Adaptive Overlays in Peer-to-Peer Netzwerken

vorgelegt von
M.Sc.Wirtschaftsinformatiker
Alexander Löser

Von der Fakultät IV Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
– Dr.-Ing. –

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender: Prof. Dr. Sahin Albayrak
Berichter: Prof. Dr. Herbert Weber
Berichter: Prof. Dr. Wolfgang Nejdl

Tag der wissenschaftlichen Aussprache: 28. September 2005

Berlin 2005
D 83

Kurzfassung

Drei aktuelle Trends haben neue Perspektiven für die Recherche in Unternehmensdaten geschaffen: Eine Explosion lokal gespeicherter Daten, der Bedarf des Austausches dieser Daten in und zwischen einzelnen Unternehmen und ein zunehmender Kundenwunsch nach einer integrativen Suche in lokalen und entfernten Quellen. Alle drei Aspekte zusammen bewirken einen Marktwert für Dienste der Art 'Integrierte Suche'.

Ein wesentlicher Teilaspekt eines unternehmensübergreifenden Suchdienstes ist die Auswahl relevanter Datenquellen, beispielsweise vernetzte Desktops im Unternehmen. Der Mehrwert dieses Dienstes entsteht in der effizienten und geschickten Auswahl von Quellen; der Dienst soll möglichst wenig Quellen anfragen und trotzdem möglichst alle relevanten Quellen finden. Aufgrund der Unübersehbarkeit und Dynamik der Daten sowie der Volatilität und Autonomie der Quellen ist die Entwicklung dieses Dienstes eine besondere Herausforderung für die Informatik.

Die vorliegende Dissertation beschreibt einen solchen Dienst am Beispiel von Peer-to-Peer Netzwerken. Inspiriert durch Milgram's Untersuchungen der Small World Netzwerke entwickeln wir eine neue Routing Strategie für ein volatiles Netzwerk, in dem ein Peer eine Person repräsentiert. Aus den Interaktionen der Peers leiten wir zusätzliche Verbindungen im Netzwerk, sogenannte Shortcuts, ab, die jeder Peer lokal in einem Index speichert. Dadurch entsteht ein Overlay Netzwerk, welches eine für das effiziente Routing besonders hilfreiche Anordnung der Peers aufweist: Peers mit ähnlichen Interessen sind direkt miteinander vernetzt. Eine dynamische Kombination von themenspezifischen, vernetzungsabhängigen und zufälligen Routing Strategien entlang der Shortcuts ermöglicht die gezielte und effiziente Auswahl relevanter Quellen mit minimaler Belastung des Netzwerkes und ohne manuelle Unterstützung durch den Benutzer. Für die Verwaltung der lokalen Shortcut Indices entwickeln wir einen neue Indexstrategie. Diese erlaubt die gezielte Aktualisierung lokal gespeicherter Shortcuts und berücksichtigt sowohl Änderungen der Verfügbarkeit von Quellen als auch von Daten im Netzwerk.

Die Ergebnisse der vorliegenden Arbeit unterstützen maßgeblich die Entwicklung eines integrierten Suchdienstes. Simulationen zeigen, dass, gegenüber vergleichbaren Ansätzen, der Recall für eine Anfrage deutlich erhöht und die Kosten für eine Anfrage drastisch gesenkt werden. Shortcut Overlay Netzwerke sind robust, sie tolerieren wechselnde Interessen sowie eine hohe Volatilität der Peers. Diese Eigenschaften, kombiniert mit der vollständig lokalen Erstellung, Auswahl und Verwaltung der Indices, machen Shortcut Overlay Netzwerke zu einer sehr vielversprechenden Alternative zu Flooding-basierten Ansätzen oder verteilten Hashtabellen.

Abstract

In research and business currently we notify three key trends: the explosion of unstructured data; the critical need to formally manage content; and internetworking and collaboration within and between enterprises.

Peer-to-Peer information systems address the need to access content wherever it resides, to produce content while maintaining control over it, and to collaborate efficiently by sharing real-time data within a distributed network of stakeholders. Enterprises that are highly dependent on sharing real-time information across geographically spread knowledge workers are likely to benefit immediately from peer-to-peer information systems.

This thesis focuses on the issue of determining a relevant peer in a completely decentralized and volatile setting without any static peers, such as necessitated by peer-to-peer information systems in virtual organizations. Example applications, such as the networked semantic desktop and legal music sharing, serve as rationale throughout the thesis. We discuss, which routing strategies exist, when they should be used, and -most importantly- how can we enhance their recall and lower their communication costs. The full autonomy of peers as well as the full control of their own resources preclude prominent resource location and query routing schemes, such as distributed hash tables. We propose a new resource location and a semantic query routing approach that exploits social metaphors of topical experts and experts' experts as well as semantic similarity of queries and information sources. The novel design principle of our approach lies in the dynamic adaptation of the network topology, driven by the history of successful or semantically similar queries. This is memorized by using bounded local shortcut indexes storing semantically labelled shortcuts and a dynamic shortcut selection strategy, which forwards queries to a community of peers that are likely to best answer queries.

Our results support the development of a completely decentralized peer-to-peer information system significant. Extensive simulations show that the clustering of peers within semantic communities drastically improves the overall performance of our algorithm even in a highly volatile setting, while our index policy locally indices the 'right' peers, that provide resources to the core interests of a requesting peer. Shortcut overlays are robust; they tolerate interests shifts and high network volatility. These attractive properties, combined with the locality preserving design of the index management and peer selection algorithm, pose shortcut overlay networks as a very promising alternative to state of the art semantic routing approaches.

Danksagung

Viele Menschen in meinem beruflichen und privaten Umfeld haben mich beim Entstehen dieser Arbeit begleitet. Zuerst danke ich meinen beiden Betreuern für ihre Unterstützung und ihren Rat während der Anfertigung dieser Arbeit. Professor Weber stellte mir in der CIS Gruppe an der Technischen Universität Berlin die Arbeitsumgebung und die wissenschaftliche Infrastruktur zur Verfügung. Professor Nejdl vom Forschungszentrum L3S half mir bei der Ausrichtung es Themas. Zusammen begleiteten mich beide Betreuer in der Anfertigung der Arbeit und halfen mir durch viele wertvolle Kommentare.

Ich bedanke mich bei meinen Kollegen der Fachgruppe 'Computergestützte Informationssysteme' für die schöne Zeit und die vielen kontroversen Diskussionen. Mein Dank geht an Dr. Ralf-Detlef Kutsche für die Unterstützung meiner Forschung. Bei Lutz Friedel und Claudia Gantzer bedanke ich mich für die prompte Lösung technischer und administrativer Probleme.

Diese Arbeit hätte nicht ohne die finanzielle Begleitung durch das Berlin- Brandenburger Graduiertenkolleg für verteilte Informationssysteme (DFG Stipendium GRK 316/3, Sprecher Prof. Dr. Oliver Günther) geschrieben werden können. Prof. Dr. Felix Naumann von der Humboldt Universität danke ich besonders für seine stetig konstruktiven Kommentare. Durch die regelmäßigen Treffen des kleinen und großen GKs erhielt ich kontinuierlich wertvolle Anregungen und lehrreiche Kritiken.

Viele, wichtige und hilfreiche Diskussionen, Anregungen zum 'Um die Ecke Denken' und die notwendige Motivation erhielt ich von der Peer-to-Peer Community am Forschungszentrum L3S, insbesondere Dr. Wolf-Tilo Balke, Dr. Martin Wolpers, Wolf Siberski und Uwe Thaden und am AIFB Karlsruhe, insbesondere von Prof. Dr. Steffen Staab und Christoph Tempich. Ganz herzlich bedanke ich mich auch bei meinen Diplomanden Bastian Quilitz, Kai Schubert und Frederick Zimmer für deren Unterstützung bei der Implementierung der Simulationsumgebung. Ein wichtiger Begleiter seit meinem ersten Studien-Semester war und ist Prof. Dr. Harald Brandenburg von der FHTW Berlin, der mir auch in schwierigen Zeiten mit Rat und Hilfe zur Seite stand. Dr. Andy Seaborne und dem Europäischen Leonardo Programm danke ich für die finanzielle und inhaltliche Unterstützung während meines Aufenthaltes in der Semantic Web Gruppe in den Hewlett Packard Research Labs Bristol.

Schließlich gilt mein Dank meinen Freunden und meiner Familie für ihre stetige Motivation und Unterstützung.

Inhaltsverzeichnis

Abbildungsverzeichnis

Teil I

Informationssuche in Peer-to-Peer Netzwerken

1 Einführung

Das Internet wurde traditionell als eine Kommunikationsinfrastruktur betrachtet. Mit ihr ist es möglich, so der Anspruch, jede Information zu jedem Zeitpunkt an jeden Ort zu transportieren. Mit der Einführung des World Wide Web trat ein anderer Aspekt der Nutzung des Internets in den Vordergrund: Nicht mehr die gezielte Kommunikation zwischen Partnern, sondern die indirekte Kommunikation über Web-Pages steht im Mittelpunkt der Betrachtung. Deren Folge ist eine Überschwemmung mit einer Flut von Informationen. Das *Future Net* [Web04] - das Internet der kommunizierenden Gemeinschaften - wird neue, abstraktere Kommunikationstechnologien zur Verfügung stellen müssen, die einer zielgerichteten Kommunikation von Individuen und Gemeinschaften von Individuen mit ihren jeweiligen spezifischen Eigenschaften gerecht wird.

Parallel zur Entwicklung des World-Wide-Web hat sich das Peer-to-Peer (P2P) Paradigma zu einem der meistdiskutierten Phänomene in der jüngeren Geschichte der Informationstechnologie herausgebildet. Peer-to-Peer Netzwerke bilden die Infrastruktur für virtuelle Gemeinschaften, die Ressourcen teilen, den Informationsaustausch beschleunigen und neuartige kollaborative Arbeitsumgebungen ermöglichen. Den enormen Erfolg von Peer-to-Peer Technologie belegen die populären File Sharing Systeme, wie Napster, Gnutella und ihre Nachfolger. Durch die Installation einer kostenlosen Software und ohne Berücksichtigung besonderer administrativer oder finanzieller Vereinbarungen wird der Computer spontan Teil einer existierenden, völlig dezentralen, selbstorganisierenden Infrastruktur.

Napster und *Gnutella* zeigen den hohen Bedarf der Musik-Community, Audio Clips über das Internet mit anderen Musik-Fans auszutauschen. Das Spektrum potentieller Anwendungsbereiche für den unternehmerischen Einsatz von Peer-to-Peer Netzwerken ist jedoch deutlich breiter. Die Nutzung von Peer-to-Peer Netzwerken für die Publikation, die Suche und den Austausch von Office Dokumenten innerhalb der 'Community' eines Unternehmens wurde bereits für eine vergleichsweise kleine Anzahl von Nutzern in Systemen, wie *YOUSERV* [AGS02] mit mehreren hundert Nutzer oder in Groove [Net04], in mehreren kleinen Gruppen mit bis zu hundert Nutzern, realisiert. Durch eine Suche und den Austausch von Dokumenten über eine Peer-to-Peer Infrastruktur wird kollaborative Wissens-Koproduktion, in virtuellen, spontan miteinander verbundenen Communities, innerhalb und außerhalb institutioneller Barrieren erst ermöglicht bzw. deutlich vereinfacht. Neue Formen der lokalen Suche, wie durch die *Google Desktop Search*, werden eine Peer-to-Peer basierte Suche in den lokalen Desktops aller Mitarbeiter eines Unternehmens oder einer virtuellen Organisation unterstützen. Jede Art von Dokument wird schnell auffindbar sein - sei es eine Textdatei, eine E-Mail, eine Tabelle oder eine andere Art von Dokument. Ein weiteres Beispiel ist das System *Bibster* [HBM+04]. Es erlaubt das Management, die globale Suche, Publikation und Identifikation inkonsistenter bibliographischer Metadaten innerhalb der Research Community. Durch die Peer-to-Peer Philosophie trägt jeder Peer einen Teil der Kosten, z.B. Bandbreite oder Speicherplatz, für die Suche und Publikation der Metadaten. Im medizinischen Bereich wird die sichere und standardisierte Suche in Patientendaten durch die behandelnden Ärzte ohne eine zentrale Instanz über ein Peer-to-Peer System realisiert. Beispielsweise erlaubt das System *Care Data Exchange* [Exc04] die Suche in von anderen Ärzten erfassten Patientendaten, wie bereits festgestellte allergische Reaktionen auf verordnete Medikamente, durchgeführte EKGs vor einem Herzanfall oder den Schwangerschaftsverlauf einer Mutter. Es erlaubt die Korrektur inkonsistenter Eingaben durch den Arzt und ermöglicht die Einsparung von Kosten für eine zentrale Infrastruktur. Darüber hinaus ist geplant, auch Patienten Einsicht in Ihre eigenen Daten zu geben. Zum Schutz der Daten müssen sich die behandelnden Ärzte über einen zentralen Index anmelden. Der Index erhält Verweise für jeden Patienten auf die in bisher behandelten Ärzte. Die eigentlichen Daten der Patienten werden lokal bei den einzelnen Ärzten gespeichert.

Die Einfachheit und „kostenlose“Nutzung vorhandener Peer-to-Peer Infrastrukturen, ohne eine Überwachung und Kontrolle durch eine zentrale Steuerungsinstanz, hat in den letzten Jahren zu einem Anwachsen der Nutzeranzahl geführt und eine enorme Nachfrage nach Verfahren und Technologien stimuliert, wie auch grosse Peer-to-Peer Netzwerke effizienter und noch kostengünstiger realisiert werden können. Durch die starke Verteilung der Inhalte über eine grosse Anzahl von Peers und deren Autonomie und Dynamik entstehen Herausforderungen bei der

Realisierung von Peer-to-Peer-basierten Information Sharing Netzwerken. Beispiele sind Verfahren für die gerechte Verteilung der Kosten über die Peers, wie Bandbreite und Speicherplatz, die Überbrückung semantischer Heterogenitäten bei der Annotation oder die Reduzierung von Netzwerkkosten beim häufig auftretenden Eintreten und Verlassen einzelner Peers in das Netzwerk (Churn). Die vorliegende Arbeit widmet sich der wichtigsten fundamentalen Fragestellung ohne deren Beantwortung die Verteilung der Inhalte obsolet wird: Was ist eine effiziente Strategie für die Suche und das Auffinden von Dokumenten in einem solchen Netzwerk?

1.1 Peer-to-Peer basierte Suche von Informationen

Peer-to-Peer Netzwerke weisen im Gegensatz zu zentralen oder Client/Server basierten Architekturen folgende interessante Besonderheiten auf:

- **Verfügbarkeit und Dynamik der Knoten.** Begründet durch die Autonomie der einzelnen Knoten und deren schiere Menge existiert eine hohe Dynamik der verfügbaren Peers im Netz. So beträgt in File Sharing Netzwerken die durchschnittliche Online Zeit eines Peers ungefähr eine Stunde [Mar02, CLL04]. Das bedeutetet einerseits, dass nach einem Ausfall oder Verlassen des Peers sämtliche von diesem Peer publizierte Inhalte nicht mehr verfügbar sind und das andererseits die Last für die Wartung und Selbstorganisation des Netzwerkes auf andere Knoten übertragen werden muss. Diese Volatilität erfordert neue Technologien für die zeitnahe Indizierung und das verteilte Management der Inhalte.

- **Selbstorganisation ohne Koordinierung durch eine zentrale Instanz.** Eine verteilte Speicherung und Suche der Inhalte erfolgt häufig aus dem Wunsch heraus, keine zentrale Instanz im System zu haben. Dieser Wunsch resultiert eher aus ideologischen Gründen, wie dem Wunsch der lokalen Kontrolle über die Inhalte, der Vermeidung eines zentralen Zensors und der gerechten Aufteilung der laufenden Kosten des Systems. Sekundär erfolgt inherent eine hohe Verteilung der Inhalte aus der technischen Notwendigkeit der Vermeidung von Single-Point-of-Failures oder aus Gründen der Skalierbarkeit und Robustheit [RBR+04, MKL+02].

- **Verschmelzung der Anbieter- und Nachfragerrolle.** Im Gegensatz zu der häufig strengen Trennung von Informationsnachfrager (Information Consumer) und Informationsanbieter (Information Provider) in zentralistischen oder Client/Server basierten Informationssystemen verschmelzen beide Rollen in einem Peer-to-Peer System. Ein Peer in einem Peer-to-Peer Netzwerk,

der häufig einen konkreten menschlichen Nutzer repräsentiert, stellt ebenso Anfragen nach Inhalten an andere Peers, wie auch von dem eigenen Peer eingehende Anfragen beantwortet werden [NWQ+02, MKL+02].

- **Technische Heterogenität der Knoten.** Aufgrund der Autonomie der einzelnen Knoten sind ihre technischen Zugriffseigenschaften, wie Bandbreite und Zugriffszeit, heterogen [YGM03].

Peer-to-Peer Netzwerke für die Suche von Dokumenten weisen darüber hinaus folgende Besonderheiten auf:

- **Unterschiedliche Popularität der Dokumente und Anfragen.** Die Dokumente in Peer-to-Peer Information Sharing Systemen sind stark im Netzwerk verteilt. Typische Grössenordnungen aktueller File-Sharing Netzwerke betragen mehrere hunderttausend Peers von denen jeder Peer wenige bis zu mehreren tausend Dokumenten anbietet. Aufgrund der vollständig dezentralen Speicherung sind populäre Inhalte redundant vorhanden und verbreiten sich bei entsprechender Nachfrage auch mit hoher Geschwindigkeit. In Musik Sharing Netzwerken nähert sich die Verteilung der Inhalte und die Verteilung der Anfragen über die Peers einer ZIPF [Zip49] Verteilung an.

- **Verwendung von Metadaten.** Metadaten, Daten über Daten, werden in Peer-to-Peer Netzwerken zur Beschreibung von zwei Klassen von Entitäten verwendet. In aktuellen File Sharing Netzwerken werden Dokumente durch einen homogenen Metadatensatz beschrieben. Beispielsweise sind in E- Learning Netzwerken die Standards *Dublin Core* und *LOM* zur Annotation von Dokumenten weit verbreitet. Ebenfalls werden in unstrukturierten Peer-to-Peer Netzwerken *Peers* durch (implizit gewonnene) Metadaten beschrieben. Diese Modelle ermöglichen den Einsatz von Ranking Verfahren für Peers, die zur Verringerung der Nachrichten für eine Suche führen können. In dieser Arbeit werden beide Formen von Metadaten in Peer-to-Peer Netzwerken untersucht.

- **Strategien für eine verteilte Suche.** Aufgrund des häufig fehlenden zentralen Index haben sich zwei Strategien für die Suche etabliert: Strategien mit einem globalen (verteilten) Index und Strategien mit einem lokalen Index. Beispiele für die Suche auf der Basis eines globalen (verteilten) Index sind Verteilte Hashtabellen (Distributed Hash Tables) und Routing Indizes. Eine Suche auf der Basis lokal vorhandener Informationen erfolgt auch durch das Überfluten des Netzwerkes oder durch das Prinzip der Shortcuts. Ein Ziel der aktuellen Forschung ist die Minimierung der Nachrichten für eine Suchan-

frage und für die Pflege des verwendeten Index. In dieser Arbeit stellen wir Suchstrategien vor, die sowohl globale und lokale Indizes berücksichtigen.

Die Existenz dieser besonderen Eigenschaften von Peer-to-Peer Systemen stellt neue Möglichkeiten für die Gewinnung von Daten über das Verhalten einzelner Peers bereit und erlaubt völlig neue Ansätze für eine ganzheitlichen Optimierung der Suche in derartigen Informationssystemen.

1.2 Adaptive Overlay Cluster

Eine der wichtigsten Funktionen in einem Peer-to-Peer Netzwerk ist die Suche nach Inhalten für eine gegebene Anfrage. Die globale Verfügbarkeit einer grossen Anzahl von Anbietern von Inhalten und deren hohe Dynamik in einem Peer-to-Peer Netzwerk erfordert neue Methoden für eine effiziente Suche. Eine Anfrage kann nicht mehr an alle zur Anfragezeit verfügbaren Quellen innerhalb des Netzwerkes geleitet werden, da die Kosten, wie die Anzahl der benötigten Nachrichten, dafür zu gross würden. Optimal wäre ein Ansatz bei dem das physische Netz aller zur Anfragezeit verfügbaren Quellen auf genau die Peers eingeschränkt werden, die für eine Anfrage auch die richtigen Inhalte bereitstellen und die Anfrage beantworten können. Wir bezeichnen eine solche Struktur als ein **Overlay** Netzwerk. In dieser Arbeit untersuchen wir Overlay Netzwerke, die Peers auf der Basis gemeinsamer oder ähnlicher Merkmale miteinander verbinden. Dadurch entstehen **Cluster** von Peers mit Dokumenten zu einem gemeinsamen Thema. Durch die Autonomie des einzelnen Peers sind diese nicht immer verfügbar bzw. verändern ihre verfügbaren Dokumente. Für einen Overlay Cluster müssen **adaptiv** neu verfügbare Dokumente und Peers selbstständig registriert, bzw. nicht mehr verfügbare Dokumente und Peers gelöscht werden. Eine Anfrage in einem solchem Netzwerk ist ein zweigeteilter Prozess:

1. Zur Anfragezeit werden Peers gesucht, die passende Inhalte für eine Anfrage speichern. Die Menge dieser Peers bildet einen adaptiven Overlay Cluster für diese Anfrage.

2. Die Anfrage wird an alle (oder an ausgesuchte Peers) aus dem adaptiven Overlay weitergeleitet. Jeder Peer, der die Anfrage erhält, sucht lokal nach für die Anfrage passenden Dokumenten.

