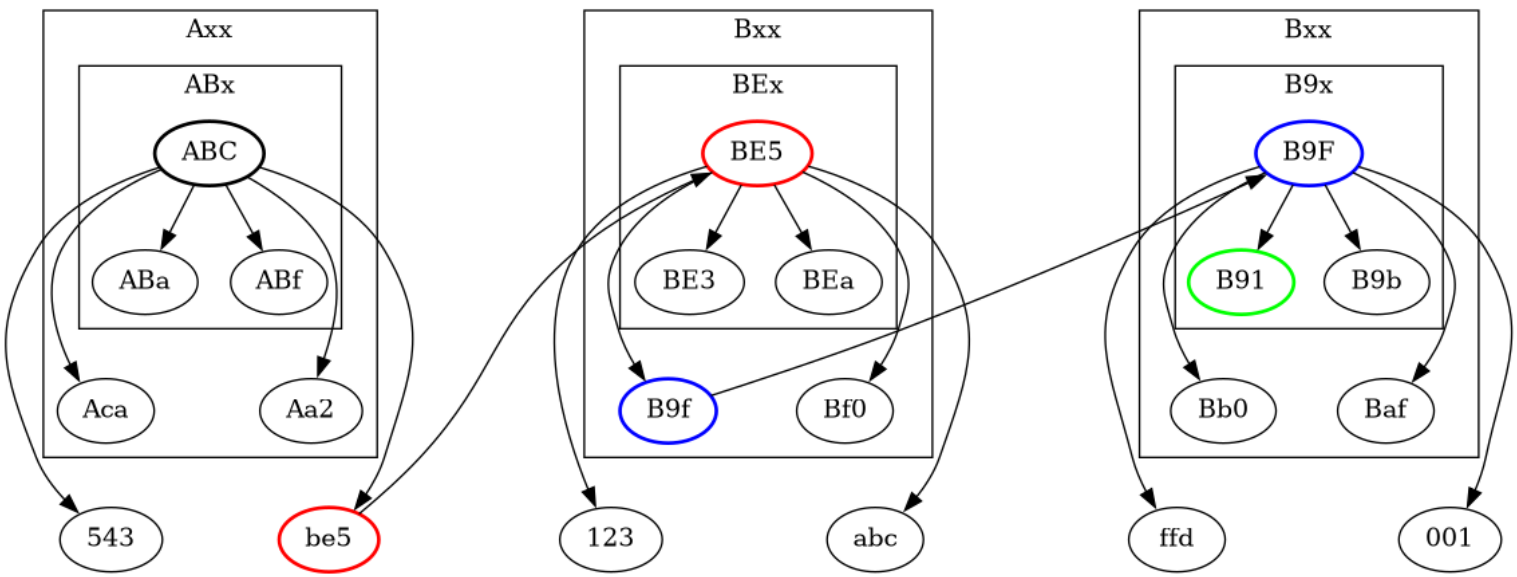
Ähnlich: Chord, Pastry.

[22] xor-Metrik: 4 xor 2 $\Rightarrow$ 100 xor 010 $\Rightarrow$ 110 $\Rightarrow$ 6.

Präfix-Buckets



Suche nach b91

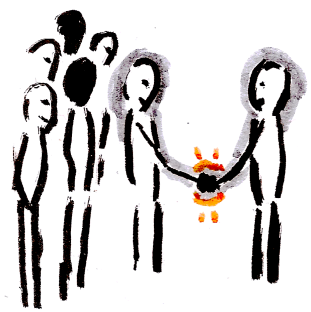


Speichern

- Suche nach Knoten nahe Hash.
- STORE: Hash + Wert.

Einstieg in Kademlia

- Kontakt zu mindestens einem bestehenden Knoten.
- Suche nach eigener ID: `FIND_NODE` (nah = zuständig für ID)
- Erhält Addressen + IDs der am nächsten liegenden Knoten
- Angefragte Knoten behalten auch die eigene Adresse und ID.



Kademlia Routing-Experiment, Einstieg

IDs nach Sitzplatz:

```
11 -   -   -   -   -   -   -   -     (Fenster)
10 -   -   -   -   -   -   -   -
01 -   -   -   -   -   -   -   -
00 -   -   -   -   -   -   -   -
   000 001 010 011 100 101 110 111
(Tür)           (Pult)
```

Präfix-Buckets (E=Eigener Bitwert):

1	4	4	2	1
EEEEx	EEExx	EExxx	Exxxx	xxxxx

Kademlia Routing-Experiment, Suche

- ID berechnen (vereinfacht: zufällig[23])
- Name in ID speichern.
- Andere Person: Name abfragen

[23] $\Rightarrow$ shared state, global; in Realität stattdessen: Hash.

Projektideen

-
-
-
-
-

Zusammenfassung

- Distanz: key-hash XOR node-ID
- Suche: Nächstgelegenen bekannten Knoten nach besseren Knoten fragen
- Kennt mehr nahe als entfernte Knoten
- Speichern wie Suchen
- Einstieg:
  - Suche nach eigener ID
  - Erreichte Knoten nutzen Adresse und ID

BitTorrent

- Verbreitetste Lösung für Swarming
- BitTorrent, IPFS, Blizzard-Updater
- Upload für schnelleren Download
- Koordiniert durch Tracker
- Keine Suche

Ziele

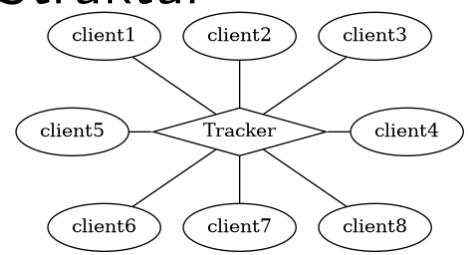
- Sie kennen die grundlegende Funktionsweise von BitTorrent.
- Sie verstehen, wo BitTorrent durch teilweise Zentralisierung Komplexität vermeidet.
- Sie können erklären, warum Torrent für Twitter keine optimale Wahl war, trotzdem aber Faktor 100 schneller, als die vorherige Lösung.

Sicht der Nutzer/-innen

- Informationen von Tracker-Seiten
- Download mit torrent-Datei oder Magnet-Link
- Unterstützt Ordner
- Heute: ipfs: Webseiten über Bittorrent
- NAT-Traversal und IP Verschleiern über Tor

Konzept von BitTorrent

Struktur

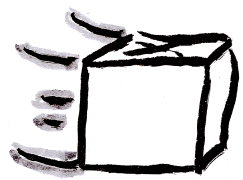


Tracker: Webseite

- Koordiniert Schwärme
- Suche, Foren, Bewertung, Verifizierung, Gemeinschaft
- Statistiken: Seeder, Leacher
- Liefert keine Daten
- Aggregiert, wer wie viel hochlädt -> Anreiz

Torrent-Datei

- Tracker URL(-s)
- Hashes für Chunks
- Namen der Datei(-en)
- Kann Ordner enthalten[24]



[24] http://www.bittorrent.org/beps/bep_0003.html

Anreiz zum Hochladen

- Uploadverhältnis wird geprüft
- Freeloader[25] verden von anderen clients gedrosselt (choked: niedrigere Downloadrate)
- In Literatur ist der Anreiz im Vergleich zu anderen Themen stark vertreten, in der Praxis sind die angebundenen Foren wichtig

[25] Freeloader: Leute, die nichts hochladen. Auch „Leech“. Gegenteil: „Seed“.

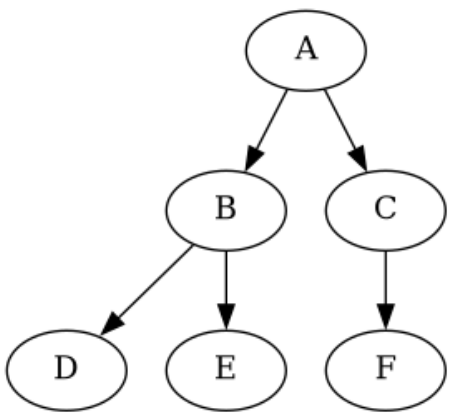
Weiteres

- VHT statt Tracker möglich (Kademlia)
- Freies Protokoll mit vielen Implementierungen
- Weiterentwicklung in der Community
- IPFS nutzt Torrents für dezentral gecachte Webseiten

Torrent für Twitter-Deployment

- Kosten bei Twitter: Übertragung über viele Schritte
- Torrent überträgt in Fragmenten.