Durch eine solche anfragespezifische Auswahl von Peers profitieren zwei Klassen von Akteuren:

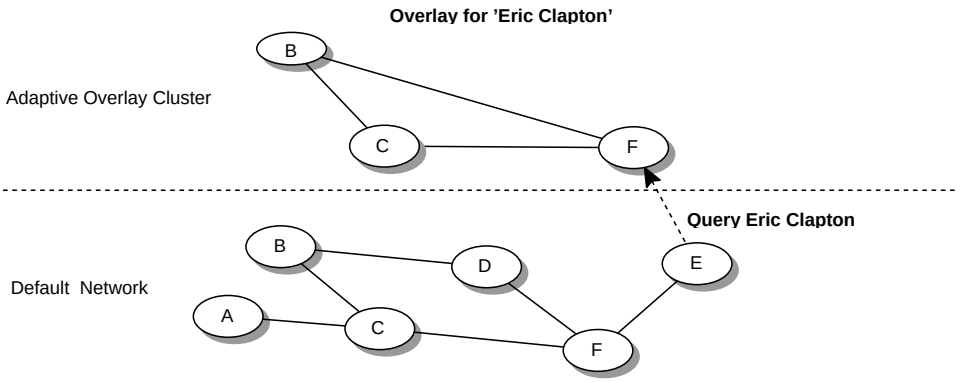


Abbildung 1.1: Adaptive Overlay Cluster

- Da die Anfrage direkt an die „passenden “Peers gesendet wird, können für den anfragenden Peer die Zeit bis zur ersten Antwort auf die Anfrage und der Aufwand zur Beantwortung der Anfrage signifikant gesenkt werden.
- Im Vergleich zu einer naiven Strategie erhalten Peers, die keine für die Anfrage passenden Inhalte publizieren, auch keine überflüssigen Nachrichten. Vielmehr werden die benötigten Ressourcen, wie Anzahl der Nachrichten im Netzwerk und Rechenkapazität lokaler Peers, auf eine kleine Anzahl ausgewählter Peers beschränkt.

Grafik 1.1 zeigt ein physisches Netzwerk mit sechs Peers $P_A, ..., P_F$. Die Knoten bilden ein Peer-to-Peer Netzwerk. Jeder Peer publiziert Dokumente, beispielsweise Audio Clips, und stellt Anfragen. In der Grafik stellt Peer P_E eine Anfrage Q_1 zu Audioclips von „Eric Clapton“. Die Peers B, C und F speichern Audio Clips zu Q_1. Sie sind untereinander direkt verbunden und bilden einen adaptiven Overlay Cluster für die gestellte Anfrage. Die Anfrage wird an einen P_E bekannten Peer geleitet, der Mitglied des Overlays für die Anfrage ist. Innerhalb des Overlays sucht jeder Peer lokal Audio Clips zu Q_1 und sendet die gefundenen Resultate an P_E zurück.

Die Realisierung einer Zuordnung von Peers zur adaptiven Overlay Cluster sollte weitestgehend automatisiert erfolgen, da eine manuelle Auswahl der Peers aufgrund der hohen Anzahl der Peers und ihrer hohen Dynamik nicht mehr manuell durchführbar ist. In dieser Arbeit untersuchen wir dazu zwei Ansätze. Im ersten Ansatz registrieren Peers in einem globalen, verteilten Index, welche Anfragen zu welchen Dokumenten sie unterstützen. Im zweiten Ansatz beobachtetet ein Peer aufgrund bereits gestellter Anfragen des eigenen oder anderer Peers, welche Peers

im Netzwerk Antworten liefern konnten und verwaltet diese Informationen in einem lokalen Index. Auf dessen Basis wählt er für eine Anfrage die besten Peers aus.

1.3 Problemstellung und Vorgehensweise

Diese Arbeit behandelt das Problem der effizienten Suche von Inhalten in Peer-to-Peer Netzwerken. Für ein gegebenes Netzwerk, bestehend aus einer Menge von dynamischen Peers die Inhalte anbieten und einem Peer der bestimmte Inhalte sucht, entwickeln wir Methoden (I) für das aktive und passive Gewinnen von Metadaten anderer Peers im Netzwerk, (II) der Auswahl der besten Peers für eine Anfrage ausschließlich auf der Basis lokalen Wissens und (III) für die Kombination verschiedener Routingstrategien. Wir teilen das Problem in folgende Teilprobleme auf:

- **Definition globaler Indexstrukturen.** Wir untersuchen zwei Ansätze, bei denen Peers Metadaten über ihre verfügbaren Dokumente in einem dezentralen Index registrieren. Dazu stellen wir zwei Index Strukturen in einem Super-Peer Netzwerk vor, bei der eine kleine Anzahl stabiler Peers den Index verwaltet und eine Suche in den Profilen dynamisch im Netzwerk vorhandener Peers ermöglicht.

- **Definition sozialer Metaphern.** Das Routing mit der Technologie der adaptiven Overlay Clustern erfolgt entlang Shortcuts zu einer themenspezifischen Menge von Peers. Wir definieren den Begriff des Shortcuts als unidirektionale Relation zwischen zwei Peers. Ausgehend von der Kommunikation in sozialen Netzwerken leiten wir Metaphern für die Bildung von Shortcuts her und definieren fünf Typen von Shortcuts.

- **Einführung von lokalen Indexstrukturen und einem Architekturmodell.** Shortcut Netzwerke repräsentieren eine adaptive und dynamische Netzwerkschicht über der physischen Netzwerkschicht. Zur lokalen und dezentralen Verwaltung dieses Netzwerkes definieren wir die Index-Struktur für die Speicherung von Shortcuts als Beziehung zwischen Anfragen eines Peers aus der Vergangenheit und bereits gefundenen Inhalten anderer Peers. Ausgehend von den Basiskomponenten eines Peers in einem Informationsharing Netzwerk stellen wir die Architektur eines Peer in einem Shortcut Netzwerk vor.

- **Gewinnung von Metadaten für die Bewertung von Peers.** Durch die hohe Verteilung der Inhalte, die Volatilität der Peers und die fehlende zentrale

Instanz sind Metadaten zur Generierung von Shortcuts und zur Auswahl von Peers für eine Anfrage besonders schwierig zu ermitteln. Zusätzlich sollen sie ohne explizite Interaktionen mit dem Nutzer erhoben werden können. Wir stellen einen Algorithmus vor, der Metadaten implizit bei der Ausführung von Anfragen sammelt. Neu mit dem Netzwerk verbundene Peers erlernen schnell neue Shortcuts zu anderen Peers.

- **Bewertung und Auswahl der besten Peers.** Für verschiedene Typen von Shortcuts definieren wir Metriken zur Auswahl der besten Peers für eine Anfrage. In Peer-to-Peer Netzwerken wird die Berechnung dieses Masses durch das fehlende globale Wissen über die Inhalte der verteilten Peers erschwert. Auf der Basis von Strategien für die Suche in sozialen Netzwerken definieren wir Algorithmen zur Auswahl der besten Peers für eine Anfrage. Abhängig von dem lokal vorhandenen Wissen über das Netzwerken wird dynamisch die optimale Strategie zur Weiterleitung der Anfrage bestimmt.

- **Entwicklung von Algorithmen für die Pflege der lokalen Indizes.** Jeder Peer pflegt lokal seinen Index mit den gesammelten Daten über Anfragen und Inhalte anderer Peers. Aufgrund der Dynamik der Peers und ihrer gespeicherten Inhalte veralteten diese Daten jedoch schnell. Wir untersuchen verschiedene Strategien und formulieren Algorithmen für die Aktualisierung der lokalen Indizes.

- **Simulation mit realen Daten.** Zur empirischen Überprüfung und Quantifizierung der Verfahren simulieren wir alle Ansätze anhand reeller Daten, die wir aus dem des Open Directory [DMO] ableiten. Wir vergleichen unseren Ansatz mit dem Gnutella Ansatz und weiteren aktuellen Routing Ansätzen für Shortcut Netzwerke im Hinblick auf die Kosten, der Genauigkeit des Routings und den Recall für eine Anfrage.

Die methodische Vorgehensweise umfasst zwei grundlegende Teilaspekte, die theoretische Fundierung der Arbeit und die praktische Untersuchung der Leistungsfähigkeit der gefundenen Algorithmen und Metriken. Wir definieren Begriffe und Terminologien für das Routing von Anfragen in Peer-to-Peer Netzwerken und untersuchen und analysieren Schwächen existierender Verfahren. Dazu wird auf aktuelle, zum Teil eigene, Veröffentlichungen aus dem Bereich des Peer-to-Peer Information Retrieval und der semantischen Suche in Peer-to-Peer Netzwerken zurückgegriffen. Der praktische Teil der Arbeit umfasst die empirische Untersuchung Effizienz der Algorithmen durch eine Simulation und auf der Basis reeller Daten. Zuerst sollen die gefundenen Metriken auf die Daten angewendet und fehlende Metadaten berechnet werden. In einem weiteren Schritt werden die in der Arbeit entwickelten

Algorithmen mit gängigen Algorithmen für die Lokalisierung von Inhalten verglichen.

1.4 Gliederung

Die Arbeit gliedert sich in drei Teile. Für jeden Teil fassen wir kurz den Inhalt zusammen und verweisen auf eigene Publikationen, in denen diese Ideen und Resultate präsentiert wurden. Zum Schluss jedes Kapitels diskutieren wir unseren Ansatz mit verwandten Arbeiten:

Teil I - Informationssuche in Peer-to-Peer Netzwerken
Kapitel 1: Einführung
Dieses Kapitel motiviert und führt in das Problem der Lokalisierung von Inhalten in Peer-to-Peer Netzwerken ein. Zur Reduzierung der Kosten für die Lokalisierung soll eine Anfrage möglichst nur an Peers gesandt werden, die passende Inhalte publizieren. Wir teilen das Problem in mehrere Teilproblemstellungen auf und stellen die Gliederung der Arbeit vor.

Kapitel 2: Anwendungen und Architekturmodelle
Wir analysieren typische Charakteristiken potentieller Anwendungen der Informationssuche in Peer-to-Peer Netzwerken und stellen die Anwendungsgebiete Legale Musiktauschbörse und Group ware kurz vor. Später klassifizieren wir in der Praxis und in der Forschung auftretende Architekturmodelle anhand des Grades der Zentralisierung und der Struktur des Overlays.

Kapitel 3: Routing auf der Basis globaler Indizes
Wir stellen zwei neue Routing Strategien vor, die mit unterschiedlichen Technologien einen verteilten globalen Index implementieren. Für das Themen-Hierarchie basierte Routing auf der Basis von verteilten Hashtabellen stellen wir das System *TOPICS* [Lös04b, LSZ04] vor. Das System *Edutella* zeigt das Routing in einem schema-basierten Netzwerken auf der Basis von Routing Indizes [LNWS03, NWS+03].

Teil II- Adaptive Overlays auf der Basis lokaler Shortcut Indizes
Kapitel 4: Der INGA Ansatz
Wir beschreiben detailliert das Konzept adaptiver Overlays auf der Basis lokaler Indizes als logische Schicht über dem physischen Netzwerk, definieren soziale Metaphern für das Routing und die Kernkomponenten der Architektur, wie lokale Indizes und Shortcuts [Lös04a, LWSN03, LNS+03].

Kapitel 5: Erzeugung von Shortcuts
Auf der Basis sozialer Metaphern entwickeln wir ein neues Overlay-Modell. Zur Erstellung des Overlays entwickeln wir Verfahren zur Gewinnung von Metadaten über einen Peer. Dazu beobachten wir Anfragen und deren Resultate [LT05].

Kapitel 6: Index Verwaltung und Routing Modell
Auf der Basis der Bewertung eines Peers für eine Anfrage entwickeln wir einen neuen Algorithmus für das Weiterleiten von Anfragen. Dazu untersuchen wir Heuristiken und leiten aus diesen Algorithmen ab. Wir untersuchen Algorithmen, die eine exakte Übereinstimmung zwischen einer Anfrage und einem Index Eintrag erfordern, Algorithmen die Ähnlichkeiten berücksichtigen und stellen Funktionen zu Berechnung der Ähnlichkeit vor [LST05, LTQ+05].

Kapitel 7: Simulation und Evaluierung
Zur empirischen Bewertung der Algorithmen werden mit reellen Daten verschiedene Szenarien simuliert. Die Ergebnisse der Simulationen werden anhand der Anzahl der Ergebnisse für eine Suche und auf den Verbrauch von Nachrichten miteinander verglichen. Ziel ist das Finden einer optimalen Kombination aus Aufwand und Nutzen für eine Suchanfrage.

Teil III - Diskussion
Kapitel 8: Zusammenfassende Diskussion
Dieses Kapitel diskutiert die wichtigsten Ergebnisse der Arbeit. Es schliesst mit der Betrachtung weiterer Forschungsrichtungen ab.

2
Anwendungen und Architekturmodelle

In diesem Kapitel geben wir eine Übersicht typischer Charakteristiken von Peer-to-Peer Systemen und stellen potentielle Anwendungen der Informationssuche vor. Im zweiten Teil diskutieren wir in der Praxis und Forschung auftretende Architekturmodelle, die wir klassifizieren anhand des Grades der Zentralisierung und der Struktur des Overlays.

2.1 Peer-to-Peer Netzwerke

Eine neue Welle von Netzwerk Architekturen bilden die Basis für neue Dienste in verteilten Informationssystemen. Beispiele sind die Internet Telephonie mit *Skype*, der Austausch von Medien über *Gnutella*, die elektronische Zusammenarbeit in Gruppen mit *Groove* oder die verteilte Analyse von interstellaren Signalen mit *Seti@Home*. Die Autoren in [Ora01, ATS04, MKL+02] verstehen unter einer solchen *Peer-to-Peer* Architektur ein 'sich selbst organisierendes System gleichberechtigter, autonomer Einheiten (Peers)', das 'vorzugsweise ohne Nutzung zentraler Dienste auf der Basis eines Rechnernetzes mit dem Ziel der gegenseitigen Nutzung von Ressourcen (CPU Zyklen, Speicherplatz oder Informationen) operiert'. Der Begriff Peer-to-Peer wird gelegentlich mit folgenden anderen Begriffen verwechselt:

- *Client-Server* repräsentiert die Ausführung von Rechnern in einem Netzwerk mit den Rollen der Clients und der Server. Typischerweise verteilen sich beide Rollen auf unterschiedliche Rechner im Netzwerk, wobei meist ein

Rechner als Server und die restlichen Rechner als Clients fungieren. Jeder Rechner kann beide Rollen, jedoch für unterschiedliche Dienste, gleichzeitig wahrnehmen. Beispielsweise agiert ein Rechner als Mail-Server und bezieht die dafür benötigte Software als FTP Client beziehen.

- *Distributed Computing* wird definiert als ein Computersystem, in dem mehrere miteinander verbundene Computer dessen Aufgaben unter einander aufteilen. Derartige Systeme beinhalten Cluster von Computern, die Ressourcen über das Internet von vernetzten Computern beziehen. Wir verwenden den Begriff auch für Systeme, die Peer-to-Peer Eigenschaften aufweisen.

- *Grid Computing* wird von [IF99] definiert als die koordinierte, gegenseitige Nutzung freier Ressourcen und das gemeinsame Lösen von Problem in virtuellen Organisationen. Ein Grid ist eine Infrastruktur die Berechnungen optimal auf ein Netzwerk von Computern aufteilt. Häufig wird die Vernetzung über das Client-Server Modell realisiert, beispielsweise beim Seti@Home Projekt.

- *Ad-hoc Communication* wird definiert als ein System, dass eine Kommunikation zwischen Computern ermöglicht, ohne das eine Infrastruktur davor bereits zwischen diesen existierte. Das Netzwerk und die beteiligten Computer kontrollieren ihre Kommunikation, Sicherheit und Identität eigenständig. Peer-to-Peer Systeme können auf der Basis existierender Ad-Hoc Infrastrukturen realisiert werden.

Letztendlich lässt sich jedoch nur schwer festlegen, welches System ein Peer-to-Peer System ist und welches nicht. Wir glauben, dass dieser Mangel an einer einheitlichen Definition vielmehr durch die Verwendung des Begriffs aufgrund einer subjektiv wahrgenommenen Peer-to-Peer Interaktion in einer Anwendung, als durch eine konkrete technologische Umsetzung entsteht.

In der Literatur [MKL+02, ATS04] werden Peer-to-Peer-Systeme durch folgende, technisch orientierte Charakteristika beschrieben, wobei oftmals nur im Idealfall alle Charakteristika gleichzeitig zutreffen:

- **Selbst-Organisation.** Peer-to-Peer Netze organisieren sich selbstständig in einem Netzwerk und ermitteln ihre Nachbarn in einem verteilten Prozess.

- **Symmetrische Kommunikation.** Peers agieren als Server, in dem sie Anfragen beantworten und Dienste zur Verfügung stellen, und als Client, in dem sie Anfragen stellen und Dienste nutzen.

- **Dezentrale Kontrolle.** Jeder Peer definiert seine Rolle und den Grad seiner aktiven und passiven Partizipation im Netzwerk autonom, ohne eine zentrale, kontrollierende Instanz.

- **Eindeutige Identifikation eines Peers.** Jeder Peer kann über einen eindeutigen Schlüssel identifiziert werden. Im Peer-to-Peer Netzwerk Gnutella wird eine GUID für jeden Peer vergeben, wenn er sich mit dem Netzwerk verbindet. Der Ansatz verfügt über den Nachteil, das der Schlüssel sich von Session zu Session ändern kann. Zur Lösung dieses Problems verfügt das JXTA Netzwerk über das Konzept des *Virtual Addressing* [TAA+03]. Für jeden Peer wird eine von dessen IP Adresse unabhängige logische PeerID über eine 128-Bit Hashfunktion berechnet. Zu Begin jeder Session identifiziert sich der Nutzer mit einem Passwort und einer UserID. Bei Erfolg wird ihm vom System seine eindeutige logische PeerID zugewiesen. Die Abbildung der logischen PeerID in die physische Adresse erfolgt im JXTA Rendevouz Netzwerk. Der Ansatz hat den Vorteil, dass ein Laptop, der über ein DHCP Netzwerk angemeldet wird, immer derselben logischen PeerID zugewiesen wird. Ebenfalls kann einer Person nur eine PeerID zugewiesen werden, die vom eingesetzten Gerät (Laptop, PDA, Desktop PC) unabhängig ist.

Die Autoren in [RBR+04] identifizieren folgende Charakteristiken, die einen Einsatz von Peer-to-Peer Technologien aus ökonomischer Sicht rechtfertigen:

1. **Verfügbares Budget.** Ist das Budget für eine zentrale Lösung ausreichend wird eine Peer-to-Peer basierte Lösung aufgrund ihrer Komplexität, der Probleme beim Testen und ihrer Ineffizienz, vergleichbar mit einer zentralen Lösung, nicht in Betracht kommen. Ist das Budget jedoch eng begrenzt, motivieren die geringen Kosten für eine Peer-to-Peer basierte Lösung. Häufig stellt sogar der kontinuierliche Aufbau eines Systems aus einzelnen Peers die einzig ökonomisch akzeptable Vorgehensweise dar.

2. **Relevanz der Ressourcen und Dienste.** Die Relevanz beschreibt den Bedarf eines Nutzers an einer nur im Peer-to-Peer System verfügbaren Ressource (einem Dokument) oder einem Dienst (einer verteilten Suchmaschine in einem Unternehmen). Eine geringe Relevanz korreliert mit einer geringen Kooperation innerhalb des Peer-to-Peer Systems und erfordert zusätzliche Anreize.

3. **Vertrauen der Nutzer.** Die Identifizierung und Isolierung 'bösartiger' Peers erfordert zusätzliche Kosten. Derartige Peers stellen falsche, unvollständige oder nutzlose Dokumente zur Verfügung oder agieren ausschliesslich als Konsument von Dokumenten.

4. **Änderungsrate des Systems.** Aktualität, Qualität und Vollständigkeit einer Anfrage variieren durch die hohe Dynamik der vorhandenen Dokumente und die Volatilität der einzelnen Peers.

5. **Bedeutsamkeit der Anwendung.** Ist eine Anwendung von zentraler Bedeutung für ein Unternehmen, werden Nutzer Systeme mit einer zentralen Kontrolle, hoher Fehlertoleranz und hoher Ausfallsicherheit bevorzugen.

2.2 Anwendungen für die Informationssuche

Wir stellen zwei Anwendungen für die Informationssuche in Peer-to-Peer Netzwerken aus der aktuellen akademischen Forschung (Semantisch vernetzte Arbeitsplätze) und aus der Praxis (Legale Musiktauschbörse) vor. Anschließend bewerten wir diese anhand der im letzten Abschnitt definierten Kriterien.

2.2.1 Legale Musiktauschbörse

In der Vergangenheit wurden File-Sharing Anwendungen immer wieder mit dem illegalen Tausch von Medien in Verbindung gebracht. Durch den offenen DRM-Standard der *Open Mobile Alliance* (OMA) [OMA04] wurden die Grundlagen für einen effektiven, hersteller- und betreiberunabhängigen Schutz digitaler Rechte geschaffen. Zukünftige legale Musiktauschbörsen, wie der Napster Nachfolger *SnoCap* [tPW05], nutzen die Vorteile der Peer-to-Peer Tauschbörsen für den Vertrieb von Musik und integrieren die Rechteinhaber in das System. Sie entkoppeln die Verwaltung der Rechte von den im Peer-to-Peer Netzwerk getauschten Medien. Jedes Dokument verfügt über eine DRM Kennung, die dessen Rechte für einen Nutzer spezifiert. Wird eine Datei von einem Nutzer kopiert bzw. abgespielt, so muss die Kopie durch den Erwerb von Rechten freigeschaltet werden. Während die Verwaltung der Rechte separat in einem zentralen Dienst erfolgt, werden die Dokumente im einem Peer-to-Peer Netzwerk publiziert, gesucht, zusammengestellt und verteilt. Die Dokumente sind über Metadaten, beispielsweise Künstler, Title, Genre, des *ID3V2 Standards* annotiert. Zur Beschreibung der Genre und Stimmung der Musik existieren Themen-Hierarchien für Musik, wie *allmusic.com*, *musicmoz.org*. Die Metadaten werden durch zentrale, frei zugängliche ID3V2 Datenbanken, wie *Freedb.org*, verwaltet und manuell durch die Nutzer annotiert.

Für den Inhaber der Rechte an der Musik stellt dieses Modell eine kostengünstige und selbstorganiserende Infrastruktur für den Vertrieb bereit. Die Kosten für Vertrieb, Speicherung und die Zusammenstellung der Musik werden durch die Nutzer des Peer-to-Peer Systems getragen. Die Verfügbarkeit der Dokumente korreliert mit der Popularität, populäre Dokumente sind hoch verfügbar. In File-Sharing

Anwendungen ist die Änderungsrate des Systems hoch, da dessen Nutzer oftmals häufig nur für kurze Zeit online sind und Dokumente häufig hinzugefügt werden. Peers vertrauen typischerweise einander, das die bereitgestellten Dokumente auch vollständig sind. Zur Sicherung der Qualität und des Vertrauens können Nutzer in diesen Systemen in Kontakt treten, einander Musiktitel empfehlen und diese legal austauschen. Für die Weitervermittlung von Musik erhalten Nutzer Anreize, wie kostenlose Downloads. Der Austausch von Musik stellt keine kritische Anwendung für den einzelnen Nutzer dar.

2.2.2 Semantisch vernetzte Arbeitsplätze

Mitarbeiter verbringen viel Zeit damit, nach Informationen zu suchen. Zeitaufwand und Kosten für die Informationssuche und -weitergabe sind hoch, oft müssen Mitarbeiter um Mithilfe gebeten werden. Das notwendige Wissen ist implizit über die Expertise einzelner Mitarbeiter und explizit in deren lokal verfügbaren Dokumenten gespeichert [NT95]. Ein Zugriff auf die lokal gespeicherten Dokumente der einzelnen Mitarbeiter würde den Recall der Dokumentensuche in einem Unternehmen vergrössern und gleichzeitig potentielle Experten für eine Anfrage identifizieren.

Verschiedene Forschungsprojekte [DF04, TEF+04, HBM+04, ACMH02] untersuchen Peer-to-Peer Technologien für eine semantische Suche nach Dokumenten im Unternehmen und in virtuellen Organisationen mit mehreren tausend Peers. *Microsoft* [Coo05] setzt die Technologie für die Segmente Home Office und Small Enterprise als Group Ware Infrastruktur für bis zu zehn Peers ein, für die der Kauf und Betrieb eines Servers zu teuer wird. Sie nutzen den Zusammenhang zwischen der Generierung, Verwaltung, Publikation und semantischen Einordnung von Dokumenten auf dem lokalem Desktop durch den Nutzer. Über eine Vernetzung der einzelnen Arbeitsplätze entsteht eine kostengünstige, selbstorganisierende Infrastruktur ohne zentrale Koordination zum Austausch von Dokumenten und zur Identifikation von Experten. Mit der Nutzung des Peer-to-Peer-Verfahrens wird die Suche dezentral und ohne Verwaltungsaufwand betrieben. Dadurch sind die Daten aktuell und es Kosten werden eingespart. Eine Relevanz der Anwendung hängt von der Bereitschaft der Mitarbeiter zur Publikation von Dokumenten, der Verfügbarkeit möglicher Alternativen (wie einer vorhandenen zentralen Ablage oder Suche im Intranet) und der Unternehmenskultur ab. Im akademischen Umfeld und in Non-Government Organisationen erwarten wir eine Offenlegung der Zuordnung eines Nutzers zu einem Peer. Durch die lokale Erzeugung von Dokumenten und die Mobilität der Endgeräte erwarten wir eine hohe Änderungsrate. Aufgrund des frühen Entwicklungsstadiums der Anwendung lassen sich keine Aussagen über ihre Bedeutsamkeit treffen.

2.3 Architekturvarianten zur Informationssuche

Die Autoren in [ATS04, MKL+02] haben Peer-to-Peer Systeme für die Lokalisierung von Dokumenten nach dem Grad der Zentralisierung und der Struktur des Overlay Netzwerkes klassifiziert. Wir übernehmen diese Struktur und stellen Prinzipien und Beispiele für die jeweilige Architektur vor.

2.3.1 Grad der Zentralisierung

Obwohl Peer-to-Peer Systeme in ihrer einfachsten Form, dem vollständig dezentralen Ansatz, keine zentralen Komponenten aufweisen, werden in der Praxis häufig Systeme mit einem unterschiedlichen Anteil von Zentralisierung verwendet. Wir unterscheiden die folgenden drei Klassen von Zentralisierung:

Vollständig dezentralisiert. In diesen Systemen haben alle Knoten die gleichen Aufgaben und agieren sowohl als Client als auch als Server. Es existiert keine zentrale Instanz im Netzwerk. Nachrichten und Dokumente werden direkt zwischen Peers ausgetauscht. Aufgrund des Fehlens eines *Single Points of Failures*, der hohen Dynamik und Unübersichtlichkeit der beteiligten Peers und der schlechten Kontrollmöglichkeit sind diese Systeme hinsichtlich technologischer Angriffe (Zerstörung von Peers, Überflutung von Peers mit Anfragen) und 'politischer Angriffe' (Austausch illegal erworbener Dokumente, Verbieten des Netzwerkes) robust. Beispiele für dieses Architekturmodell sind *Gnutella* [DZW04] bis zur Version V0.4, *Freehaven* [DFA00], *CAN* [RFH+01], *CHORD* [SMK+01], Pastry [RD01], *P-Grid* [Abe01] und *Tapestry* [ZKJ01].

Super-Peer. Die Autoren in [CLL04, SGG03] argumentieren, dass nicht alle Peers in einem Peer-to-Peer Netzwerk über gleiche technische Leistungsfähigkeit verfügen. Super-Peer Netzwerke sind zwischen vollständig zentraliserten Netzwerken und vollständig dezentralen Netzwerken einzuordnen. Sie führen Hierarchie, in Form von Super-Peer Knoten, in das Netzwerk ein. Super-Peers verfügen über eine besondere Leistungsfähigkeit und nehmen komplexe Aufgaben, wie das Routing oder die Verwaltung der Peers, innerhalb des Netzwerkes wahr [GMY02, Gon01]. Ein Super-Peer in einem solchen Netzwerk agiert als zentraler Server für eine Anzahl von Clients. Diese richten ihre Anfragen an einen Super-Peer und erhalten Resultate von einem Super-Peer. Super-Peers sind über ein 'pures' Peer-to-Peer Netzwerk miteinander verbunden und leiten Nachrichten innerhalb dieses Netzwerkes weiter. Dieses Architekturmodell verteilt die Last für die Bearbeitung von Anfragen und die Verwaltung des Netzwerkes auf wenige vorwiegend statische

Super-Peer im Netzwerk. Ein Nachteil des Ansatzes ist die komplexe und dynamische Identifikation und Festlegung von Super-Peers [YGM03].

Hybrid. In diesen Systemen existiert ein zentraler Server, der die Interaktion zwischen den einzelnen Peers und deren Verwaltung über ein zentrales Verzeichnis steuert. Peers registrieren ihr Profil in einem zentralen Verzeichnis und stellen Anfragen, die mit den registrierten Profilen verglichen werden. Sind passende Peers für eine Anfrage identifiziert worden, erfolgt der Austausch von Dokumenten direkt zwischen zwei Peers.

Dieses Modell erfordert eine zentrale Infrastruktur um die Profile der einzelnen Peers zu verwalten. Diese Infrastruktur stellt einen *Single Point of Failure* dar. Bei einer Erhöhung der Anzahl von Peers im Netzwerk erhöhen sich auch die Kosten für die zentrale Infrastruktur. Ein Vorteil des Modells ist die Kontrolle der im Netzwerk publizierten Dokumente und angeschlossenen Peers. *Napster* [Nap03] und dessen Nachfolger *SnoCap* verwenden diese Architektur, um eine Kontrolle über die konsumierte Musik zu gewährleisten und Daten über das Verhalten der Nutzer zu sammeln.

2.3.2 Struktur des Overlay und der Indizes

Unter Struktur verstehen wir die zufällige oder durch Regeln definierte Anordnung der Peers und Dokumente in der Overlay Struktur. Die Anordnung der Dokumente und Peers kann über den Einsatz von Indizes vorgenommen werden. Wir unterscheiden globale Indizes, bei denen ein verteilter Index von allen Peers *gemeinsam* nach einem Muster verwaltet und für Anfragen ausgewertet wird und lokale Indizes, bei denen jeder Peer seinen *eigenen* Index verwaltet und für eigene Anfragen den eigenen oder den Index eines entfernten Peers benutzt.

Unstrukturiert ohne Indizes. In diesen Netzwerken ist die Anordnung von Dokumenten völlig unabhängig von der Overlay Struktur. Indizes werden nicht verwendet. Ein Funktion ist die Lokalisierung von Dokumenten. Typische Suchansätze überfluten das Netzwerk mit Anfragen durch Tiefen- oder Breitensuche, bis die gewünschte Anzahl von Dokumenten gefunden wurde oder die Anfrage eine bestimmte Entfernung, gemessen in der Anzahl kontaktierter Peers, erreicht hat [Rit01]. Der Nachteil unstrukturierter Modelle ist die Überflutung des Netzwerkes mit Anfragen.

Häufig werden diese Modelle mit Super-Peer Modellen kombiniert, um die Bearbeitung einer Anfrage und die damit verbundene hohe Anzahl von Nachrichten auf wenige Peers zu reduzieren [YGM03, TAA+03]. Eine weitere Form ist die Verwendung von *Random Walks*, wobei Anfragen an zufällig ausgewählte Peers

	Zentralisierung		
	Ohne	Super Peer	Hybrid
Unstrukturiert	Gnutella v0.4 FreeHeaven	Gnutella v0.6 Kazaa	Napster SnowCap
Semi-Strukturiert	ForeSeer INGA	GIA	
Strukturiert (Peers)	Structella	Edutella	
Strukturiert (Peers & Dok.)	CHORD CAN PASTRY P-Grid	TOPICS	

Abbildung 2.1: Klassifikation von Information Sharing Systemen

gesendet werden. In Verbindung mit einer Replikation von Dokumenten verringert dieser Ansatz signifikant die Anzahl der Nachrichten und die Zeit bis zum ersten Finden eines Dokuments [LCC+02].

Semi-strukturiert auf der Basis lokaler Indizes. Diese Netzwerke verfügen über ein unstrukturiertes Overlay. Durch Interaktionen mit benachbarten und entfernten Peers werden Informationen über deren publizierte Dokumente ausgetauscht und in einem lokalen Index verwaltet. Anfragen zwischen Peers werden auf der Basis dieser Indizes an Peers mit exakt passenden oder ähnlichen Dokumenten weitergeleitet. Das Overlay der initalen und meist zufällig zugeordneten Nachbarn eines Peers verändert sich mit der Anzahl der Interaktionen zu einem Overlay aus logisch zueinander gehörenden Nachbarn.

Die Autoren von [CGM02a] verwenden lokale Routing Indizes, in denen Informationen über verfügbare Dokumente der Nachbarn eines Peers gespeichert werden. Die Nachbarn sind durch die initiale Netzwerk Struktur bestimmt. Mit Hilfe der Indizes wird eine Anfrage nur an Nachbarn mit exakt passenden Dokumenten weitergeleitet. In *ForeSeer* [CW04] wird der Ansatz verbessert, indem die Dokumente der Nachbarn eines Peers mit Hilfe von *Bloom Filtern* [Blo70] zusammengefasst werden. Dadurch wird für den Austausch der Profile zwischen zwei Peers weniger Bandbreite benötigt. Zusätzlich werden den initalen Nachbarn weitere Nachbarn hinzugefügt, die in der Vergangenheit besonders häufig auf eine Anfrage antworten konnten. Eine Anfrage wird entweder an initiale Nachbarn oder an Nachbarn mit exakt passenden Dokumenten weitergeleitet. Im *GIA* Netzwerk [CRB+03] werden Anfragen an Nachbarn weitergeleitet, die eine hohen Anzahl

von Anfragen pro Sekunde verarbeiten oder eine hohe Bandbreite haben. In Kapitel 4 stellen wir das *INGA* System vor, bei dem ein Peer adaptiv seine Nachbarn in verschiedenen lokalen Indizes auf der Basis sozialer Metaphern und ähnlicher Interessen verwaltet. Eine Anfrage wird entweder an Nachbarn mit den ähnlichsten Interessen oder mit einer besonders guten Vernetzung weitergeleitet.

Unstrukturierte und semi-strukturierte Overlays eigenen sich besonders für Applikationen, bei den folgende Eigenschaften angenommen werden können [CRB+03, KBS02, ATS04]:

- Die Peers weisen eine hohe Dynamik auf.
- Das Netzwerk enthält vorwiegend populäre und häufig replizierte Dokumente.
- Die Nutzer sind mit einer unvollständigen Auswahl der besten Dokumente für eine Anfrage zufrieden gestellt.
- Die Nutzer möchten unmittelbar und lokal die im Index gespeicherten Daten kontrollieren können.
- Die Suche erfolgt über Keywords.

Strukturiert auf der Basis globaler Indizes. Motiviert durch die schlechte Skalierbarkeit unstrukturierter Netzwerke wurden Overlay Netzwerke entwickelt, die Peers und Dokumente durch fest definierte Regeln in eine Overlay Struktur einfügen. Peers und Dokumente werden durch Schlüssel beschrieben und in einem globalen Index auf der Basis einer verteilten Hashtabelle verwaltet. Wir unterscheiden zwischen Overlay Strukturen, die entweder *nur Peers* oder *Peers und Dokumente* im gemeinsamen Index verwalten.

- **Peers.** Gossiping Systeme benötigen eine Technologie zur Realisierung eines effizienten Broadcast, so dass jeder Peer eine Nachricht genau nur einmal erhält. *Structella* [MCR03] speichert alle verfügbaren Peers in einer verteilten Hashtabelle. Eine Nachricht wird in der verwendeten Ring Topologie im Uhrzeigersinn an jeden Peer geleitet. Zur Optimierung wird der Ring in $O(log\ N)$ Segmente aufgeteilt. Eine Nachricht wird an jeden ersten Peer jedes Segments gesandt und von diesem im Uhrzeigersinn an die nachfolgenden Peers innerhalb des Segments weitergeleitet.

  In *Edutella* [NWS+03] (siehe auch Kapitel 3.1) wird eine Overlay Struktur verwendet, um einen effizienten Broadcast einer komplexen Anfrage an

alle Peers zu realisieren und um neue Peers in Routing Indizes zu integrieren. Dazu werden die Peers in Form der HyperCup Topologie angeordnet [SSDN02], die eine Overlay Struktur auf der Basis von Caley Graphen bildet. Zur Optimierung des Broadcasts werden Routing Indizes und ein Super-Peer Netzwerk eingesetzt.

- **Dokumente und Peers.** In diesen Systemen wird jedem Dokument und jedem Peer ein eindeutiger Schlüssel zugewiesen. Die Overlay Struktur ist durch Regeln genau spezifiert. Dokumente (bzw. Verweise zu diesen) und Peers (bzw. deren Adresse) werden an genau definierten Stellen in das Overlay eingefügt. Diese Systeme verfügen über ein Mapping zwischen Schlüsseln von Dokumenten, Schlüsseln von Peers und deren Position im Overlay Netzwerk in Form einer verteilten Hashtabelle. Jeder Peer verwaltet einen Teil der Hashtabelle. Zur Lokalisierung eines Dokuments für eine Anfrage wird durch Anwenden einer Hashfunktion ein Schlüssel gebildet und die Anfrage wird zu dem Peer geleitet, der diesen Schlüssel verwaltet. Typische Beispiele sind *CAN* [RFH$^+$01], *CHORD* [SMK$^+$01], Pastry [RD01], *P-Grid* [Abe01] und *Tapestry* [ZKJ01]. In Kapitel 3.2 stellen wir das *TOPICS* System vor, welches semantische Profile von Peers in einer verteilten Hashtabelle speichert und ein Browsen in ihnen ermöglicht.

Strukurierte Overlays eigenen sich besonders für Anwendungen bei denen folgende Eigenschaften angenommen werden können [CRB$^+$03, KBS02, ATS04]:

- hohe Skalierbarkeit für eine hohe Anzahl von Peers und Dokumenten
- exakte Übereinstimmung der Anfrage mit einem Dokument bzw. dessen Schlüssel
- geringe Volatilität der Peers und Dokumente
- geringe Replikation der Dokumente

Durch Virtualisierung eines Schlüssels in einem strukturierten Netzwerk tritt jedoch das Problem auf, das für eine Anfrage der Schlüssel des Dokuments bereits zum Moment der Anfrage bekannt sein muss. Daraus resultieren zwei weitere Probleme [KBS02]:

- **Zerstörung der Lokalität.** Durch die Verwendung von Hash-Schlüsseln für die Identifizierung von Dokumenten wird die logische Lokalität der an einem Peer gespeicherten Dokumente zerstört. Möglichkeiten für das Browsing in den Dokumenten, das Pre-fetching der Dokumente und eine effiziente Suche in den Dokumenten innerhalb eines lokalen Peers gehen verloren.

- **Verlust von Informationen auf Anwendungsebene.** In Dateisystemen werden Dokumente typischerweise in nutzerspezifische Hierarchien eingeordnet. Innerhalb der Hierarchie stehen Dokumente die nahe in der Hierarchie gespeichert wurden auch häufig in enger logischer Beziehung. Durch die Verwendung von Hash-Schlüsseln für Dateien wird diese logische Beziehung jedoch zerstört.

Weitere offene Forschungsprobleme sind die Optimierung strukturierter Overlays für Netzwerke mit hohem Churn und Verfahren zum Lastausgleich innerhalb der verteilten Hashtabelle. In Abschnitt 3.2 stellen wir ausgewählte Verfahren dafür vor.

2.4 Zusammenfassung

Dieses Kapitel beinhaltet zwei Beiträge. Im ersten Teil definieren wir Merkmale von Anwendungen für die Informationsssuche in Peer-to-Peer Systemen und beschreiben anhand der Merkmale die Anwendungen 'Musiktauschbörse' und 'Semantisch vernetzte Arbeitsplätze'. Im zweiten Teil klassifizieren wir in der Praxis und Forschung verwendete Peer-to-Peer Systeme für die Informationssuche nach dem Grad der Zentralisierung und der Struktur der Overlays und klassifizieren die in dieser Arbeit vorgestellten Systeme, Edutella und TOPICS als ein Super-Peer basiertes System mit einem strukturierten, globalen Index und das INGA als ein semi-struktueriertes System mit lokalen Indizes.

3

Routing auf der Basis globaler Indizes

In diesem Kapitel werden zwei neue Ansätze für die Lokalisierung von mit Metadaten annotierten Ressourcen in einem Peer-to-Peer Netzwerk vorgestellt. Beide Ansätze unterstützen die Speicherung komplexer Metadatenstrukturen in einem vollständig verteilten Index. Der *Edutella* Ansatz [NWS+03, LNWS03, NWS+04] erlaubt die Formulierung und das Routing von Anfragen für komplexe Metadatenschema. Wir stellen den Aufbau des Index in Verbindung mit einem Super-Peer Netzwerk vor und erläutern das Routing-Verfahren am Beispiel der Metadatenschema Dublin Core und LOM. Der *TOPICS* Ansatz [Lös04b, LSZ04] ermöglicht das Browsen in verteilten Ressourcen, die über Metadaten in einer Themen - Hierarchie klassifiziert wurden. Wir stellen den Aufbau des globalen Index auf der Basis einer verteilten Hash Tabelle vor und zeigen das Browsen im Index.

3.1 Edutella

Die heutige universitäre Ausbildung ist durch eine stetig wachsende Anzahl von Studenten und eine immer geringeren Anzahl von betreuenden Lehrkräften gekennzeichnet. Diese Situation wird noch vervollständigt durch eine wachsende Anzahl von interdisziplinären Studienprogrammen und inhomogenen Studiengruppen. Zusammenfassend führen diese Faktoren zu einer immer mehr selbstgesteuerten und auch „unbetreuten“ Suche der Studenten nach passenden Materialien für Ihr Studienziel. Ebenfalls stehen Dozenten und Lehrer vor der Aufgabe, Materia-

lien für eine immer grössere Anzahl von Studenten mit heterogenen Leistungsvoraussetzungen zu erstellen. Dabei wird eine zu geringe Anzahl von existierenden Materialien, auch anderer Institutionen, wiederverwendet. Durch Initiativen, wie „Neue Medien in der Bildung“ oder das vom MIT initiierte Vorhaben *Open Course Ware Project* [OCW03], sind in den letzten Jahren viele autonome, weltweit verteilte Datenbanken mit qualitativ hochwertigen E-Learning Inhalten zu verschiedensten Studiengebieten entstanden [LGH02]. Aufgrund der Verteiltheit und der Heterogenität der Anbieter haben Nutzer des Netzwerkes jedoch grosse Schwierigkeiten für ihr Lernziel passende Materialien zu finden.

Das E-Learning Netzwerk, *Edutella* [NWQ+02] bietet eine technologische Plattform zur einer integrierten Bereitstellung heterogener, autonomer E-Learning Datenbanken in einem Netzwerk für den Web-weiten Zugriff der Studenten. Es wurde für ein Szenario mit folgenden Eigenschaften entwickelt:

Globaler verteilter Index. Aufgrund der fehlenden politischen und administrativen Vereinbarungen zwischen den Bildungsanbietern ist kein gemeinsamer Index möglich. Vielmehr soll jeder Bildungsanbieter einen kleinen Teil des Index lokal verwalten und durch eine geeignete Infrastruktur dem Nutzer einen globalen Zugriff ermöglicht werden.