Wunsch

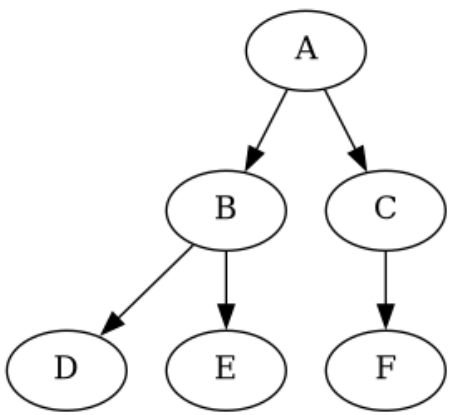


Cut-through routing / streaming.

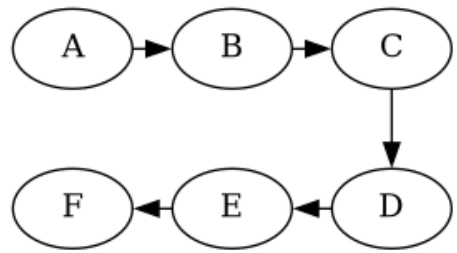
Torrent für Twitter-Deployment

- Kosten bei Twitter: Übertragung über viele Schritte
- Torrent überträgt in Fragmenten.

Wunsch



Wirklichkeit



cat . . . ssh tee . . .

Cut-through routing / streaming.

Projektideen

-
-
-
-
-

Zusammenfassung

- Tracker und Clients
- Tracker: Statistiken und Koordination
- Torrent-Datei mit Chunk-Infos

Freenet/Hyphanet

Zensur-Resistente Kommunikation auf Freund-zu-Freund Netzwerk

Dezentrale Datenbank mit pubkey-Zugriff

- Ziele
- Verwendung
- Einstieg
- Small-World
- Suche
- Verteilung
- Mutability
- Kommunikation
- Schnittstellen

Ziele für den Freenet-Abschnitt

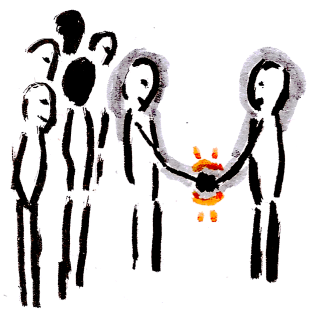
- Sie kennen Ähnlichkeiten und Unterschiede zwischen Kademlia und Freenet
- Sie erkennen feste Freund-zu-Freund Verbindungen
- Sie erkennen die Small-World-Anforderung
- Sie verstehen, wie Freenet Daten versioniert und neue Versionen findet, ohne existierende Daten ändern zu können
- Sie kennen die dezentrale Spam-Abwehr in Freenet

Verwendung

- Web-Schnittstelle
- Plugins mit E-Mail
- Externe Programme wie Chat und Foren mit Freenet als Datenbank via HTTP-ähnlicher API (FCP)

Einstieg in Freenet

- Opennet:
  - Ähnlich Kademlia: Wähle bekannte Seednode[26], Seednode sucht nach ID $\rightarrow$ Referenzen
  - Unterschied zu Kademlia: Nicht nur IP, sondern Referenz mit Schlüssel
- Friend-to-Friend:
  - Feste Verbindungen
  - Knoten tauschen ihre IDs, um das soziale Small-World-Netzwerk zu rekonstruieren
    $\Rightarrow$ Overlay-Kosten minimieren.

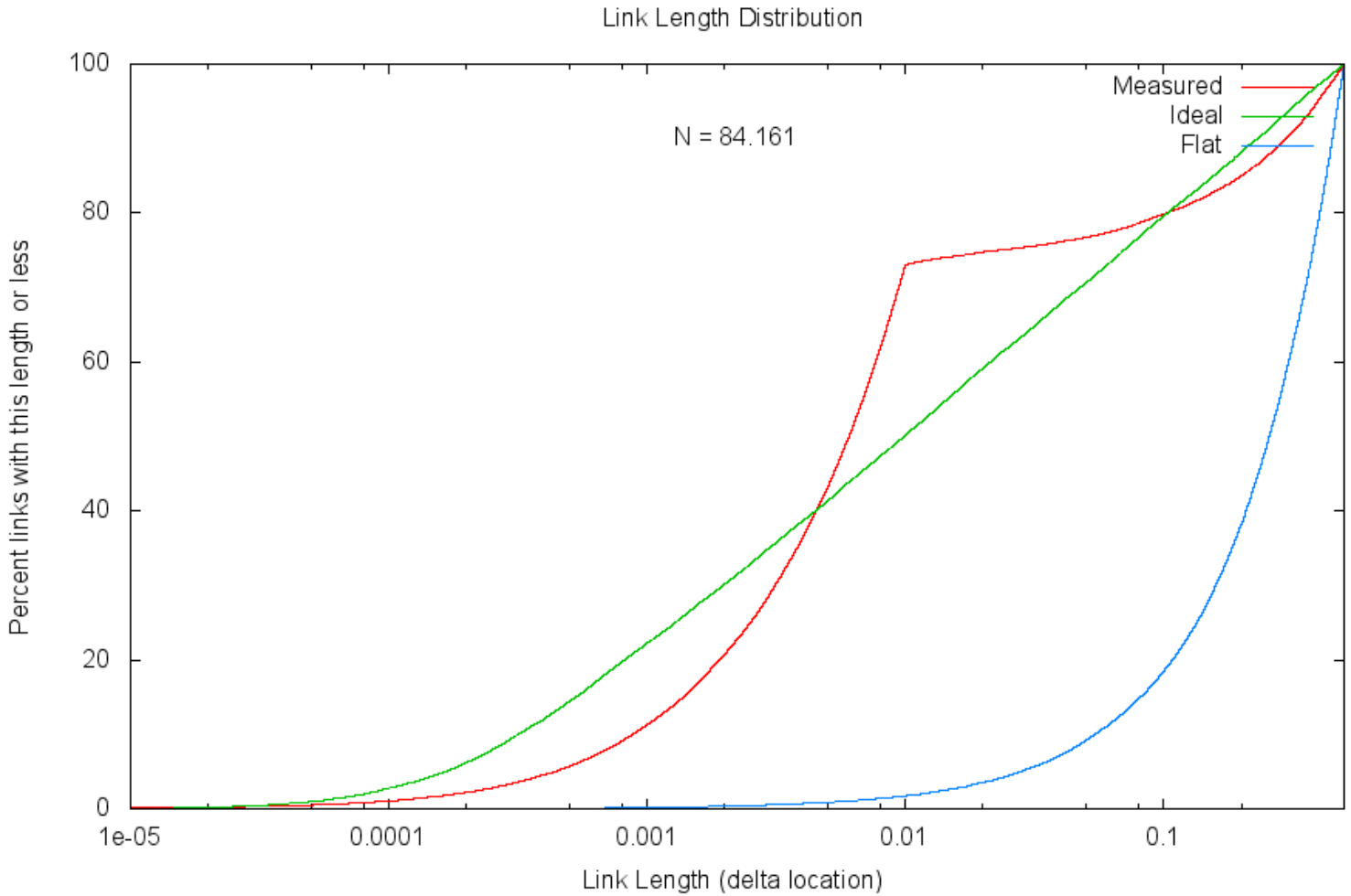


[26] Seednode: Bekannter Knoten, der Verbindungen zu anderen vermittelt.

Small-World-Netzwerk (skalenfreies Netzwerk)

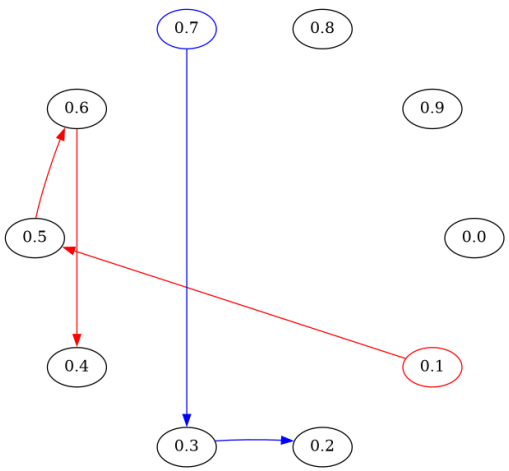
- Viele kurze und wenige lange Verbindungen.
- 6 deegrees of separation via Post: Unsere Bekanntschaften bilden ein small-world Netzwerk
- Kleinberg-Netzwerk: Wahrscheinlichkeit verbunden zu sein: $\frac{1}{d^x}$, d = Distanz, x = Dimension.
- Freenet: $x = 1$

Theoretische und gemessene Link-Längen



Freenet Suche

- Wie Kademlia, aber hop für hop weitergeleitet → keine Globale Erreichbarkeit oder Sichtbarkeit
- Suche nach public key möglich
- Keyspace: [0.0 : 0.1)



Arten von Schlüsseln

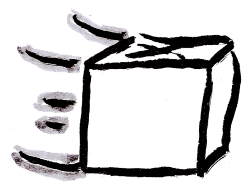
- CHK: Content Hash
- KSK: Keyword Subspace: Passwort
- SSK: Signed Subspace: Public Key
- USK: Updatable Subspace: SSK mit Version

Format:
`XXK@routing,encryption/tarball-name/path/to/file.ext`

Ohne Pfad und Name möglich (kleiner $\rightarrow$ Optimierung).