Hoch verfügbare, autonome Quellen. Anbieter, wie beispielsweise Lehrstühle an Universitäten, stellen ihre Inhalte innerhalb der Plattform bereit, ohne die Autonomie über die Materialien zu verlieren. Die Quellen, bzw. die unterliegenden Rechner, weisen eine geringe Volatilität auf und können zusätzliche Dienste, wie das Routing von Anfragen, übernehmen.

Domänenspezifische Metadaten. Dokumente sind mit E-Learning Metadaten, beispielsweise nach dem LOM [IEE02] oder Dublin Core Standard [BMB02], annotiert.

Komplexe, schema-basierte Anfragen. Anfragen erfolgen in den Metadaten der Dokumente und können mehrere Prädikate beinhalten.

Edutella wurde für mehrere hundert weltweit verteilte Anbieter von Materialien und mehrere zehntausend Nutzer konzipiert. Ein blindes Versenden einer Anfrage an alle Anbieter ist nicht erwünscht. Vielmehr sollen für eine Anfrage passende Anbieter von E-Learning Materialien, abhängig von den in der Anfrage verlangten und an einer Quelle unterstützen Schema Elementen, ausgewählt werden. Zur Reduzierung der Nachrichten kombiniert das Edutella Netzwerk folgende drei Technologien:

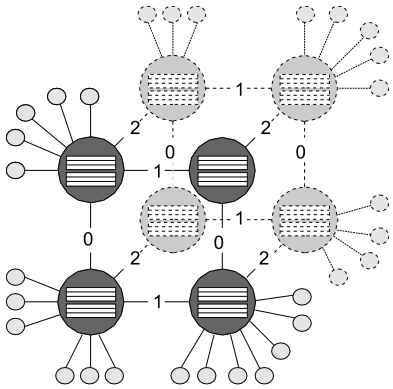


Abbildung 3.1: HyperCuP Super-Peer Topologie

Super-Peer Netzwerk. Das Netzwerk verwendet eine Super-Peer Netzwerk Struktur, bei der permanent verfügbare Peers sich zu statischen Super-Peer verbinden. Die Netzwerk Struktur erlaubt die lokale Verwaltung der spezifischen Ressourcen für die Publikation und Verwaltung von Inhalten an einem Peer, während der Super-Peer die notwendige Bandbreite und Rechenleistung für die Weiterleitung der Anfragen zur Verfügung stellt. Super-Peer basierte Peer-to-Peer Infrastrukturen (siehe auch Abschnitt 2.3.1) verfügen über eine zweistufige Routing Architektur: Zuerst wird eine Anfrage von einem Peer an in das Super-Peer Netzwerk (*Backbone*) geleitet. Abhängig von den in der Anfrage gesuchten Schema Attributen wird die Anfrage über das Super-Peer Netzwerk an weitere, mit dem Super-Peer Netzwerk verbundene Peers verteilt.

HyperCup Topologie. Für ein effizientes Weiterleiten von Anfragen innerhalb des Super-Peer Netzwerkes sind diese in Form der *HyperCup* Topologie [SSDN02] angeordnet. Ein Vorteil der HyperCup Topologie ist ein effizienter Broadcast von Anfragen. Während eines Broadcasts stellt jeder Knoten im Super-Peer Netzwerk die Wurzel für die Konstruktion eine Spanning Tree über das gesamte Peer-to-Peer Netzwerk dar. Ein Beispiel für die Anordnung der Super-Peers innerhalb der Netzwerk Struktur zeigt Grafik 3.1.

Ein neuer Super-Peer verbindet sich mit dem Netzwerk durch eine Anfrage an einen anderen, bereits verbundenen Super-Peer. Dieser leitet die Anfrage an weitere Super-Peers in der HyperCup Topologie. Um den neuen Super-Peer zu integrieren und bereits verbundene Super-Peers zu benachrichtigen, werden mit dieser Topologie $O(\log(N))$ Nachrichten versandt. Die Pfadlänge innerhalb der Topologie beträgt $\log_2 N$. Jeder Knoten verfügt

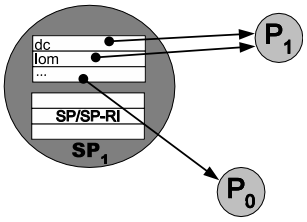


Abbildung 3.2: Super-Peer/Peer Routing Index

über eine $\log_2 N$ Nachbarn für jeden Knoten im Super-Peer Netzwerk, wobei N die absolute Anzahl von Knoten im Super-Peer Netzwerk darstellt.

Schema basierte Routing Indizes. Erhält ein Super-Peer eine Anfrage, sind zunächst alle an dem Super-Peer verbundenen Peers potentielle Kandidaten für eine Weiterleitung der Anfrage. Zur Reduzierung der Nachrichten in einem Super-Peer Netzwerk erfolgt das Routing auf der Basis von Indizes [NWS+03] und Routing Tabellen [CGM02a]. Der Edutella Ansatz verwendet zwei verschiedene Indizes: Super-Peer/Peer Indizes und Super-Peer/Super-Peer Indizes. Ziel ist ein Weiterleiten einer Anfrage nur an Peers, die eine Anfrage aufgrund ihres Schemas auch beantworten können.

3.1.1 Registrierung von Peers im Index

Für die Registrierung eines Peers an einem Super-Peer, und im Index, sendet der Peer eine *Advertise*-Nachricht an den Super-Peer. Sie beinhaltet Informationen über unterstützte Schemata des Peers. Jeder Peer muss mindestens ein Schema, beispielsweise eine Teilmenge des LOM oder Dublin Core Schemas, für eine erfolgreiche Registrierung definieren. Aufgrund der Volatilität der Peers ist eine Registrierung nur innerhalb eines bestimmten Zeitraums gültig. Wir wählen einen ähnlichen Ansatz wie im DHCP: Nach Ablauf der Zeit muss der Peer sich erneut an einem Super-Peer registrieren. Fällt ein Super-Peer aus, müssen sich die betroffenen Peers erneut an einem anderen Super-Peer registrieren.

Der Super-Peer/Peer Index. Er beschreibt die Schema Informationen aller an einem Super-Peer registrierten Peers. Er dient zur Auswahl von Peers für eingehende Anfragen. Darüber hinaus bildet er die Grundlage für einen weiteren Index, den Super-Peer/Super-Peer Index, der das Routing innerhalb des Super-Peer Netzwerkes bestimmt. Beide Indizes beinhalten Informationen über registrierte Peers auf folgenden Granularitätsstufen:

Schema Index. Wir nehmen an, dass verschiedene Peers unterschiedliche Schema unterstützen. Jedes Schema wird eindeutig über den verwendeten *Name Space* identifiziert. Der Super-Peer/Peer Routing Index beinhaltet einen eindeutigen Bezeichner für das Schema und die Peers, die das Schema unterstützen.

Property/Sets of Properties Index. Ein Routing Index kann auch Teile eines Schemas einem Peer zuordnen. Eine solche Teilmenge besteht aus einzelnen Attributen, sog. *Properties*. Sie werden über ihren Namen und das Schema, dem sie zugeordnet sind, eindeutig identifiziert. Jedem Attribut sind die Peers, die das Attribut unterstützen, zugeordnet.

Property Value Range Index. Einige Attribute beinhalten Werte aus bereits existierenden Themen-Hierarchien. Für diese Attribute können mögliche gültige Wertebereiche definiert werden. Applikationen im Bereich des E-Learning nutzen vordefinierte Wertebereiche, z.B. für die Einschränkung der Themen für eine Anfrage. In Beispiel 3.4 wird der Wertebereich für das Attribut dc:subject auf das Thema ccs:software-eng. eingeschränkt.

Property Value Index. Analog zu klassischen Datenbanken erlauben wir die Einschränkung auf spezifische Werte für ein Attribut. Der Index verweist auf einen Peer, der diese Attribut-Wert Kombination unterstützt. In Beispiel 3.4 wird das im E-Learning häufig unterstützte Attribut dc:language auf die deutsche Sprache eingeschränkt.

Der Super-Peer/Super-Peer Index. Diese Indizes repräsentieren Auszüge und Zusammenfassungen aus dem Super-Peer/Peer Index. Sie beinhalten die selben Metadaten wie Super-Peer/Peer Indizes, verweisen jedoch auf die direkten Nachbarn im Super-Peer Netzwerk (siehe auch Grafik 3.3). Anfragen werden zu benachbarten Super-Peers entlang des Super-Peer/Super-Peer Index weitergeleitet und dann über den Super-Peer/Peer Index an einzelne Peers verteilt.

Beispiel 3.1.1 (Edutella SP-SP Index). Ein Beispiel für einen Super-Peer/Peer Index zeigt Grafik 3.6. Tabelle 3.4 zeigt dazu die unterschiedlichen Granularitäten des Index an Super-Peer SP_2. Der Index beinhaltet alle direkten Nachbarn des Super-Peers(SP_1, SP_3, SP_4), die das Schema DC unterstützen. Ebenfalls beinhaltet der Index zwei Nachbarn, Peer SP_1 und Peer SP_4, die ebenfalls das LOM Schema unterstützen. Weiterhin beinhaltet der Index Einträge über den Wertebereich *Property Value Range* in Verbindung mit der ACM CCS[1] Themen-Klassifikation:

[1] http://www.acm.org/class/1998/

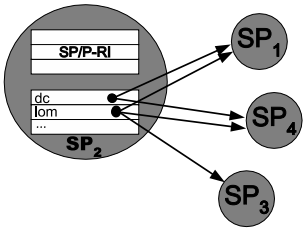


Abbildung 3.3: Super-Peer/Super-Peer Routing Index

Granularity	**Index of SP_2**		
Schema	dc		SP_1, SP_3, SP_4
	lom		SP_1, SP_4
Property	dc:subject		SP_1, SP_3, SP_4
	dc:language		SP_1, SP_4
	lom:context		SP_1, SP_4
Property Value Range	dc:subject	ccs:networks	SP_3
	dc:subject	ccs:software-eng.	SP_1, SP_4
Property Value	lom:context	"undergrad"	SP_1, SP_4
	dc:language	"de"	SP_1, SP_4

Abbildung 3.4: SP/SP Index von SP_2

ccs:networks ist ein übergeordnetes Konzept für ccs:ethernet und ccs:clientserver.

3.1.2 Bearbeitung von Anfragen

Die Aufgabe der verschiedenen Indizes ist die zusätzliche Reduzierung der Nachrichten für eine Anfrage. Dazu wird eine Anfrage über den Super-Peer/Peer und den Super-Peer/Super-Peer Index an die Super-Peers bzw. Peers weitergeleitet, die für diese Anfrage die benötigten Schema Elemente im Index registriert haben. Eine Übereinstimmung bedeutet, das ein Peer die Anfrage beantworten kann, garantiert jedoch keine nicht-leere Menge an Resultaten.

Beispiel 3.1.2 (Anfragebearbeitung in Edutella). Wir illustrieren die Funktionsweise der Indizes an folgendem Beispiel:

> Suche Lernmaterialien in deutscher Sprache aus dem Bereich Software-Engineering für Studenten im Grundstudium.

Granularität	Anfrage	
Schema	dc, lom	
Property	dc:subject, dc:language, lom:context	
Property Value Range	dc:subject	ccs:sw'engineering
Property *Value*	lom:context dc:language	"undergrad" "de"

Abbildung 3.5: Unterstützte Anfragen für unterschiedliche Granularitäten

Wir formalisieren diese Anfrage und verwenden Schema Elemente für Dokumente aus dem Schema DC und für Lernmaterialien aus dem Schema LOM. Zusätzlich verwenden wir die ACM CCS Hierarchie zur Klassifikation der Dokumente:

> *Suche jede Resource bei der die Eigenschaft dc:language mit "de", dc:subject mit ccs:softwareengineering und lom:context mit "undergrad" übereinstimmt.*

Tabelle 3.5 zeigt die notwendigen Granularitäten innerhalb der Anfrage, beispielsweise wird mit DC und LOM auf Schema Level zugegriffen und bei $lom : context = undergrad$ auf Werteebene usw. Grafik 3.6 zeigt, wie Peer P_0 die Anfrage an Super-Peer SP_1 sendet. Im unserem Beispiel kann die Anfrage von Peer P_1 und P_4 beantwortet werden. Diese Peers sind an Super-Peer SP_1 und SP_4 registriert, deren Index beinhaltet Metadaten über die Resources r and s die zur Anfrage passen. Auf der Basis einer Übereinstimmung im *Schema-Level Index* leitet Super-Peer SP_1 die Anfrage nur an Peer P_1 weiter, der die Schemata *lom* and *dc* unterstützt. Auf der Basis einer Übereinstimmung im *Property-level-index* wird die Anfrage von Super-Peer SP_1 an Peer P_1 weitergeleitet, da dieser Peer die Schema Elemente dc:subject, dc:language and lom:context unterstützt. Aufbauend auf dem Super-Peer/Super-Peer Index an SP_1 und SP_2 (siehe auch Abbildung 3.4) wird die Anfrage an den Super-Peer SP_4 weitergeleitet. Hier erfolgt ebenfalls ein Abgleich der Anfrage im Super-Peer/Peer Index. Aufgrund der Übereinstimmung der in der Anfrage geforderten Schema Elemente mit den von P_4 im Index registrierten Schema Elementen wird die Anfrage an P_4 weitergeleitet. $*$

3.1.3 Aktualisierung des Index

Strategie und Ziel. Eine Aktualisierung des Super-Peer/Peer Index an einem Super-Peer erfolgt dann, wenn ein neuer Peer sich im Index registriert, wenn die Schema Informationen über einen bereits registrierten Peer sich ändern oder wenn ein Peer den Index verlässt. Verlässt ein Peer den Super-Peer, werden alle Referenzen über den Peer im Index gelöscht. Dieser Fall tritt ebenfalls ein, wenn ein Peer

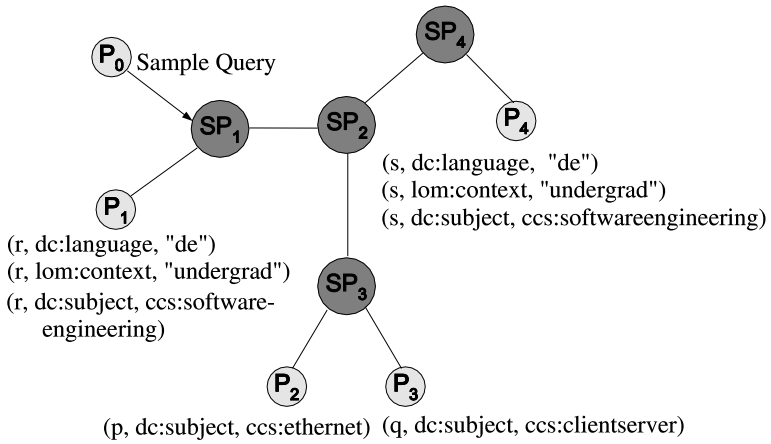


Abbildung 3.6: Beispiel für das Edutella Routing Modell

seine Registrierung nicht mehr regelmässig wiederholt. Registriert sich ein Peer an einem Super-Peer, werden die Schema Informationen des Peers mit den bereits am Super-Peer vorhandenen Informationen anderer Peers verglichen. Existiert bereits ein Eintrag für am Peer verwendetes ein Schema oder ein Attribut, wird nur der Peer im Index registriert, sonst wird das neue Schema oder das Attribut dem Index hinzugefügt. Der Index stellt einen invertierten Index dar, in dem verwendete Schema, Attribute usw. auf einzelne Peers abgebildet werden.

Algorithmus. Der Algorithmus für den Eintrag eines Peers an einem Super-Peer lässt sich wie folgt formalisieren:

- Wir definieren S als eine Menge von Schema Elementen. Wir verstehen unter einem Schema Element dabei einzelne Attribute und vollständige Schema, wie DC: $S = \{s_i \| i = 1...n\}$.
- Wir nehmen an, das ein Super-Peer bereits eine Menge von Schema Elementen S_x in seinem Super-Peer/Peer Index speichert. Wir verstehen unter dem Super-Peer/Peer Index eine Abbildung $s_i \mapsto \{P_j \| j = 1...m\}$.
- Wir definieren P_y als einen neuen Peer, der eine Registrierung am Super-Peer SP_x mit einer Menge von Schema Elementen ausführen möchte.

Der Algorithmus für die Registrierung eines Peers an einem Super-Peer lautet:

1. Wenn $S_y \subseteq S_x$, dann addiere P_y zur Liste der Peers an jedem $s_i \in S_y$

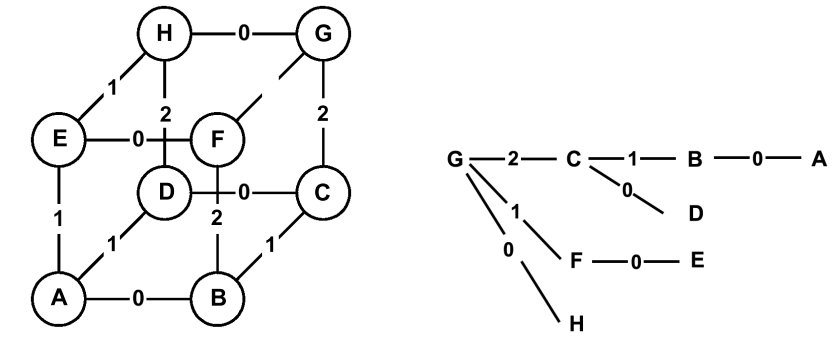


Abbildung 3.7: HyperCuP Topology and Spanning Tree Example

2. Sonst, wenn $S_y \setminus S_x = \{s_n, ..., s_m\} \neq \emptyset$, aktualisiere SP/P Index durch Addieren von neuen Zeilen $s_n \mapsto P_y, ..., s_m \mapsto P_y$.

Aktualisierung des Super-Peer/Super-Peer Index. Wir beschreiben die Aktualisierung des Super-Peer/Super-Peer Index im Super-Peer Backbone Netzwerk im Fall, dass an einem Super-Peer ein neuer Peer registriert wurde. Wir nehmen an, das jede Registrierung einen Aktualisierung des Super-Peer/Super-Peer Index auslöst. Da der Super-Peer/Super-Peer Index eine Zusammenfassung aller verwendeten Schema Elemente an einem Super-Peer repräsentiert, erfolgt eine Aktualisierung nur, wenn ein neues Schema Element hinzugefügt wird. Wir betrachten als Beispiel das Netzwerk in Grafik 3.6 und den Super-Peer/Super-Peer Index von Super-Peer SP_2, der in Tabelle 3.4 dargestellt ist. Ein neuer Peer P_x registriert sich an an Super-Peer SP_1 mit dem Attribut dc:language. Diese Registrierung löst keine Aktualisierung des Super-Peer/Super-Peer Index aus, da dieses Schema Element bereits im Super-Peer/Peer Index von SP_1 registriert war. Die Registrierung eines weiteren Peers P_y an Sp_1 löst eine Aktualisierung aus, da dieses Element im Super-Peer/Peer Index neu registriert werden muss.

Strategie und Ziel. Wie bereits in den vorhergehenden Abschnitten erwähnt wurde, ist das Super-Peer Netzwerk in Form einer HyperCup Topologie organisiert, die implizit jeden Super-Peer als Wurzel in einem *Spanning Tree* über das gesamt Super-Peer Netzwerk definiert. Ähnlich dem Weiterleiten von Anfragen agiert jeder Super-Peer ebenfalls als Wurzel für einen Spanning Tree, jedoch in der 'entgegengesetzten Richtung'. Grafik 3.7 zeigt das am Beispiel für Super-Peer G und einen einfachen Würfel, der drei Dimensionen aufspannt, so das jeder Knoten drei Nachbarn hat.

Für die Aktualisierung des Super-Peer/Super-Peer Index, nach einer vorherigen Aktualisierung des Super-Peer/Peer Index, erstellen wir einen Spanning Tree von Super-Peer G aus und gehen wie folgt vor: Super-Peer G sendet eine Nachricht zur Aktualisierung an alle seine Nachbarn. Jede Nachricht beinhaltet die Dimension der Kante, über die sie gesandt wird. Super-Peers, die die Nachricht erhalten, aktualisieren ihren Super-Peer/Super-Peer Index und leiten die Nachricht an alle Super-Peers mit einer geringeren Dimension weiter. Ein Super-Peer leitet eine Nachricht nicht weiter, wenn keine Aktualisierung des Super-Peer/Super-Peer Index erfolgt ist.

Algorithmus. Wir formalisieren den Algorithmus für die Konstruktion des Spanning Trees wie folgt:

- Für alle $s_i \in S_x \cap S_y$ füge die Dimension von SP_x der Liste der Dimensionen in der Spalte s_i hinzu, wenn die Dimension noch nicht existiert.
- Für alle $s_i \in S_x \setminus S_y$ füge eine neue Spalte $s_i \mapsto dimension(SP_x)$ ein.

3.2 TOPICS

Neben komplexen Schema-basierten Beschreibungen der Dokumente werden in verschiedenen Domänen Dokumente auch anhand semantischer Netze beschrieben. Aufgrund ihres einfachen Aufbaus und der geringen Anzahl semantischer Beziehungen zwischen den Konzepten wird häufig eine Themen-Hierarchie verwendet. Einzelne E-Learning Repositories bieten Lernmaterialien zu verschiedenen Themen an. Zur Klassifikation der Themen werden Themen-Hierarchien der *Global Network Academy*, der *ACM Library* oder des *Open Directories* verwendet. Ein weiteres Beispiel sind digitaler Musikplattformen. Kunden tauschen Musik, die zu einem bestimmten Genre gehört. *AllMusic* und *MusicMoz* klassifizieren eine halbe Million Musikstücke in einer Themen-Hierarchie nach Genre, Künstler und (Musik)Stimmung. Das *Topics Netzwerk* unterstützt die Lokalisierung von se-

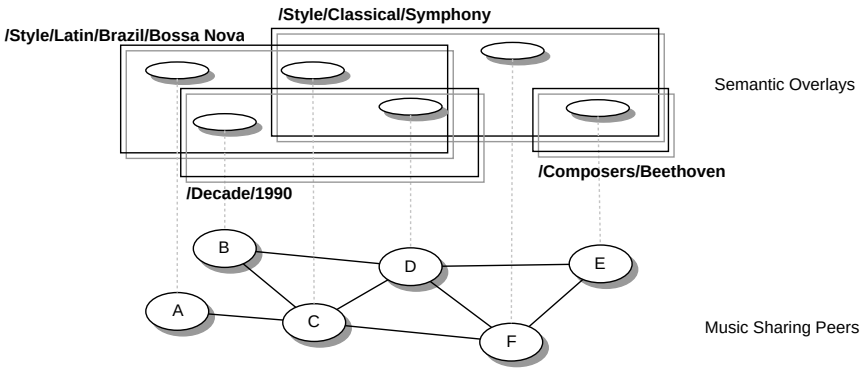


Abbildung 3.8: Semantisches Overlay Netzwerk

mantisch annotierten Dokumenten in einem verteilten Netzwerk. Es wurde für ein Szenario mit folgenden Eigenschaften entwickelt:

Hohe Volatilität der Peers. Dokumente können lokal auf den Rechnern der einzelnen Nutzer gespeichert werden. Aufgrund deren Autonomie können einzelne Peer das Netzwerk jederzeit verlassen oder beitreten.

Browsen. Gewünscht ist ein Browsen in der Themen-Hierarchie durch den Nutzer. Wird ein passendes Thema gefunden, soll eine Anfrage nach konkreten Dokumenten nur an die Peers geleitet werden, die passende Dokumente für das Thema bereitstellen.

Semantische Overlays Das TOPICS Netzwerk adaptiert *Semantic Overlay Networks* [LNS+03, CGM02b]. In derartigen Netzwerken wird das physikalische Netzwerk in Themen-basierte Overlays aufgeteilt. Abhängig von den an einem Peer verfügbaren Dokumenten gehört ein Peer einem oder mehreren Overlays an.

Beispiel 3.2.1 (Semantisches Overlay Netzwerk.). Grafik 3.8 zeigt sechs Peers, die Musik zu unterschiedlichen Genres bereitstellen. Peers die zum selben Thema gehören, formen ein *Semantic Overlay Network (SON)*, beispielsweise

$$\textit{/Style/Latin/Brazil/Bossa Nova} \mapsto (A, B, C, D)$$

$$\textit{/Sytle/Classics/Symphonies} \mapsto (B, C, D)$$

Die Grundzüge der TOPICS Architektur sind in Abbildung 3.9 dargestellt. Die Architektur besteht aus zwei Komponenten: einem verteilten Index und autonomen Peers, die sich im Index registrieren und in diesem browsen. Zur technischen Realisierung des Index kombiniert das TOPICS Netzwerk drei Komponenten:

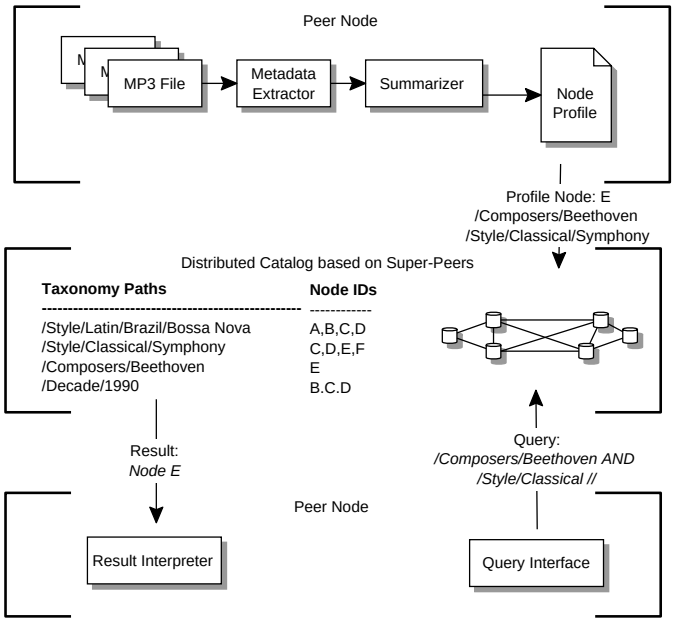


Abbildung 3.9: Architekturmodell für das Themen-Hierarchie-basierte Routing

Super-Peer Netzwerk. Ähnlich dem Edutella Netzwerk setzt sich der Index aus einem Super-Peer mit wenigen statischen Peers zusammen [GMY02]. Super-Peers erlauben Peers eine Registrierung und verteilen deren Profil im Index. Eine Anfrage eines Peers wird zuerst an das Super-Peer Netzwerk weitergeleitet, in dem die Suche im verteilten Index erfolgt. In einem zweiten Schritt wird das Resultat zum anfragenden Peer weitergeleitet.

Verteilte Hashtabellen. Sogenannte *Distributed Hash Tables (DHT)* eignen sich für Netzwerke mit geringer Volatilität und zur Suche von Objekten über exakte Schlüssel. Sie erlauben die Suche nach Objekten über deterministische Routing Algorithmen und skalieren für eine hohe Anzahl von Peers. Aufgrund des statischen Super-Peer Netzwerkes und der eindeutigen Identifizierung eines Themas in einer Themen-Hierarchie haben wir uns für diese Technologie entschieden.

Semantisches Hash-Schema. DHTs erlauben nur die Abbildung einfacher Strukturen von einen Schlüssel auf ein Objekt. Zur Abbildung und für den Zugriff auf die Struktur einer Themen-Hierarchie stellen wir in Abschnitt 3.2.1 ein geeignetes Konzept vor.

Die Operationen eines einzelnen Peers lassen sich in die Registrierung/Deregistrierung und das Browsen im Index aufteilen:

- **Registrierung/Deregistrierung.** Für die Registrierung eines Peers im Index werden zuerst die Metadaten aus den Dokumenten extrahiert. Aktuelle Metadaten für MP3, beispielsweise der ID3v2 Standard, stellen bereits eine einfache Genre Klassifikation zur Verfügung. Wir nehmen an, das in Zukunft jede Musikdatei über eine ausführliche Klassifikation verfügt.

  Eine weitere Komponente, der *Summarizer*, analysiert die Metadaten der Dokumente und fasst für jeden Peer die Beschreibungen zusammen. Ein Beispiel für eine Zusammenfassung in XML Notation wird im obigen Beispiel auf der rechten Seite gezeigt. Es besteht aus einer ID des Peers, der Dekade, der Anzahl der an diesem Peer zu der Dekade verfügbaren Dokumente, dem Genre und der Anzahl der Dokumente zu dem Genre. Das fertig berechnete Profil wird beim Eintritt des Peers im Index registriert. Verlässt der Peer den Index so wird das Profil gelöscht. Eine genaue Beschreibung der einzelnen Algorithmen für die (De-) und Registrierung von einzelnen Peers erfolgt in [Zim04].

Metadaten-Modell einer Datei:

```
<MP3File>
    <File>
        Jobim_Corcovado.mp3
    </File>
    <Decade>
        <1990/>
    </Decade>
    <Style>
        <Latin>
            <Brazil>
                <Bossa Nova/>
            </Brazil>
        </Latin>
    </Style>
</MP3File>
```

Metadaten-Modell eines Peers:

```
<Node>
    <ID>D</ID>
    <Decade>
        <1990/>
        <Num>33</Num>
    </Decade>
    <Style>
        <Latin>
            <Brazil>
                <Bossa Nova/>
            </Brazil>
        </Latin>
        <Num>21</Num>
    </Style>
    <Style>
        <Classical>
            <Symphony/>
        <Classical>
        <Num>3</Num>
    </Style>
</Node>
```

- **Browsen im Index.** Ein Peer kann den Index nach einem Genre oder einer Dekade anfragen. Für die Anfrage verwenden wir eine Teilmenge der

verbreiteten XPATH Sprache[W3C03], die einen guten Kompromiss zwischen Anfragemöglichkeit und Einfachheit darstellt und die Auswahl von Teilbäumen erlaubt. Ein Dokument stimmt mit einem XPATH Ausdruck überein, wenn die Auswertung des Ausdruckes ein nicht leeres Ergebnisobjekt erzeugt. Der folgende Ausdruck wählt Peers aus, die Musikdateien zum Genre *Bossa Nova* publizieren:

$$q1 = /Node/Style/Latin/Brazil/BossaNova/$$

Das Ergebnis der Anfrage ist eine Menge S_i von Peer, die mit der Anfrage übereinstimmen, beispielsweise für die Anfrage $S_{q1} = (A, B, C, D)$.

3.2.1 Abbildung von Themen-Hierarchien in einer DHT

Das Ziel von *Distributed Hash Tables* (DHT), wie CAN [RFH+01], Chord [SMK+01], Pastry [RD01], P-Grid [Abe01] und Tapestry [ZKJ01], ist das effiziente Auffinden von Daten in einer verteilten, dezentralen Infrastruktur. Dazu besitzen alle Systeme einen Routing-Algorithmus, der die Operation `lookup(key)` bei N Knoten im Netzwerk unter Verwendung von $\mathcal{O}(\log N)$ Nachrichten ausführen kann. Dieser Schlüssel wird dynamisch einem im Netzwerk existierenden Knoten zugewiesen, der auch als Root des Schlüssels bezeichnet wird. Zur eindeutigen Zuweisung von Schlüsseln zu Knoten besitzt in einer DHT jeder Knoten eine eindeutige ID. Zusätzlich wird eine Funktion zur Zuweisung von Schlüssel zu Knoten-ID definiert. Um Nachrichten effizient zum Root eines Schlüssels zu routen, wird über der physikalischen Netzwerkebene noch eine logische Ebene zum Routing mittels Schlüsseln erstellt. Dazu besitzt jeder Knoten eine Routingtabelle mit Knoten-IDs und entsprechenden IP-Adressen bestimmter Knoten des Netzwerkes. Zum Root eines Schlüssels wird geroutet, indem aus der Routingtabelle ein Knoten ausgewählt wird, dessen ID sukzessive näher am Schlüssel ist.

Zur Realisierung der Abbildung der Beziehungen der Themen-Hierarchie bietet die DHT die Methoden `put(key, data)` und `lookup(key)` um beliebige Daten unter einem Schlüssel zu speichern, beziehungsweise die Daten unter einem Schlüssel zu erfragen. Themen sind in einer Themen-Hierarchie eindeutig über ihren Pfad definiert, den wir als Schlüssel verwenden.

Definition 3.2.1 (Schlüssel für ein Thema). *Sei t ein Thema, t_{path} der Pfad durch den das Thema definiert ist und h eine Hashfunktion dann ergibt sich der Schlüssel $k_{t_{path}}$ für t durch*

$$k_t = h(t_{path})$$

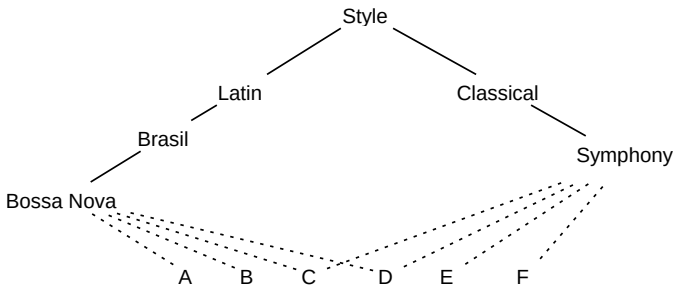


Abbildung 3.10: Verteilung von Dokumenten über eine Themen-Hierarchie

Beispiel 3.2.2. Sei t das Thema *Bossa Nova* definiert durch */Style/Latin/Brazil/-Bossa Nova* dann wird der Schlüssel berechnet als

$$\$EA66 = h(/Style/Latin/Brazil/BossaNova)$$

Unter jedem Schlüssel speichern wir die Peers, die Dokumente für das Thema im Index registriert haben. Dazu verwenden wir das Konzept der invertierten Liste.

Definition 3.2.2 (Objekt für einen Schlüssel (1)). *Unter jedem k_t speichern wir alle IDs passender i Peers $PID_1...PID_i$ für t.* ♣

Beispiel 3.2.3 (Objekt für einen Schlüssel (1)). Für die in Abbildung 3.8 beschriebenen Peers beinhaltet die verteilte Hashtabelle folgende Einträge:

Hash Key	Objekt
$EA66	A,B,C,D
$CCEE	C,D,E,F
$AB55	B,C,D
$6520	E

```
$EA66 /Style/Latin/Brazil/Bossa Nova
$CCEE /Style/Classical/Symphony
$AB55 /Decade/1990
$6520 /Composers/Beethoven
                                  *
```

Werden unter einem Schlüssel k_t keine Peers gefunden, so soll die Suche bei den in subsumierender Beziehung zu t stehenden Termen fortgesetzt werden können.

Definition 3.2.3 (Objekt für einen Schlüssel (2)). *Getrennt von den IDs der Peers werden unter einem k_t alle Terme $t' \sqsubseteq t$ gespeichert. Dazu wird für jedes Thema $t \in t_{path}$ mit Hilfe der Hashfunktion h ein Schlüssel berechnet, beginnend mit dem spezifischsten Thema.* ♣

Beispiel 3.2.4 (Objekt für einen Schlüssel(2)). Wir wenden die Hashfunktion auf jeden Teilpfad an:

```
$EA66 /Style/Latin/Brazil/Bossa Nova
$AB33 /Style/Latin/Brazil
$3A56 /Style/Latin
$FF7A /Style

$CCEE /Style/Classical/Symphony
$BA7F /Style/Classical
```

Zur Erhaltung der $Subsumption$ Beziehung speichern wir im Index für das nächstspezifischere Thema t_i mit $t \sqsubseteq t_i$ ein Mapping $\sqsupseteq (t_i, t)$:

Hash Key	**Objekt**
$FF7A	$\sqsupseteq$ /Style/Classical $\sqsupseteq$ /Style/Latin
$BA7F	$\sqsupseteq$ /Style/Classical/Symphony
$3A56	$\sqsupseteq$ /Style/Latin/Brazil
$AB33	$\sqsupseteq$ /Style/Latin/Brazil/Bossa Nova/

*

Die vollständige Hashtabelle speichert im Objekt eines Schlüssel sowohl die registrierten Peers und vorhandene Beziehungen:

Hash Key	**Objekt**
$3A56	$\sqsupseteq$ /Style/Latin/Brazil
$AB33	$\sqsupseteq$ /Style/Latin/Brazil/Bossa Nova/
$BA7F	$\sqsupseteq$ /Style/Classical/Symphony
$CCEE	C,D,E,F
$EA66	A,B,C,D
$FF7A	$\sqsupseteq$ /Style/Classical $\sqsupseteq$ /Style/Latin

3.2.2 Semantisches Browsen in einer DHT

Die semantischen Strukturen des verteilten Index erlauben einem Nutzer des Browsen im Index. Dabei treten folgende drei Fälle auf.

Exakte Übereinstimmung. Eine Beispiel für eine exakte Anfrage ist *Gib alle Peers, die Musik von Beethoven anbieten*:

$$q1 = /Composers/Beethoven/$$

Die Berechnung der PIDs unterscheidet sich nicht von einem Standard Lookup in der verteilten Hashtabelle. Für $q1$ wird der Hashwert $\$6520$ entsprechend Definition 3.2.1 berechnet. Mit der Funktion $lookup(h(q1))$ wird folgende PID für $q1$ ermittelt: q1=E. Da nur ein Lookup notwendig ist betragen die Kosten für den Lookup $O(log\ N)$ Nachrichten.

Spezielleres Thema. Der Nutzer kennt nur den Pfad für ein spezielleres Thema. Ein Beispiel für eine solche Anfrage ist: *Gib alle Peers mit Bossa Nova Gitarren Stücken*:

$$q2 = /Style/Latin/Brazil/BossaNova/Guitar/$$

Eine Anwendung der Hash Funktion $h(q2)$ und der Funktion $lookup(h(q2))$ im Index erfolgt mit einer leeren Resultatmenge. Der Nutzer wählt das nächstgenerelle Thema aus der Struktur der Anfrage.

$$q2' = /Style/Latin/Brazil/BossaNova/$$

und ermittelt das Resultat $q2' = \{A, B, C, D\}$

Generelleres Thema. Durch die Speicherung der semantischen Beziehungen in der verteilten Hashtabelle können wir für eine Anfrage auch Peers berücksichtigen, deren Inhalte auch für ein spezielleres Thema registriert sind, als das in der Anfrage definierte Thema. Ein Nutzer kann dadurch den semantischen Baum iterativ nach spezielleren Themen durchsuchen und ist damit nicht mehr auf einen exakte Übereinstimmung mit der Anfrage angewiesen. Ein Beispiel für eine solche Anfrage ist: *Gib mir alle Peers die klassische Musik anbieten*:

$$q3 = /Style/Classic//$$

Wiederum wird auf $q3$ die Hashfunktion angewendet und das Ergebnis im Index angefragt mit der Funktion $lookup(h(q3))$. Das Ergebnis ist in diesem Fall ein Verweis auf die Unterkategorie $/Style/Classics/Symphony$. Da der Nutzer keine passenden Peers erhalten hat, bildet er eine neue Anfrage $q3'$. Für $q3'$ wird iterativ ein weiterer Lookup ausgeführt $lookup(h(q3'))$, das Ergebnis ist q3' = {C,D,E,F}. Die Anzahl der Nachrichten steigt mit der Anzahl der besuchten Mappings m. Die gesamten Kosten für die eine solche Anfrage betragen $O(log\ N) * |m|$.

3.2.3 Lastverteilungstrategien

Die unterschiedliche Popularität und Nachfrage einzelner Themen belastet die Super-Peer Knoten des globalen Index unterschiedlich stark. Im ungünstigsten Fall erhält

ein Super-Peer eine hohe Anzahl von Anfragen bzw. eine hohe Anzahl von Registrierungen von Peer-Modellen innerhalb einer kurzen Zeitspanne. Diese ungleiche Verteilung kann zur völligen Auslastung einzelner Super-Peers führen, während auf anderen Super-Peers noch freie Kapazitäten verfügbar sind. Die Folge ist eine schlechte Skalierung des verteilten Index für Anfragen mit besonders populären Themen. In Peer-to-Peer Systemen wird diesem Problem durch Verfahren der Lastverteilung (*Load Balancing*) begegnet. Wir unterteilen das Problem der *Lastverteilung* für unser Szenario in folgende Teilprobleme:

- Lastverteilung für die Registrierung im Index
- Lastverteilung von Anfragen an den Index

Wir stellen ausgewählte State-Of-The-Art Ansätze am Beispiel von CHORD [SMK+01] kurz vor und wenden diese auf unser Szenario an. Aufgrund der Komplexität der einzelnen Ansätze verweisen wir für eine ausführliche Diskussion auf die im folgenden Abschnitt aufgeführte Literatur. Eine Implementierung der vorgestellten Ansätze für das Szenario erfolgte in [Zim04].

Lastverteilung für die Registrierung im Index

Um den globalen Index aktuell zu halten, registriert jeder Peer bei jedem Eintreten in das Netzwerk und innerhalb eines bestimmten Zeitintervalls, beispielsweise einmal innerhalb 3600 Sekunden, sein vollständiges Model im Index. Dadurch entsteht Last an den Super-Peers für die Speicherung der Modelle der Peers und für die Übertragung der Modelle. Durch die Zipf-verteilte [Zip49] Popularität der Themen erhalten Super-Peers, die für besonders populäre Themen zuständig sind, auch besonders viele Anfragen von Peers für eine Registrierung. Dadurch entsteht an diesen Peers ein erhöhtes Aufkommen von Nachrichten und ein erhöhter Speicherplatzbedarf für die Speicherung der Modelle.

Eine Lösungsvariante für das CHORD Netzwerk ist das in [SMK+01] vorgestellte Konzept der *Virtual Server*. Jeder Super-Peer besitzt mehrere virtuelle IDs mit denen er an der DHT teilnimmt und so vorgibt, mehrere nicht zusammenhängende virtuelle Index Knoten zu repräsentieren. Die Grösse des Adressraumes eines Super-Peers ergibt sich damit aus der Summe der einzelnen Grössen der Adressräume der virtuellen Index Knoten. Wird die Last für einen Super-Peer zu hoch, kann ein virtueller Index Knoten, und damit nur ein Teil des Adressraumes, von einem überlasteten Super-Peer auf einen weniger belasteten Super-Peer transferiert werden. Das beinhaltet den Transfer der Schlüssel und der unter dem Schlüssel registrierten Objekte. Ein Transfer des virtuellen Super-Peers wird durch die Chord-Operationen *leave* und *join* durchgeführt. Das Ziel ist die Optimierung

der überlasteten Super-Peers (*Heavy Nodes*) durch den Transfer zu unbelasteten Knoten (*Light Nodes*). Dazu speichert der Lastverteilungsalgorithmus die Lastinformationen der Super-Peers in einer Reihe von Verzeichnissen. Diese Verzeichnisse führen periodisch eine Neuzuweisung von virtuellen Super-Peers durch, um die Last besser zu verteilen. Algorithmen für die Neuzuweisung der Verzeichnisse werden für statische Netzwerke in [RLS+03] und für dynamische Netzwerke in [GBL+04] vorgestellt.