Verteilung in Freenet

- Netz speichert Inhalte → verteilter Cache
- Dateien verschlüsselt, als 32 kiB Fragmente mit 100% Redundanz abgelegt
- Manifest enthält Schlüssel der Fragmente als CHKs
- Effektiv LRU-Cache:[27]
  - Speichern überschreibt zufällig gewählte Fragmente
  - Zugriff stellt überschriebene Fragmente wieder her
- Upload auf existierenden Schlüssel+Pfad: Kollision → In der Praxis immutable



[27] LRU: Least Recently Used. Ältestes wird zuerst gelöscht.

Freenet als Datenbank

- Suche nach Public Key + Pfad
- → persönlicher Keyspace
- → tarballs für strukturierte Daten
- → pub-sub-Protokolle auf dezentraler Datenbank
- → Webseiten, Foren, Chat, ...

1 Minute Round-Trip-Time

Optimierung: Schlüssel Abonnieren, um 10k Schlüssel zu beobachten und Updates schnell zu sehen.

Mutability: O(1) Zugriff auf neuste Version

- Nutzende: `SSK@.../meine-seite-1/...` → `SSK@.../meine-seite-2/activelink.png`
- Optimiert: `USK@.../meine/seite/1`
  - `SSK@[key]/[sitename]-DATEHINT-[year]`

`HINT`
`46`
`2013-7-5`

DATEHINT-[year], DATEHINT-[year]-WEEK-[week], DATEHINT-[year]-[month], DATEHINT-[year]-[month]-[day]

Swapping: Friend-to-Friend wird Small World

9	4	**3**
5	2	8
1	6	7

6	**4**	9
5	2	8
1	3	**7**

3	4	9
5	2	8
1	**6**	7

6	7	9
5	2	8
1	3	4

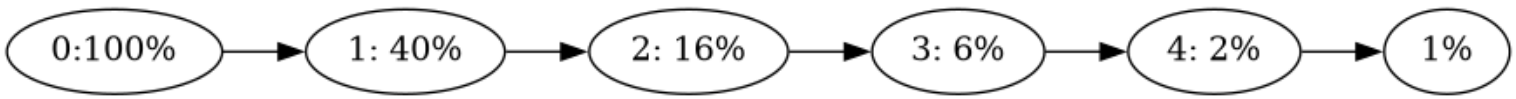
Spam-Abwehr

WoT (Web of Trust): Eine von zwei praktisch genutzten Möglichkeiten. Die andere ist FMS (Freenet Message System).

- ID = USK
- Trust -100 bis 100
- Rank: Distanz $\rightarrow$ capacity
- Score: Summe über alle Wertungen: trust * rank
- Skaliert bei 22 Nachrichten pro Tag und Person[28]

[28] https://www.draketo.de/english/freenet/deterministic-load-decentralized-spam-filter

Capacity



- Rank 1 40 %. rank 1: 100 trust, 40 Punkte als Score.
- Rank 2 16 %
- Rank 3 6 %
- Rank 4 2 %
- Rank 5 und niedriger: 1 %

Integer-Mathematik: `2 * 6 / 100 = 0`.