Lastverteilung der Anfragen an den Index

Durch die Zipf-Verteilung [Zip49] der unterschiedlich populären Anfragen [Sri01, SGG03] werden einige Knoten mit einer besonders hohen Anzahl von Anfragen belastet [YRS02]. Bei einer Lastverteilung mit dem *Virtual Server* Ansatz wird nur die ungleiche Last beim Einfügen von Modellen berücksichtigt. Ein speziell auf Anfragen ausgerichteter Algorithmus würde diese Lastverteilung nochmals verbessern.

Verschiedene Designer [SMK+01, RFH+01] von Peer-to-Peer Systemen haben das Caching von Index Einträgen entlang des Anfragepfades vorgeschlagen. Diese Strategie wird auch als *Path Caching with Expiration* (PCX) bezeichnet, da die im Cache vorhandenen Ergebnisse nach einer bestimmten Zeit als veraltet angesehen und gelöscht werden. PCX ist wünschenswert, da es die Last für populäre Anfragen über mehrere Knoten verteilt, die Latenz reduziert und Hot Spots vermeidet. Im Chord Ansatz gibt *O(log n)* Verweise auf einen Knoten des Netzwerkes. Im letzten Routingschritt zu einem Knoten kann eine Anfrage nur von einer kleinen Anzahl anderer Knoten stammen. In PCX erfolgt ein Cachen entlang des Anfragepfades sinnvoll, da auch bei wenigen im Cache vorhandenen Ergebnissen mit hoher Wahrscheinlichkeit eine Anfrage auf ein bereits im Cache vorhandenes Ergebnis trifft bevor sie den eigentliches Zielknoten erreicht. In diesen Fällen kann die Anfrage sofort aus den Daten im Cache beantwortet werden. Dadurch wird die Bandbreite für das Routing der Anfrage bis zu ihren eigentlichen Zielknoten und die Bandbreite für das Routing der Ergebnisse zurück eingespart. Zusätzlich wird die Antwortzeit von Anfragen verringert, da weniger Routingschritte benötigen werden.

Eine wichtige Frage ist die Aktualität des Caches in einem Peer-to-Peer Netzwerk. Aufgrund der hohen Dynamik der Peers veralten die Inhalte der Caches schnell. Die Zeitdauer der Gültigkeit eines Eintrags im Cache muss bei *PCX* kurz sein, wenn keine veralteten Ergebnisse zurückgeliefert werden sollen. Eine Folge davon ist, dass in bis zu 50 % aller Fälle eine Anfrage auf einen passenden und auch gültigen Cacheeintrag trifft, diesen aber nicht benutzen kann, da er als veraltet angesehen wird [CK01]. *Roussopoulos* [RB03] schlägt zur Optimierung das

Controlled Update Propagation (CUP) Verfahren vor, um Caches aktuell zu halten. Knoten können sich an bestimmten Schlüsseln registrieren, so dass sie Updates für Inhalte unter diesem Schlüssel erhalten. Für jeden Schlüssel zu dem ein Knoten Anfragen erhält, wird die Anzahl der Anfragen in einer bestimmten Zeit registriert. Anhand einer Metrik aus der Popularität, der Entfernung in Routingschritten von dem für den Schlüssel verantwortlichen Super-Peer und der Häufigkeit von Updates für einen Schlüssel, kann ein Knoten berechnen ob es sich lohnt, weiter Updates für den Schlüssel zu erhalten oder sich für Updates zu registrieren. Erhält ein Knoten ein Update, so spart sich der Knoten die Kosten für das Weiterleiten der Anfrage und den Empfang des Suchergebnisses, hat dafür aber Kosten für den Empfang der Updates zu erbringen. Wenn die Kosten der Einsparungen den Overhead übersteigen, so entscheidet sich ein Super-Peer dafür Updates für einen Schlüssel zu erhalten. Ist es nicht mehr von Vorteil Updates zu erhalten, so de-registriert sich ein Knoten am Schlüssel. Simulationen der Autoren von CUP haben gezeigt, dass je mehr Anfragen an ein System gestellt wurden, desto grösser waren die Einsparungen mit CUP an Bandbreite und desto geringer war die Latenzzeit von Anfragen im Vergleich zu PCX.

3.3 Verwandte Arbeiten

Verteilte Datenbanken [Vz99] benutzen Kataloge zur Speicherung von Informationen über die Fragmentierung der Datenbank. Jede Datenbank verfügt über eine eigene Replica des Kataloges und berechnet auf dieser Grundlage, ob eine Anfrage lokal oder an einer entfernten Datenbank ausgeführt werden soll. Da alle Datenbanken eines verteilten Datenbank Managementsystems (DDBMS) unter der Kontrolle einer einzelnen Autorität verbleiben, ist die Anzahl der einzelnen Datenbanken beschränkt.

Globale Kataloge zur Lokalisierung von Datenquellen und Dateien wurden kürzlich zur Optimierung der Auswahl von verteilten, text-basierten Datenquellen für eine Anfrage vorgestellt. Im Gegensatz zu TOPICS und Edutella basiert der Ansatz von [FPV+98] auf einem zentralen Verzeichnis, das einen unerwünschten *Single Point of Failure* darstellt. Zur Vermeidung dieses Problems stellen [GGM95] [DADA96] einem hierarchischen Broker Ansatz vor, der jedoch zusätzlichen administrativen Aufwand für die Pflege der asymmetrischen Netzwerk Struktur benötigt. Die in diesem Kapitel vorgestellten Ansätze TOPICS und Edutella unterscheiden sich von dieser Design-Philosophie, da sie keine zentrale Kontrolle des Systems voraussetzen.

Eine ähnliche Idee wie TOPICS und Edutella verfolgen verteilte Publish/Subscribe Ansätze. Ein Beispiel für ein solches System ist *INS/Twine* [BBK02]. Die

Beschreibungen der Ressourcen werden in einer verteilten Hashtabelle gespeichert und abgefragt. Zur Vermeidung einer ungleichen Belastung der Knoten in der DHT werden die Beschreibungen gleich über die Knoten verteilt. Zusätzlich wird jede Beschreibung redundant gespeichert. Das System unterstützt keine semantischen Beziehungen zwischen den Beschreibungen einzelner Ressourcen.

SCRIBE [RKCD01] realisiert einen Multi-Cast auf Anwendungsbene. Es basiert auf dem DHT-Protokoll *Pastry* [RD01]. Nutzer und Nachrichten können thematisch Gruppen zugeordnet werden. Nachrichten werden dann innerhalb der Gruppe an alle Mitglieder weitergeleitet. Das System erlaubt nur ein exakten Vergleich zwischen dem Thema einer Nachricht und der Gruppe.

Galanis beschreibt in [GWJD03] einen Katalog der ebenfalls auf der Basis verteilter Hashtabellen realisiert wird. Der Ansatz speichert XML Schemata in einem globalen verteilten Index. Anfragen erfolgen auf der Basis komplexer XML-Schemata. Das Matching ist jedoch auf exakte Übereinstimmungen zwischen einer Anfrage und dem Schema begrenzt und unterstützt keine semantischen Beziehungen.

Ein Vision für die semantische Interoperabilität wird von Bernstein in [BGK+02] beschrieben. Er stellt das *(Local Relational Model) LRM* vor, das die Transformation von Anfragen aufgrund der lokalen Schemata der einzelnen Peers erlaubt. Durch Query-Rewriting wird die Anfrage über Mapping Regeln auf lokale Schemata abgebildet.

Ähnlich dem Edutella Ansatz integriert das *PIAZZA* Projekt [HIMT03] verschiedene Informationsquellen in ein Peer Data Management System. Es verfügt über zusätzliche Mappings zwischen den Schemata der einzelnen Peers und leitet Anfragen entlang der Schemata. In Kontrast zu Edutella verwendet PIAZZA ein hybrides Peer-to-Peer Modell. Die Indizierung der einzelnen Peers und das Mapping erfolgt in einer zentralen Komponente.

3.4 Zusammenfassung

Ziel dieses Kapitels war die Vorstellung von Konzepten zur Realisierung eines globalen und verteilten Index. Er soll die Registrierung und Lokalisierung von Ressourcen mit komplexen Metadatenstrukturen unterstützen. Dazu stellen wir zwei neue Frameworks, Edutella und TOPICS, vor. Beide Ansätze verwenden einen globalen, jedoch vollständig verteilten Index in einem Peer-to-Peer Netzwerk. Ein solcher Index ist besonders für Szenarien interessant, in denen keine finanziellen oder administrativen Voraussetzungen für eine zentrale Infrastruktur geschaffen werden können, jedoch eine kleine Anzahl permanent verfügbarer Rechner dezentral vernetzt ist. Technologisch realisieren beide Ansätze den Index über eine Super-

Peer Infrastruktur. Zur Verringerung der Nachrichten für das Routing und die Registrierung der Peers verwenden wir in Edutella neben der Super-Peer Infrastruktur einen effizienten Broadcast Algorithmus und einen zweigeteilten Routing-Index. Der TOPICS Ansatz verfügt über eine Index Struktur auf der Basis verteilter Hashtabellen, die im Super-Peer Netzwerk verwaltet wird. Im Index speichern wir die semantische Struktur einer Themen-Hierarchie und Peers, die ein konkretes Thema unterstützen.

Teil II

Adaptive Overlays auf der Basis lokaler Shortcut Indizes

4
Der INGA Ansatz

Die Suche in Peer-to-Peer Netzwerken kann sowohl mit einem globalen Index und mit lokalen Index Strukturen ermöglicht werden. In dieser Arbeit betrachten wir beide Ansätze. Im letzten Kapitel haben wir bereits Overlay Netzwerke auf der Basis eines globalen und in einem statischen Super-Peer Netzwerk verteilten Index vorgestellt. In diesem Kapitel entwickeln wir eine neue Art von Overlay Netzwerken auf der Basis von Shortcuts, die wir in lokalen Indices speichern. Die grundlegende Idee des INGA Ansatzes (*Interest based Node Grouping Architecture*) ist es, die Anzahl der anzufragenden Peers durch die Analyse bereits gestellter erfolgreicher Anfragen des eigenen oder anderer Peers einzuschränken. Jeder Peer verwaltet dazu Informationen über Shortcuts, Anfragen und erfolgte Antworten in einem lokalen Index. Zum Aufbau der Shortcuts stellen wir soziale Metaphern auf und führen kurz in den Zusammenhang zwischen dem Small World Phänomen und sozialen Netzwerken ein. Zum Abschluss dieses Kapitels zeigen wir die Architektur eines Peers im INGA Netzwerk.

4.1 Anforderungen

In den letzten Jahren sind in Forschung und Industrie verstärkt drei Trends zu beobachten: eine Explosion unstrukturierter lokaler Daten, die Notwendigkeit der Strukturierung dieser Daten und der Austausch von Daten und Strukturen in und zwischen einzelnen Unternehmen. Aufgrund der Unübersehbarkeit und Unüber-

schaubarkeit von Personen und vorhandenen Dokumenten wird der Informationssuche in grossen Unternehmen und verteilten Organisationen ein wesentlicher Anteil der gesamten Arbeitszeit gewidmet. Verschiedene Anbieter haben das Problem erkannt und stellen Software für das Durchsuchen des lokalen Desktops bereit. Erfolgreiche Beispiele dafür sind *Google Desktop Search*, die *Microsoft Desktop Search* oder das *Copernic Search System*.

Ein Großteil aller relevanten Informationen wird in einem Unternehmen jedoch 'Peer-to-Peer' - von Kollege zu Kollege - ausgetauscht, aber nur wenn man den 'richtigen Kollegen' auch kennt. Diese Funktionalität der vernetzen Desktop Suche wird jedoch von den derzeitig existierenden lokalen Desktop Such Technologien noch nicht zur Verfügung gestellt. Zur Reduzierung des Aufwandes und der Kosten der Informationssuche in einem Unternehmen und seinem Netzwerk, wie Partnern und Kunden, werden neben den verfügbaren Web-Suchmaschinen immer häufiger lokale Suchwerkzeuge benötigt, die für eine Suche im *vernetzten Unternehmen* optimiert sind. Nur wenn ein Unternehmen die Wünsche seiner Kunden kennt, kann es seine Produkte und Dienstleistungen gezielt auf deren Bedürfnisse abstellen. Nur wenn ein Unternehmen das kollektive und individuelle Wissen seiner Mitarbeiter erschließt, verkürzt es seine Produktentwicklungszeiten, minimiert die Gefahr von Fehlentwicklungen und macht sich unabhängig von einzelnen Wissensträgern und gewinnt die entscheidenden Wettbewerbsvorteile, die den langfristigen Erfolg sicherstellen. Im privaten Bereich existieren bereits erfolgreiche Peer-to-Peer Anwendungen, wie Gnutella und Kazaa, die eine einfache lokale Publikation und Suche von Musik-Dateien in einem Netzwerk ermöglichen. Da kein Single Point of Failure existiert, sind diese Netzwerke robust. Durch ihre Selbstorganisation benötigen sie keine zentrale Administration und erlauben durch den verteilten Ansatz keine zentrale politische Kontrolle. Bisher wurden diese Technologien jedoch nur für die Internet-weite Suche nach Medienclips verwendet.

Inspiriert durch [FFK04] schlagen wir vor, durch den Austausch und die Suche von Dokumenten innerhalb und ausserhalb institutioneller Barrieren eine kollaborative Wissens - Koproduktion zu ermöglichen bzw. deutlich zu vereinfachen. Wir sehen das Einsatzspektrum von Peer-to-Peer Anwendungen in Unternehmen mit einem hohem Potential an *Knowledge Workern*, die an ihren lokalen Desktops Dokumente zu ihren Interessen ansammeln und neue Dokumente produzieren. In diesem und den folgenden Kapiteln kombinieren wir in der *Interest-based Node Grouping Architecture*, kurz INGA Desktop Netzwerk, die Vorzüge der verteilten Suche, die Selbstorganisation und die Robustheit von Peer-to-Peer Technologien mit der Suche in lokalen Desktops. Unser Ziel ist die Beschreibung von technologischen Konzepten für einen neuen Typ von Infrastrukturen für den Austausch und die Suche von lokal verfügbaren Dokumenten.

In der Theorie eigenen sich Peer-to-Peer Netzwerke ideal für die Realisierung eines vernetzten Desktop-Netzwerkes. Ein Hauptproblem in der Praxis besteht jedoch in einer kostengünstigen und effizienten Realisierung der *Suche* in den verteilten Desktops. Prinzipiell stehen dafür zwei Typen von Peer-to-Peer Netzwerken zur Verfügung: Unstrukturierte [Gnu04] und strukturierte [SMK+01, RFH+01, RD01, Abe01] Peer-to-Peer Netzwerke. Im Folgenden stellen wir ausgesuchte Designmerkmale einer Infrastruktur zur Desktopsuche vor und diskutieren die Vor- und Nachteile der jeweiligen Peer-to-Peer Konzepte:

- **Lokale Kontrolle.** Die Akzeptanz des INGA Desktop Netzwerkes hängt stark von der Kontrolle der eigenen Daten ab. In einem Desktop Netzwerk sollte die Speicherung und Verwaltung nutzerbezogener Informationen ausschließlich lokal erfolgen. Sie ermöglicht einem Nutzer die unmittelbare Kontrolle über das eigene Profil, beispielsweise über gespeicherte Anfragen und publizierte Dokumente.

  In unstrukturierten Netzwerken erfolgt die Speicherung und der Zugriff auf das Profils eines Peers ausschließlich lokal. Im Gegensatz dazu wird in strukturierten und hybriden Netzwerken [Nap03] das Profil des Nutzers an einem entfernten Peer registriert und abgefragt, der Nutzer gibt dadurch die Kontrolle über seiner Daten ab. Das führt einerseits zu einer technologischen (*Single Point of Failure*)und zu einer 'politischen' Abhängigkeit (*Auswertung der Nutzerdaten für Werbemails, Sperren einzelner Dateien und Nutzer*).

- **Geringe Managementkosten für den einzelnen Peer.** Durch die Installation einer kostenlosen Software und ohne Berücksichtigung besonderer administrativer oder finanzieller Vereinbarungen wird ein Peer spontan Teil einer existierenden, völlig dezentralen, selbstorganisierenden Infrastruktur. Die Kosten für die Publikation und das Löschen von Dokumenten und der Bekanntmachung der Identität des Peers sollen je nach Nutzung und Beteiligung über die Peers im Netzwerk verteilt werden. Insgesamt soll im INGA Desktop Netzwerk jeder Peer nur geringe Ressourcen zur Verwaltung von Dokumenten und des Netzwerkes beisteuern.

  Bedingt durch das konsistente Hashing in einem strukturierten Netzwerk verwaltet jeder Peer zu einem grossen Anteil Profile anderer Peers. Dieser Ansatz hat mehrere Nachteile: Die Informationen sind meist für den eigenen Peer von geringer Relevanz und nicht direkt nutzbar. Jeder Peer muss jedoch zusätzliche technische Ressourcen, wie Bandbreite und Speicherplatz, zur Verfügung stellen. Für eine gerechte Verteilung der Kosten werden Technologien zum Lastausgleich und für die Replikation eingesetzt. Für volatile

Netzwerke sind die dafür anfallenden Kosten hoch.[LHH+04] zeigen, das, durch den Aufwand für das häufige Publizieren und Löschen von Einträgen im Index für populäre Dokumente, strukturierte Peer-to-Peer Netzwerke eine geringere Effizienz aufweisen, als unstrukturierte Netzwerke mit auf Überflutung basierenden Ansätzen.

Unstrukturierte Netzwerke weisen deutlich geringere Kosten für die Verwaltung des Netzwerkes und der Verwaltung der auf einem Peer gespeicherten Dokumente auf. Typischerweise verwaltet jeder Peer nur sein eigenes Profil. Derartige Netzwerke nutzen die ungleiche Verteilung populärer Inhalte für die Optimierung der Suche aus und passen sich mit geringem Aufwand an eine dynamische Verfügbarkeit von Dokumenten und Peers an.

- **Effizientes Routing von Anfragen.** In Netzwerken mit mehreren 10.000 Teilnehmern und einer hohen Anzahl von Anfragen pro Peer sind die Kosten für eine Suchanfrage, respektive die Anzahl der Nachrichten, ausschlaggebend für die Effizienz des Routing Verfahrens. Um das INGA Desktop Netzwerk auch für eine grosse Anzahl von Nutzern zu ermöglichen, muss eine effiziente Routing Technologie eingesetzt werden.

  Durch die Verwendung von deterministischen Routing Algorithmen skalieren strukturierte Netzwerke gut für eine grosse Anzahl von Peers. Diese Netzwerke eigenen sich besonders für die Lokalisierung von Dateien mit einer exakten Übereinstimmung mit der Anfrage und von Dateien, die durch einen eindeutigen Schlüssel gekennzeichnet werden können.

  Suchstrategien in unstrukturierten Netzwerken basieren auf der unkontrollierten Überflutung des Netzwerkes mit Nachrichten. Sie sind einfach und robust, auch für hoch volatile Netzwerke, benötigen jedoch eine hohe Anzahl von Nachrichten [GMY02].

Gewünscht ist eine Infrastruktur, die über die Vorzüge eines unstrukturierten Netzwerkes, wie geringe Publikationskosten bei hoher Dynamik, Lokalität des Peer Profils, effizientes Bearbeiten komplexer Anfragen, verfügt, jedoch deren Nachteile, wie die hohe Anzahl von Nachrichten für Suchanfragen und den geringen Recall für unpopuläre Dokumente, behebt. Diese Anforderungen überschreiten die Leistungsfähigkeit aktueller Peer-to-Peer Technologien. Ideal wäre ein Ansatz, bei dem ein anfragender Peer mit hoher Wahrscheinlichkeit bereits 'seine Community', Peers mit potentiellen Antworten, kennt und eine Anfrage nur an diese Auswahl leiten muss.

Wir haben einen Ansatz entwickelt, der auf den Prinzipien der Suche in sozialen Netzwerken aufbaut. Ein Peer gewinnt über die Zeit Informationen, welche

anderen Peers besonders 'gut' Antworten beisteuern konnte und verfeinert dieses Wissen über sein 'persönliches Experten Netzwerk' durch ständige Interaktion weiter. Dadurch verlagert sich die Herausforderung der Suche in einem Peer-to-Peer Netzwerk zur Suche nach Personen, die entweder eine spezielle Anfrage beantworten können oder an eine Person weiterleiten können, die jemanden kennt, der unsere Anfrage beantworten kann. Die Suche dieser Personen hängt stark von unserem vorhandenen Expertennetzwerk ab: den uns bekannten Experten und deren Experten usw. In unserem Ansatz übertragen wir das Aufbauen von persönlichen Netzwerken über soziale Kontakte und das effiziente Finden von Experten auf die technische Netzwerkinfrastruktur eines Peer-to-Peer Systems.