Schnittstelle: Web

- Browse
- Hoch-/Herunterladen
- Plugins
- Freund-zu-Freund-Nachrichten
- Lesezeichen mit Update-Info (5 min Latenz)

Schnittstelle: FCP

Async für Programme:

- Put/Putdir/Get
  - Password: `KSK@...`
  - Inhalt: `CHK@.../datei.endung`
  - Schlüssel: `SSK@pubkey/ordner/datei`
  - Updatable: `USK@pubkey/ordner/version/datei`
- Subscribe to key
- Plugins kontrollieren

Latenz in der Praxis

- Bis zu 1kiB, raw, realtime mode: <30s
- Große Dateien, im Manifest: ~5 min

Realtime

```
PriorityClass . 2 ;; high
MaxRetries . 0 ;; default: 10
RealTimeFlag . true
DontCompress . true
ExtraInsertsSingleBlock . 0
ExtraInsertsSplitfileHeaderBlock . 0
```

Bulk

```
PriorityClass . 3 ;; medium
RealTimeFlag . false
DontCompress . false
```

Kommunikation über Freenet

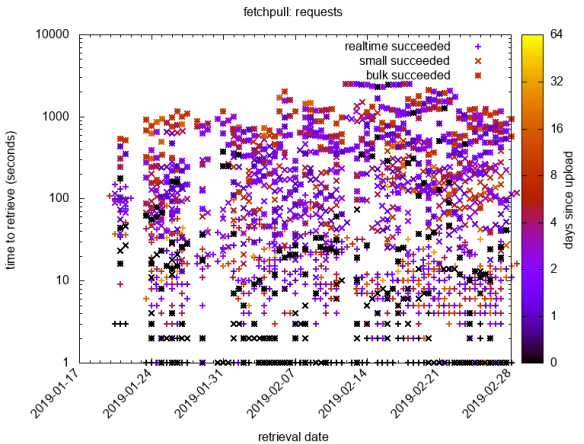
- Einstieg: Seed-keys + Captcha[29]-Queue: KSK-Prefix
- Suche: Nutzerspezifische Seiten mit Links, Update-Infos
- Verteilung: Gossip[30] keys, Dateien einfach hochladen
- Störungsresistenz: Web of Trust mit langsam steigender Sichtbarkeit

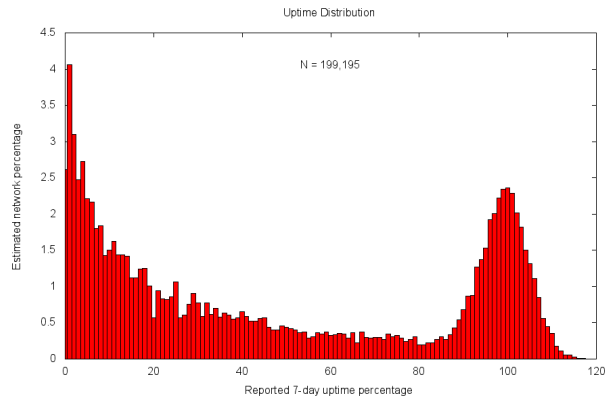
Autospawn node => Freenet als Backend, unsichtbar

[29] CAPTCHA: Meist Bilder, auf denen Zeichen erkannt werden müssen, um automatische Systeme auszuschließen.

[30] Gossip: Informationen während normaler Kommunikation von Knoten zu Knoten verteilen.

Stats


fetchpull: requests
realtime succeeded
small succeeded
bulk succeeded
time to retrieve (seconds)
retrieval date
days since upload
fetchpull: inserts
realtime succeeded
small succeeded
time to upload (seconds)
upload date
days until download


Uptime Distribution
N = 199,195
Estimated network percentage
Reported 7-day uptime percentage
Peer Count Distribution
N = 2,855
Percent of Reports
Reported Peers

Projektideen

-
-
-
-
-

Zusammenfassung

- Einstieg: Suche nach meiner ID bei Seednode
- Suche: Greedy Hash auf Small World
- Verteilung: Chunk-Tree mit Redundancy
- Kommunikation:
  - Einstieg: Seed-keys + CAPTCHA-Queue
  - Suche: Index-Seiten, Update-Erkennung
  - Verteilung: Dateien hochladen, Webseiten
  - Propagating Trust mit langsam steigender Sichtbarkeit

Verschiedenes

- Queuing
- Magnet-Links
- WebRTC
- Verstreutes
- Aktuelles
- Zusammenfassung

Queuing in p2p-Netzen

- Alle Upload-Warteschlangen sind immer voll
- Queuing-Strategien bestimmen, welche Dateien gut funktionieren
- FIFO Dateien: Große blockieren kleine (wie Alben in Musik-Spiellisten)
- FIFO Chunks: Overhead durch Ständigen Auf- und Abbau von Verbindungen
- HRRN: Große Dateien müssen warten, stört Preview
- Komplexer: Torrent-Superseed u.ä.