4.2 Soziale Netzwerke und das Small World Phänomen

Das *Small World Phenomenon* ist ein 1967 geprägter soziologischer Begriff, der innerhalb der sozialen Vernetzung in der modernen Gesellschaft den hohen Grad abkürzender Wege (*Shortcuts*) durch persönliche Beziehungen bezeichnet. Es bezieht sich auf eine Prognose, nach der jeder Mensch auf der Welt mit jedem anderen über eine überraschend kurze Kette von Bekanntschaftsbeziehungen verbunden ist. Dies ist erstaunlicherweise möglich, obwohl die 'Dichte' des sozialen Netzwerks aller Akteure - gemessen als das Verhältnis der realen zu den rechnerisch möglichen Kontakten der Kontaktpersonen eines jedweden Akteurs - extrem gering ist, nämlich nahe 0. Der Begriff geht auf den US-amerikanischen Sozialpsychologen Stanley Milgram [Mil67] zurück, der in den 1960er Jahren experimentell feststellte, dass beliebige Menschen durch eine Kette aus nur wenigen miteinander bekannten Personen verbunden sind. Er gab Versuchspersonen einen Brief an eine ihnen völlig unbekannte Zielperson, den sie an einen Bekannten schicken sollten, von dem sie glaubten, dass er dem Adressaten näherstehen würde. Dieser sollte dann ebenso verfahren, bis der Brief schliesslich sein Ziel erreichte. Milgram ermittelte in seinen Experimenten, das solche Ketten durchschnittlich aus 6 Personen (*six degrees of separation*) bestehen. Aufbauend auf seinen Versuchen charakterisierte er Small World Netzwerke durch zwei Eigenschaften:

1. Es existieren kurze Pfade (Shortcuts).

2. Die beteiligten Personen sind erfolgreich im Routen entlang kurzer Pfade.

Bekannte Small World Netzwerke sind zum Beispiel das amerikanische Stromnetz, nahezu alle Teilmengen von sozialen Netzwerken, eine Submenge der Seiten des WWW, Zitate und Verweise in wissenschaftlichen Artikel oder auch die Router des Internet.

4.2.1 Charakteristiken von Small World Netzwerken

Die mathematisierte Netzwerkforschung hat im Zuge der Beschäftigung mit Small World Netzwerken eine Pluralität von Strukturmustern festgestellt und dabei ihr besonderes Augenmerk auf sogenannte *skalenfreie* Netze gelegt. Dabei handelt es sich um Netzwerke, bei denen einige wenige Knoten (sogenannte hubs) potentiell unendlich viele Verbindungen aufweisen, während ein Großteil der übrigen Knoten relativ wenige Beziehungen zu anderen Knoten hat. Um den Begriff des Small World Netzwerkes genauer zu definieren, verwenden wir die, von [Wat03, Kle00b] beschriebenen, folgenden fünf Charakteristiken:

1. **Kleiner Netzwerkdurchmesser.** Der Abstand, gemessen in Hops, zwischen zwei zufällig gewählten Knoten in einem Graphen ist gering.

   Definition 4.2.1 (Durchschnittliche kürzeste Anzahl von Hops). *Sei $\mathcal{G} = (V, E)$ ein Graph, bestehend aus Knoten V und Kanten E, mit denen die Knoten verbunden sind. $D_{min}(i, j)$ repräsentiert die kürzeste Distanz zwischen zwei Knoten i und j mit $i, j \in V$. Weiterhin bezeichne N die Anzahl der Knoten in $\mathcal{G}$ mit $|V| = N$. Der durchschnittliche Netzwerkdurchmesser in $\mathcal{G}$, bezeichnet als $Dia(\mathcal{G})$ wird definiert als:*

   $$Dia(\mathcal{G}) = \frac{1}{\begin{pmatrix} N \\ 2 \end{pmatrix}} \sum_{i \neq j} D_{min}(i, j)$$

   $Dia(\mathcal{G})$ ist das Verhältnis der Summe aller kürzesten Pfade zwischen beliebigen zwei Knoten in $\mathcal{G}$ und allen anderen möglichen paarweisen Verbindungen in $\mathcal{G}$. ♣

2. **Dünne Vernetzung.** Im Vergleich mit einem vollständig verbundenen Graphen, in dem jeder Knoten mit jedem anderen Knoten über eine Kante verbunden ist, verfügt die Mehrzahl der Peers in Small World Netzwerken nur über eine geringe Anzahl von Verbindungen zu anderen Knoten.

3. **Power-Law Verteilung.** Small World Netzwerke weisen typische Eigenschaften eines *skalenfreien* Netzwerkes auf, die über keine typische Anzahl von Verbindungen pro Knoten verfügen. Vielmehr ist in Small World Netzwerken die Anzahl der Verbindungen Power-Law verteilt. Dabei handelt es sich um Netzwerke, bei denen einige wenige Knoten (sog. hubs) eine besonders hohe Anzahl von Verbindungen aufweisen, während ein Grossteil der übrigen Knoten relativ wenige Beziehungen zu anderen Knoten hat.

4. **Entstehen von Clustern.** In Small World Netzwerken ist die Wahrscheinlichkeit hoch, dass zwei Knoten, die jeweils eine Kante zu einem dritten Knoten haben, auch untereinander verbunden sind. In sozialen Netzwerken sind die Freunde einer Person meistens auch untereinander bekannt, weil sie sich über den gemeinsamen Freund kennengelernt haben (Transitivitätsprinzip). Mathematisch wird diese Tatsache über den Clustering-Koeffizienten beschrieben, der für Small World-Netzwerke überdurchschnittlich hoch ist.

   Definition 4.2.2 (Clustering Koeffizient). *Sei D die Distanz zweier Knoten $u, v \in V$. Die Nachbarschaft eines Knoten v ist gegeben als eine Menge von Knoten $\Gamma_v = \{u : D(u, v) = 1\}$. Sei k_v die maximale Anzahl der Links zu benachbarten Knoten für einen Knoten $v \in V$ und sei C_v der lokale Clustering Koeffizient von v mit*

$$C_v = \frac{|E(\Gamma_v)|}{k_v * (k_v - 1)}$$

   wobei $E(\Gamma_v)$ einen Operator darstellt, der die Anzahl der Links in der Menge von Γ_v zählt. Der Clustering Koeffizient $\mathcal{C}(\mathcal{G})$ eines Graphen $\mathcal{G}$ wird definiert als

$$\mathcal{C}(\mathcal{G}) = \frac{1}{N} \sum_{v \in V} C_v$$

   Anders ausgedrückt, $\mathcal{C}(\mathcal{G})$ misst die 'Kompaktheit' des Graphen.

5. **Hohe Fehlertoleranz.** Die spezielle Vernetzung eines skalenfreien Netzes macht ein solches Netzwerk robust gegen den Ausfall einiger zufälliger Knoten oder Kanten. Falls aber stark vernetzte Knoten, sogenannte *hubs*, gezielt entfernt werden, zerfällt das Netzwerk schnell in Teilnetze. Dies ist unter anderem der Grund, warum der Ausfall nur weniger Router im Internet weitreichende Auswirkungen haben kann.

4.2.2 Definition von Shortcuts auf Basis sozialer Metaphern

Motiviert durch die Unzulänglichkeiten strukturierter und unstrukturierter Peer-to-Peer Ansätze und inspiriert durch die Eigenschaften von Small World Netzwerken haben wir einen neuen Ansatz für das Routing in einem unstrukturierten Netzwerk entwickelt. Als Analogie zu sozialen Netzwerken setzen wir dazu einen Peer mit einer Person in einem sozialen Netzwerk gleich. Unser Ansatz basiert vollständig auf lokalem Wissen, das ein Peer über bereits gestellte Fragen und erzielte Antworten gewonnen hat. Jeder Peer spezialisiert sich auf einen bestimmten Teilbereich

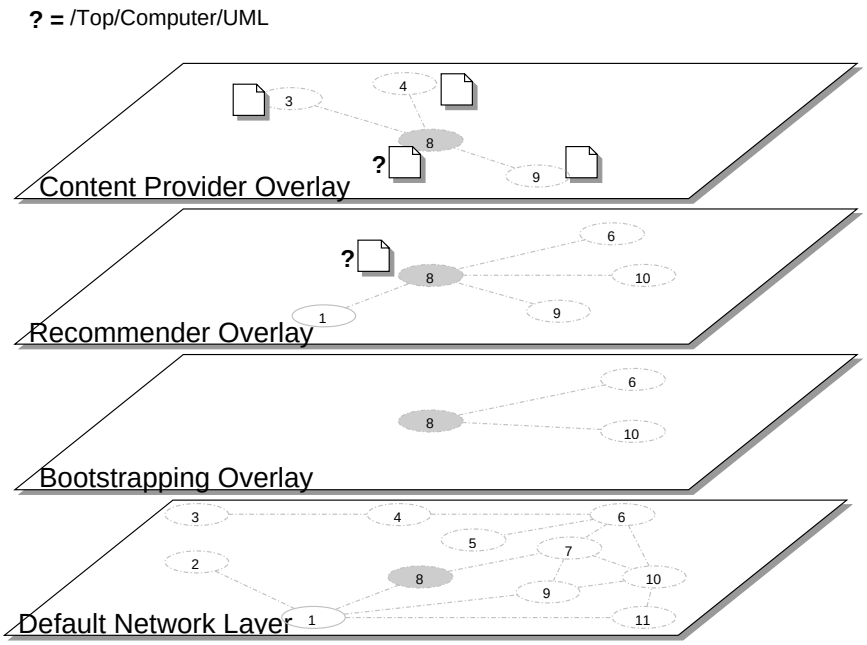


Abbildung 4.1: Visualisierung der Indizes an einem Beispiel

von Interessen. Ein Peer speichert nur Wissen über andere Peers, die ihn selbst interessieren. Dieses Wissen wird in Form von Shortcuts lokal gespeichert.

Zur Erstellung der Shortcuts verwenden wir eine Reihe von sozialen Metaphern. Aufgrund von Beobachtungen unterscheiden wir vier Typen von Personen in unserem Netzwerk, an die wir Anfragen richten. In dieser Arbeit verstehen wir unter eine Anfrage einen Term, der in einer Themen-Hierarchie definiert wurde (siehe Definition 4.2.3). Ein Beispiel ist die Bildung einer Anfrage nach Dokumenten zur Beschreibungssprache UML aus dem Term $/Top/Computer/UML$. Die folgenden Metaphern können jedoch auch für Anfragen, die aus mehreren Termen bestehen, angewandt werden. Wir definieren die folgenden Metaphern:

1. Eine Anfrage wird an eine Person gestellt, die diese Anfrage in der Vergangenheit bereits korrekt beantwortet hat. Wir nennen eine solche Person einen *Content Provider* und organisieren die korrespondierenden Peers im *Content Provider Layer*.

2. Kennen wir keine Person, die eine ähnliche Anfrage in der Vergangenheit beantwortet hat, suchen wir nach Personen, die eine ähnliche Anfrage in der Vergangenheit gestellt hat. Wir nehmen an, das diese Person mit hoher Wahrscheinlichkeit bereits einen Content Provider kennt, der unsere Anfrage

beantworten kann. Wir nennen derartige Personen *Recommender* und organisieren die korrespondierenden Peers im *Recommender Layer*.

3. Wenn wir keine Person zur Beantwortung einer Anfrage bereits kennen, schicken wir die Anfrage zu einer Person, die bereits ein 'breites' soziales Netzwerk zu anderen Personen entwickelt hat. Solche Personen formen unser initiales soziales Netzwerk. Die korrespondierenden Peers bilden den *Bootstrapping Layer*.

4. Kennen wir immer noch keine Person zur Beantwortung der Anfrage leiten wir die Anfrage an zufällig ausgewählte Personen weiter. Um auch neue Personen für bereits bekannte Anfragen kennenzulernen, wenden wir diese Form der Weiterleitung zusätzlich in zufälligen Abständen an. Wir nennen derartige Personen unser *Default Network* und leiten eine Anfrage an, die durch die initiale Netzwerk Struktur vorgegebenen Nachbarn weiter.

Die verschiedenen Layer werden durch verschiedene, unabhängige Indizes an einem Peer lokal repräsentiert. Grafik 4.1 visualisiert den Inhalt der Indizes am Beispiel von Peer 8. Der Peer verfügt über zwei Nachbarn im Default Overlay und Bootstrapping Overlay, die über keine gemeinsame Schnittmenge verfügen. Der Index von Peer 8 hat drei Content Provider und vier Recommender Peers identifiziert und in seinem Index lokal gespeichert.

Peers lernen durch ihre Interaktionen andere Peers im Netzwerk kennen. Auf der Basis der oben geschilderten Annahmen erstellen Peers lokale Shortcuts zu anderen Peers. Die Shortcuts aller Peers bilden ein adaptives Overlay Netzwerk, in dem das Routing erfolgt. Für eine Anfrage wählt ein Peer auf der Basis seines lokalen Wissens die besten Peers für die Weiterleitung der Anfrage aus. Wir wählen entweder Peers aus, die eine Anfrage beantworten können, mit hoher Wahrscheinlichkeit für die Anfrage passende Peers kennen, bzw. Peers, die gut vernetzt sind. Dazu berücksichtigen wir die in den sozialen Metaphern implizit vorgegebene Reihenfolge. Wir versuchen Peers auszuwählen, die eine exakte Antwort auf die Anfrage bereits gegeben haben und wählen dann Peers aus, die ähnliche Anfragen bereits gestellt haben, bzw besonders gut vernetzt sind oder, letztendlich, durch Zufall mit unserem Peer benachbart sind.

Im Gegensatz zu den im vergangenen Kapitel beschriebenen Ansätzen benötigen wir keine Registrierung des Peer-Profile in einem zentralen oder verteilten Index. Jeder Peer verfügt über die vollständige Kontrolle der lokalen Daten und der Weiterleitung von Anfragen. Ein weiterer Vorteil eines Shortcut Netzwerkes ist die 'egoistische' Verwaltung der technischen Ressourcen eines Peers. Ein Peer stellt nur Speicherplatz und Bandbreite zur Verwaltung von Shortcuts zu entfernten Peers und für das Weiterleiten von Anfragen zur Verfügung, die mit seinen eigenen

Interessen übereinstimmen. Schliesslich nutzen wir für das effiziente Routing der Anfragen die Small World Charakteristiken des Shortcut-Overlay aus.

4.2.3 Formale Definition von Shortcut Netzwerken

In diesem Abschnitt führen wir das Konzept der adaptiven Overlay Netzwerke auf der Basis von Shortcuts formal ein. Wir modellieren unser System als eine Menge von N Knoten in dem jeder $n_i \in N$ eine Menge von D_i Dokumenten verwaltet. Jeder Knoten repräsentiert einen Peer. Dabei kann ein Dokument an mehr als einem Knoten vorhanden sein. Wir bezeichnen die Menge aller Dokumente als $\mathcal{D}$.

Definition 4.2.3 (Themenhierarchie). *Sei $\mathcal{T}$ eine Themen-Hierarchie. Jedes semantische Konzept in ihr entspricht einem Thema $t \in \mathcal{T}$, das durch einen Path t_{path} eindeutig definiert wird.* ♣

Definition 4.2.4 (Dokument). *Jedes Dokument $d(t_1, ..., t_i) \in \mathcal{D}$ ist mindestens einem Thema $t_1, ..., t_i \in \mathcal{T}$ zugeordnet.* ♣

Anfragen nach Dokumenten erfolgen durch das Stellen einer Anfrage $q(t)$.

Definition 4.2.5 (Anfrage). *Eine Anfrage $q(t)$ enthält genau ein Thema t, mit $t \in \mathcal{T}$.* ♣

Wir modellieren eine Übereinstimmung zwischen einer Anfrage $q(t)$ und einem Dokument $d(t_1, ..., t_i)$ als eine binäre Funktion $M(q(t), d(t_1...t_i))$, deren Resultat 1 bei einer Übereinstimmung und sonst 0 ist.

Definition 4.2.6 (Übereinstimmung zwischen Dokument und Anfrage).

$$M(q(t), d(t_1...t_i)) = \begin{cases} \textit{1, wenn } t \subseteq (t1, .., t_i) \mid t, t_1, ..., t_i \in \mathcal{T} \\ \textit{0, sonst} \end{cases}$$

. ♣

Eine Anfrage kann an einem Peer entweder mit keinem, mit einem oder mit mehreren passende Dokumenten übereinstimmen. Die Anzahl der Übereinstimmungen an einem Peer für eine Anfrage bezeichnen wir als *Query Hits*.

Definition 4.2.7 (Query Hits und erfolgreiche Anfrage). *Wir definieren die Anzahl der Treffer (*hits*) an einem Knoten n_i für eine Anfrage q als*

$$\mathcal{H}(q(t), n_i) = \sum_{d \in D_i} M(q(t), d_i)$$

und bezeichnen alle Anfragen, die mit mindestens einem Dokument an einem Peer übereinstimmen, als erfolgreiche Anfrage. ♣

Unstrukturierte Peer-to-Peer Netzwerke, wie Gnutella, erzeugen ein einzelnes Overlay Netzwerk. In diesen ist ein Peer mit einigen anderen Peers, seinen Nachbarn, über zufällig gewählte Shortcuts verbunden, die wiederum über Shortcuts mit anderen Nachbarn verbunden sind.

Definition 4.2.8 (Shortcut). *Wir bezeichnen einen Shortcut als einen Triple* $(n_i, n_j, l, type)$ *in dem* n_i *und* n_j *unidirectional miteinander verbundene Peers,* l *eine Zeichenkette (String) und* $type$ *den Typ des Overlays repräsentiert.* ♣

Jeder Peer verfügt neben den zufälligen gewählten Shortcuts des Default Overlay Netzwerkes über weitere, entsprechend den sozialen Metaphern ausgewählte, Shortcuts für drei weitere Typen von Overlay Netzwerken: Content Provider, Recommender und Bootstrapping Overlay. Für jeden Typ definieren wir folgende Semantiken für l und $type$:

Definition 4.2.9 (Content, Recommender, Bootstrapping und Default Overlay). *Wir definieren eine Menge von Peers* $n_a, ..., n_z$*, die Dokumente für ein Thema* t *publizieren und die dem Knoten* n_i *bekannt sind, als einen Content Provider Overlay für das Thema* t *mit* $l = t$ *sowie* $type = co$ *und schreiben* $\mathcal{CO}^t_{n_i}$*.*
Beispiel:

$$\mathcal{CO}_8^{/Top/Computer/UML} = \{3, 4, 9\}$$

Wir definieren eine Menge von Knoten $n_a, ..., n_z$*, die Anfragen für das Thema* t *gestellt haben und die dem Knoten* n_i *bekannt sind, als einen Recommender Overlay für das Thema* t *mit* $l = t$ *sowie* $type = ro$ *und schreiben* $\mathcal{RO}^t_{n_i}$*.*
Beispiel:

$$\mathcal{RO}_8^{/Top/Computer/UML} = \{1, 6, 9, 10\}$$

Wir definieren eine Menge von Knoten $n_a, ..., n_z$*, die besonders gut vernetzt und dem Knoten* n_i *bekannt sind, als einen Bootstrapping Overlay mit* $l = null$ *sowie* $type = bo$ *und schreiben* $\mathcal{BO}_{n_i}$*.*
Beispiel:

$$\mathcal{BO}_8 = \{1, 6, 10\}$$

Wir definieren eine Menge von Knoten $n_a, ..., n_z$*, die mit* n_i *in der initialen Netzwerk Struktur miteinander verbunden sind, als Default Network Overlay mit* $l = null$ *sowie* $type = do$ *und schreiben* $\mathcal{DO}_{n_i}$*.*
Beispiel:

$$\mathcal{DO}_8 = \{1, 7\}$$

Grafik 4.1 zeigt die verschiedenen Overlays am Beispiel für Peer acht und dem Thema */Top/Computer/UML*.

Definition 4.2.10 (Adaptives Overlay Netzwerk auf der Basis von Shortcuts). *Alle Shortcuts mit demselben Label l bilden ein gemeinsames Adaptives Overlay Netzwerk ON_l. Gegeben zwei Peers n_i, n_j mit $n_i \neq n_j$, definieren wir ON_l als*

$$ON_l = \{(n_i, n_j) \in N \times N \wedge |\exists\, a\ Shortcut\ (n_i, n_j, l, type)\}$$

Jedes ON_l unterstützt drei Funktionen:

1. Create(n_j,l,type)*, bei dem ein Peer n_j einen Shortcut der Form $(n_i, n_j, l, type)|n_i, n_j \in ON_l$ zu einem entfernten Peer n_j erzeugt.*
2. Search(q(l)) *bei der durch eine Anfrage $q(l)$ eine Teilmenge von Content Provider Peers aus ON_l zurückgegeben wird, die $q(l)$ beantworten können.*
3. Delete(n_j,l,type) *in der ein Peer einen Shortcut zum Peer n_j für den Type $type$ und für das Netzwerk ON_l löscht .* ♣

Die Implementation dieser Funktionen wird in Kapitel 5 und 6 vorgestellt.

4.3 INGA Systemarchitektur

Die INGA Infrastruktur bietet typische Peer-to-Peer Funktionalität, wie das Publizieren und Suchen von Dokumenten, an. Sie unterscheidet sich jedoch von anderen Peer-to-Peer Systemen dadurch, dass Peers miteinander kooperieren und Informationen für das Weiterleiten von Anfragen austauschen. Jeder Peer speichert Shortcuts in Form von Tupeln *Themen* $\times$ *Peers*. Dabei speichert jeder Peer 'egoistisch' nur Shortcuts zu anderen Peers, die den Peer selbst interessieren. Um Antworten auf Anfragen zu erhalten, 'sozialisiert' sich der Peer mit anderen Peers. Er durchsucht deren Shortcut Index und gewinnt neue Shortcuts. Unser Ansatz verwendet das Peer-to-Peer Paradigma einerseits für den Austausch von Dokumenten und andererseits für den Austausch von Shortcut zwischen den Peers. Im Folgenden stellen wir die Komponenten unserer Infrastruktur vor und erläutern ihre Funktion und Aufgabe. Grafik 4.2 gibt einen Überblick über einen Peer in der INGA Infrastruktur.