Magnet-Links

```
magnet:?xt=urn:bitprint:TIGER_TREE.SHA1
&xt=urn:btih:BITTORRENT_INFO_HASH
&xt=urn:sha1:HASH
&xl=LENGTH
&dn=NAME
&as=LINK_WITHOUT_HASH
&xs=LINK_WITH_HASH
&kt=SEARCH_STRING
```

Netzwerk-unabhängig, Link zu HTTP und p2p-Quellen, weitverbreitet

WebRTC

- Läuft im Browser (Javascript)
- Liefert Audio, Video, . . . , und **Peer-Socket**
- Erste Verbindung vom Server moderiert – erspart praktische Probleme
- p2p-Systeme, die nicht installiert werden müssen
- Beispiel: WebTorrent `https://webtorrent.io/`

Verstreutes

- Optimierung für Netzbetreiber oft angedacht: Clients im gleichen (Sub-)Netz bevorzugen Bei Gnutella „p4p“. Bei Pastry (Windows) laut Ghosh umgesetzt.
- Beispiel für non-greedy routing[31]: Random Walk in ants (Programm). Nicht durchgesetzt.
- Geld auf Probleme werfen: MaidSafe hatte 2000$ Hardwarekosten pro Monat. 2019 abgeschaltet.[32] Freenet hat $<$20$ pro Monat Kosten.

[31] Greedy-Routing: Anfragen mit rein lokaler Information an den am besten passenden Knoten weiterleiten.

[32] Quelle: https://www.reddit.com/r/safenetwork/comments/erpvee/dumb_question_is_safe_live/

Aktuelles

Was gerade jetzt passiert:

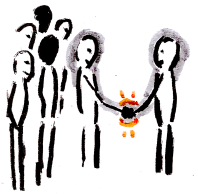
- Spritely Golem: p2p distributable content for the fediverse[33]
- Decentralized Internet and Privacy at FOSDEM[34]
  - DAT, GNUnet, Fediverse, Tor, ...
- In Karlsuhe: 21. Gulaschprogrammiernacht: https://entropia.de/GPN21 8. bis 11. Juni 2023

[33] https://gitlab.com/spritely/golem/blob/master/README.org

[34] https://fosdem.org/ — viele Vorträge zu decentralization, privacy, ...

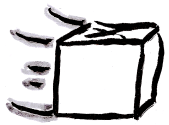
Zusammenfassung: Grundprobleme

- **Einstieg**: Wie finde ich meinen Platz im Netz?



- **Suche**: Wo gibt es, was ich brauche?

- **Verbreitung**: Wie vermeide ich Flaschenhälse?



- **Kommunikation**: Wie fließen Informationen durchs Netz?

- ***Störungsresistenz***: *Wie skaliert Gewünschtes besser als Unerwünschtes?*

Zusammenfassung: Implementierungen

	Einstieg	Suche
Gnutella	WebCache	Slow-Start + Keyword-Multicast
Kademlia	Suche nach eigener ID	xor-Hash-Hierarchie
BitTorrent	Tracker-URL	Kademlia / Tracker / Web
Freenet	Seed-Nodes suchen ID	Greedy Hash auf Small World
WebRTC	WebRTC Server	-

	Verteilung	Störung
Gnutella	Alt+NAlt, Range, Merkle-Tree	Heuristik/Credence
Kademlia	*unterschiedlich*	-
BitTorrent	Torrent	Wertung auf Tracker
Freenet	Chunk-Tree with Redundancy	Propagating Trust
WebRTC	-	-

Viel Erfolg beim Projekt!

Ich wünsche mir, dass einige von Ihnen
in 5 Jahren zurückblicken und sagen:

Was ich in verteilte Systeme über p2p-Netze gelernt habe,
war einer der Grundsteine meines Erfolges.

Verweise I

Mueller, P. A. and Oppenheimer, D. M. (2014). The pen is mightier than the keyboard: Advantages of longhand over laptop note taking. *Psychological Science*, 25(6):1159–1168. PMID: 24760141.

Sana, F., Weston, T., and Cepeda, N. J. (2013). Laptop multitasking hinders classroom learning for both users and nearby peers. *Computers & Education*, 62:24 – 31.

Bilder:

Merkle Tree Patent 1982
https://worldwide.espacenet.com/patent/search/family/022107098/publication/US4309569A?q=pn%3DUS4309569
Eingereicht 1979 als Methode Diffie-Authentication günstiger zu machen.