4.3.1 Netzwerk Management

Als Basisinfrastruktur verwenden wir das JXTA Netzwerk [Gon01, TAA+03]. Das Open Source Project gehört zu den führenden Industrieplattformen für Peer-to-Peer Technologie. Es wurde ursprünglich von SUN Microsystems initiiert und wird von einer breiten Gemeinschaft akademischer und industrieller Nutzer weiterentwickelt

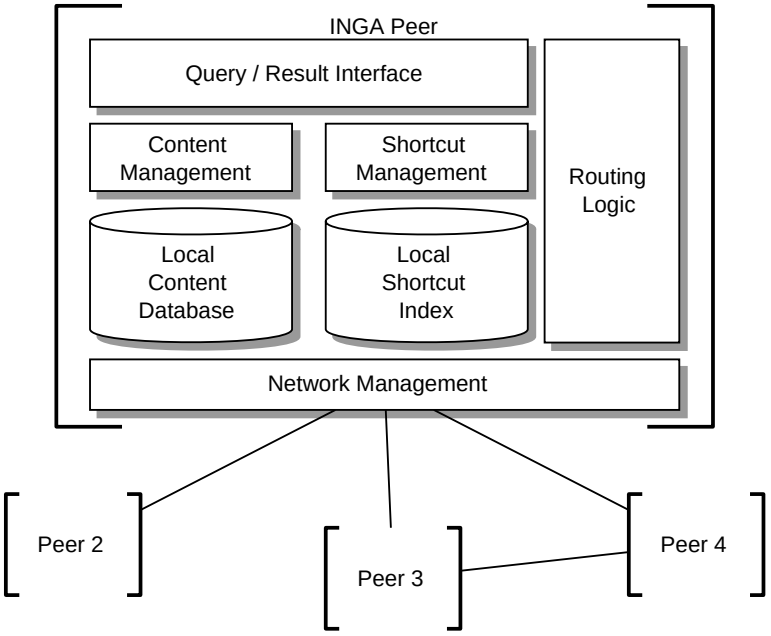


Abbildung 4.2: Architektur eines INGA Peers

und verbessert. Aktuelle JXTA Projekte [Tra04] erfolgen zum Beispiel im Bereich der Navigationsdaten der *US Coast Guard*, der Suche in verteilte Immobiliendaten *National Association of Realtors of America* und der Suche in verteilten Lernobjekt Repositories mit *Edutella* [NWS+03].

Unser Ansatz ist auch auf andere Infrastrukturen übertragbar, sofern diese mindestens die folgenden zwei Funktionen zur Verfügung stellen:

- **Austausch von Nachrichten mit anderen Peers.** Nachrichten sind die Basis-Einheit in der Peers Daten austauschen. Wir nehmen an, das jeder Peer eine konkrete Implementierung dazu aufweist. Das JXTA Netzwerk erlaubt den Austausch von Nachrichten mit anderen Peers unabhängig von ihrer Position im Netzwerk (Firewall, NAT). Dazu werden entweder das TCP/IP oder das HTTP Protokoll benutzt.

- **Eindeutige Identifikation eines Peers.** Jeder Peer im Netzwerk muss über einen eindeutigen Schlüssel identifiziert werden. Im einfachsten Fall ist das die IP Adresse des Peers. Weitere Verfahren, beispielsweise für das JXTA Netzwerk und Gnutella, werden in Abschnitt 2.1 beschrieben.

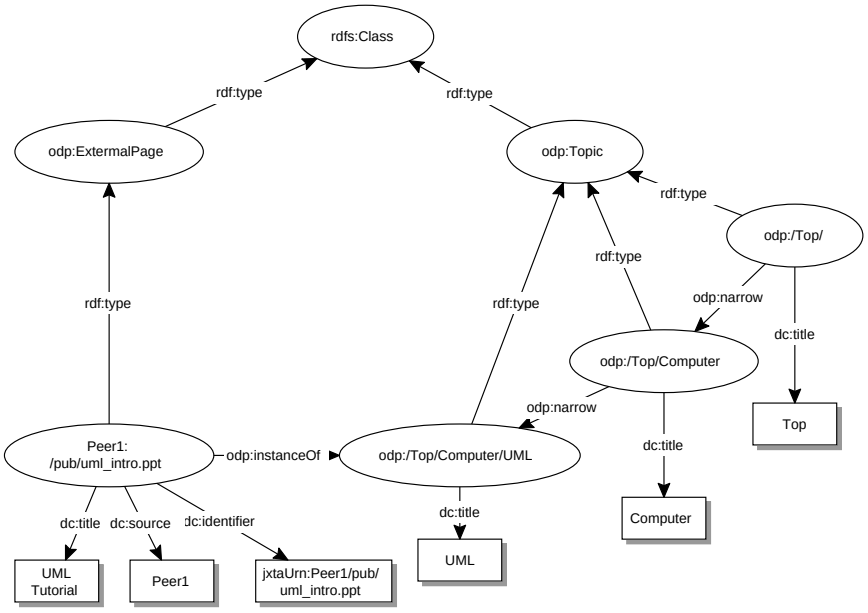


Abbildung 4.3: INGA Datenmodel und Beispiel

4.3.2 Lokales Dokumenten Repository

Analog zu File Sharing Netzwerken kann jeder Peer ausgewählte Dateien anderen Peers zur Verfügung stellen. Dateien, die bereits von einem anderen Peer geladen wurden, können ebenfalls publiziert werden. Wir beschreiben zuerst das Datenmodell und später die Aufgaben der Komponente:

Datenmodell zur Publikation von Dateien. Informationen zu Dateien werden syntaktisch in Form von RDF Metadaten dargestellt. In diesem und den folgenden Kapiteln nehmen wir an, dass jede Datei mindestens in einer Themen-Hierarchie klassifiziert ist und mit Metadaten annotiert ist. Semantisch werden die Metadaten durch das existierende RDF(S) Datenmodell [BG03, LS99], den Dublin Core Standard [Ini04] und das Datenmodel des Open Directory beschrieben [BG03]. Im Beispiel in Grafik 4.3 publiziert ein Peer eine Datei zum Thema *UML Tutorials*. Jede Ressource von Typ *odp:ExternalPage*, also eine lokale Datei an einem Peer, stellt eine konkrete Instanz für eine Ressource von Typ *odp:topic*, also ein Thema aus der ODP Themen-Hierarchie, dar. Einzelne Themen stehen semantisch untereinander in einer Spezialisierungsbeziehung, die durch die Relation *odp:narrow* ausgedrückt wird.

Publizieren von Dateien und Metadaten. Jeder Peer hat die Möglichkeit, Infor-

mationen über Dateien zu publizieren, die er anderen Peers zur Verfügung stellen möchte. Dazu veröffentlicht der Peer lokal Metadaten über seine vorhandenen Dateien. In dieser Arbeit wurde die Implementierung auf der Basis des *Sesame RDF Repositories* [BKvH02] durchgeführt.

Anfragen formulieren und Resultate zurückgeben. Jeder Peer verfügt über eine Schnittstelle zur Formulierung von Anfragen. Zur technischen Realisierung wurde in der Arbeit die im Sesame verfügbare SeRQL Sprache [BKvH02] für die Formulierung von Anfragen verwendet.

Beispiel 4.3.1 (SeRQL Anfrage). Die folgende Anfrage sucht nach eindeutigen Schlüsseln für alle Dateien mit dem Thema *UML*:

```
SELECT {dc:identifier}
FROM {odp:externalPage}
     <odp:instanceOf>  {<odp:/Top/Computer/UML>}
using namespace
odp = <!http://dmoz.org/rdf#>,
dc = <!http://purl.org/dc/elements/1.0#>
```

*

4.3.3 Management und Auswahl von Shortcuts

Im Gegensatz zu aktuellen Filesharing Netzwerken, wie Gnutella oder Kazaa, sammelt jeder INGA Peer 'Wissen' über die publizierten Dateien anderer Peers im Netzwerk. Dieses Wissen speichert der Peer in einem lokalem *Shortcut Index*. Auf dessen Basis trifft der Peer die Entscheidung, an welche anderen Peer eine Anfrage weitergeleitet werden soll. Die Verwaltung des Index erfolgt in der *Shortcut Management Component*. Ihre Aufgaben sind:

Erstellen und Löschen von Shortcuts. Aus erfolgreichen Anfragen werden Informationen über den antwortenden Peer und Peers die die Anfrage weitergeleitet haben, im Shortcut Index gespeichert. Kann der Index keine weiteren Shortcuts mehr aufnehmen, werden durch eine *Index-Policy* Einträge aus dem Index entfernt, bzw. ausgetauscht. In Kapitel 5 und 6 diskutieren wir Verfahren zur Erstellung und zum Löschen von Shortcuts.

Auswahl der besten Peers für eine Anfrage. Für eine Anfrage, entweder eines entfernten oder des lokalen Peers, werden lokal die besten Peers ausgewählt, an die eine Anfrage weitergeleitet wird. Die Auswahl der Peers basiert auf dem Wissen über andere Peers, deren Performance in der Vergangenheit und der Ähnlichkeit des Themas der Anfrage zu bereits gespeicherten Shortcuts.

In Abschnitt 6.2 stellen wir verschiedene Strategien für die Auswahl der besten Peers auf der Basis lokalen Wissens vor.

4.3.4 Format der Anfrage- und Resultatnachrichten

Peers tauschen Nachrichten in Form von Anfragenachrichten (*Query Messages*) und Resultatnachrichten (*Result Messages*) miteinander aus. Um schneller Informationen über andere Peers zu sammeln, werden bestimmte zusätzliche Informationen in den Nachrichten von den Peers beim Empfang ausgewertet. In diesem Abschnitt beschreiben wir das Format der Anfrage- und Resultat Nachrichten.

Anfrage und Resultatnachrichten beinhalten immer den transitiven Pfad, den die Anfrage, beginnend vom anfragenden Peer, bereits zurückgelegt hat. Wir bezeichnen ihn als Message Pfad und benutzen ihn, um Informationen zum Aufbau von Shortcuts zu gewinnen.

Definition 4.3.1 (Message Path). *Sei P_{query} ein anfragender Peer und $P_a, ..., P_z$ eine Liste von Peers, die den transitiven Weg der Anfrage beschreiben. Wir definieren den Message Path als Liste MP mit*

$$MP = (P_{query}, P_a, ..., P_z)$$

Kann ein Peer P_{answer} auf eine Anfrage antworten definieren wir den Message Path als Liste MP_{answer} mit

$$MP_{answer} = (P_{query}, P_a, ..., P_z, P_{answer})$$

. ♣

Jede Anfrage eines Peers wird in Form einer *Query Message* an andere Peers weitergeleitet.

Definition 4.3.2 (Query Message). *Sei*

- $q(t) — t \in \mathcal{T}$ *die Anfrage*
- QID *ein für die Anfrage eindeutiger Schlüssel*
- MP *der transitive Weg der Anfrage*
- BC *ein ganzzahliger positiver Wert, der die* Bootstrapping Capability *von P_{query} beschreibt (siehe Definition 5.3.3)*
- maxHops *die maximale Länge Anzahl der Hops, die die Anfrage zurücklegen darf*

dann definieren wir eine Query Message als ein Quintuple

$$M_{Query} = (q(t), QID, MP, BC, maxHops)$$

. ♣

Anmerkung: Das Modell einer Anfrage eines INGA Peers ist angelehnt an das Modell von Gnutella [Kan99]. Analog zu Gnutella und zur Vermeidung des Sendens der Nachricht an Peers, die eine Anfrage bereits erhalten haben, beinhaltet die Nachricht die PeerIDs aller Peers, die die Nachricht bereits erhalten haben. Sie sind im *Message Path*, beginnend mit dem anfragenden Peer, geordnet. Weiterhin enthält jede Anfrage einen eindeutigen Schlüssel, die QID. Wir vermeiden damit, dass ein Peer mehrmals auf die gleiche Anfrage antwortet und damit die Anzahl der Nachrichten im Netzwerk erhöht. Eindeutige Schlüssel für Anfragen werden in JXTA durch einen speziellen Zufallsgenerator erzeugt. Die Variable BC beinhaltet Informationen über den Grad der Vernetzung des anfragenden Peers, deren Berechnung in Definition 5.3.3 vorgestellt wird. Peers die eine Anfrage erhalten, lernen dadurch schnell andere Peers kennen, die eine hohen Grad der Vernetzung aufweisen.

Das Resultat auf eine erfolgreiche Anfrage wird in einer *Result Message* übertragen.

Definition 4.3.3 (Result Message). *Sei*

- $q(t)$ *mit* $t \in \mathcal{T}$ *die Anfrage*
- MP_{answer} *der transitive Weg der Anfrage, inklusive dem antwortendem Peer*
- $\mathcal{H}(q(t), P_{answer})$ *die Query Hits der Dokumente an dem antwortenden Peer die zu der Anfrage passen*

dann definieren wir eine Result Message als ein Triple

$$M_{Result} = (q(t), MP_{answer}, \mathcal{H}(q, P_{answer}))$$

. ♣

Anmerkung: Im Gegensatz zum Gnutella-Protokoll sendet ein Peer eine Antwort direkt an den anfragenden Peer zurück. Dessen PID wird aus dem *Message Path* ermittelt .

4.4 Verwandte Arbeiten

Die Auswahl von Experten für eine Anfrage innerhalb einer Organisation und die Suche nach dem passenden Peer für eine Anfrage sind eng verwandte Fragestellungen. Wir unterscheiden verwandte Arbeiten innerhalb der Suche in Peer-to-Peer Netzwerken, auf dem Gebiet der verteilten Desktop Suche und Arbeiten zur Lokalisierung von Experten.

Der *Gnutella* Ansatz [Gnu04] leitet eine Anfrage ungerichtet an eine grosse Anzahl von Peers. Typischerweise kann ein Grossteil der erreichten Peers die Anfrage nicht beantworten, wodurch das Netzwerk mit vielen unnötigen Nachrichten belastet wird. Zur Vermeidung einer Überflutung des Netzwerkes mit Nachrichten wurden die Systeme *Edutella* in Abschnitt 3.1 und *TOPICS* in Abschnitt 3.2 vorgestellt. Diese Systeme verwenden einen globalen, verteilten Index auf der Basis eines *strukturierten Peer-to-Peer Overlays*. Durch die technisch bedingte Verteilung der Einträge im Index entsteht jedoch zusätzlicher Aufwand für die Lastverteilung, Replikation und Aktualisierung des Index. Ähnlich Gnutella basiert der INGA Ansatz auf einem *unstrukturierten Peer-to-Peer Overlay*. Im Gegensatz zu beiden Ansätzen verfügt er über einen adaptiven Overlay auf der Basis der Interessen der Peers und kombiniert geringe Verwaltungskosten eines unstrukturierten Netzwerkes mit der gerichteten Suche in einem Overlay Netzwerk. Peers, die durch ihre Interessen miteinander in Beziehung stehen, werden durch Shortcuts dynamisch miteinander verbunden. Eine Anfrage wird entlang des Shortcut Overlays an die lokal bekannten besten Peers geleitet. Ein ähnlicher Ansatz wird in [CGM02b] vorgestellt, Die Autoren verwenden ein semantisches Overlay Netzwerk, in dem Dokumente Kategorien zugeordnet werden. Es bleibt jedoch offen, wie der Index gebildet wird und wie die Peers den Index verwalten.

Das Problem der Desktop Suche wird häufig in Zusammenhang mit einer Infrastruktur für die kollaborative Zusammenarbeit in Gruppen betrachtet. [DF04] gibt mit dem *Networked Semantic Desktop* eine Vision des vernetzten Desktops wieder. Eine Implementation der lokalen, jedoch nicht verteilten, Desktops Suche, erfolgte in *Gnowsis* [Sau03]. *YouServ* [AGS02] ermöglichen eine einfache und kostengünstige Publikation lokal gespeicherter Dokumente über einen zentralen Index. *Groove Networks* erlaubt eine kollaborative Arbeitsumgebung und den Austausch von Dokumenten für kleine Arbeitsgruppen durch einen vollständig replizierten Index.

MINDS und *ReferralWeb* sind zwei grundlegende Ansätze für *Referral Systems*. In diesen Systemen leiten Agtenten eingehende Anfragen an Nutzer mit den passenden Profil weiter. Ähnlich den Shortcuts verfügt jeder Agent über ein eigenes Netz von Referrals, über das die Anfragen weitergeleitet werden. *MINDS* [HMSB87] stellt Heuristiken für das Erlernen und die Erstellung von Referrals

vor, während *ReferralWeb* [KSS97] sich auf das Finden geeigneter Startknoten im Netzwerk konzentriert. Beide Systeme verwenden als Basis die sozialen Netzwerke der zu den Agenten korrespondierenden Nutzer. Im Gegensatz zu unserem Ansatz verwenden die Systeme nur das Prinzip der erfolgreichen Antworten, berücksichtigen auch keine Anfragen oder eine besonders gute Vernetzung der einzelnen Nutzer. Ein weiterer wichtiger Unterschied liegt in dem zentralen Index, der die Publikation der Profile der einzelnen Nutzer und deren regelmässige Aktualisierung erfordert.

4.5 Zusammenfassung

In diesem Kapitel wurde die Idee adaptiver Overlay Netzwerke, ihre Terminologie und die Systemarchitektur vorgestellt. Peers erstellen und verwalten Shortcuts zu anderen Peers in lokalen Indizes auf der Basis sozialer Metaphern und leiten Anfragen entlang des Shortcut Netzwerkes weiter. Unser Ansatz erweitert die grundlegende Architektur eines Peers in einem unstrukturierten Netzwerk um weitere Komponenten, wie das Management von Shortcuts und die Auswahl von Shortcuts für eine Anfrage.

Der Ansatz unterscheidet sich von strukturierten Netzwerken durch den fehlenden globalen Index und erlaubt eine vollständig lokale Kontrolle des Profils eines Peers. Weitere Kosten, zum Beispiel für den Lastausgleich und die Pflege eines globalen Index fallen nicht an. Die Erstellung des Overlay Netzwerkes erfolgt implizit durch Beobachtung der Aktionen eines Nutzers und erfordert keine weitere manuelle Pflege durch den Nutzer.

5 Erzeugung von Shortcuts

Nachdem im letzten Kapitel die grundlegende Idee des INGA Ansatzes vorgestellt wurde, beschreiben wir in diesem Kapitel die Erstellung von Shortcuts. Zur Einführung klassifizieren wir die von uns identifizierten Typen von Shortcuts. Danach stellen wir neu entwickelten Strategien und Technologien für das schnelle Erlernen von *anfrageabhängigen* und *anfrageunabhängigen* Shortcuts vor.

5.1 Merkmale von Shortcuts

In diesem Abschnitt definieren wir Merkmale zur Klassifikation von Shortcuts. Anhand dieser Merkmale identifizieren wir sechs Klassen von Shortcuts, die zusätzliche Kanten zwischen ausgewählten Peers erzeugen und dadurch unterschiedliche Overlays über der initialen Netzwerk Struktur bilden. Folgende Eigenschaften wurden von uns in die Klassifikation (siehe Grafik 5.1) übertragen:

Bi/Uni-direktionale Shortcuts. *Bi-direktionale* Shortcuts erfordern neben einer initialen Bestätigung auch eine laufende Aktualisierung durch beide Peers. Dieser Vorgang basiert nicht mehr nur auf dem alleinigen lokalen Wissen eines Peers, sondern erfordert den Austausch von Informationen zwischen den beteiligten Peers. Dadurch entstehen zusätzliche Nachrichten im Netzwerk. *Uni-direktionale* Verbindungen werden in der technologischen Basis-Infrastruktur dieser Arbeit, dem JXTA Netzwerk und in Gnutella oder dem HTTP Protokoll, angewendet. Shortcuts, die auf der Basis dieses einfaches

Modells erzeugt werden, ermöglichen einem Peer eine Entscheidung nur auf der Basis seines eigenen lokalen Wissens, ob ein Shortcut erstellt oder gelöscht wird. Aufgrund der Einfachheit des Modells und seiner (Entscheidungs)-Autonomie unabhängig von anderen Peers im Netzwerk betrachten wir in dieser Arbeit nur uni-direktionale Shortcuts.

In/Out-Bound Shortcuts. Peers, die dem Netzwerk neu beigetreten sind, kennen noch keine anderen Peers im Netzwerk. Eine Kernaufgabe des Shortcut Managements ist die implizite Erstellung möglichst vieler und relevanter Shortcuts innerhalb kurzer Zeit. Dazu berücksichtigen wir sowohl Strategien für die Erstellung *Out-Bound uni-direktionaler Shortcuts* und *In-Bound uni-direktionaler Shortcuts*.

Auf der Basis der in Abschnitt 4.2.2 definierten sozialen Metaphern verwenden wir Outbound uni-direktionale Verbindungen zur Erstellung von Shortcuts, die durch den Sender einer Anfrage ohne Bestätigung durch den Empfänger erstellt werden. Beispiele dafür sind Shortcuts, die auf der Basis gefundener Antworten zu einer Anfrage ermittelt werden (Content Provider Shortcuts), Shortcuts, die auf der Basis der durch den eigenen Peer gestellten und von einem anderem Peer erfolgreich weitergeleiteten Anfrage erzeugt werden (aktive Recommender Shortcuts) und Shortcuts, die wir initial und zufällig zu anderen Peers im Netzwerk aufbauen (Default Network Shortcuts).

Inbound uni-direktionale Verbindungen verwenden wir zu Erstellung von Shortcuts, die durch den Empfänger einer Anfrage gebildet werden. Beispiele dafür sind Shortcuts zu Peers, die eine Anfrage stellen, die uns selbst interessiert (passive Recommender Shortcuts) oder Shortcuts zu Peers, die uns in ihrer Anfrage mitteilen, wie gut sie im Netzwerk mit anderen Peers vernetzt sind (Bootstrapping Shortcuts).

Query/Non-Query Dependent Shortcuts. Um eine Anfrage möglichst nicht kostenintensiv über das Netzwerk fluten zu müssen, unterscheiden wir zwischen Shortcuts, die einem bestimmten Thema (siehe Definition 4.2.3) zugeordnet sind und Shortcuts, die nicht diese Eigenschaft aufweisen. Damit berücksichtigen wir einerseits Anfragen, die themenspezifisch entlang von Content Provider oder Recommender Shortcuts weitergeleitet werden können. Andererseits unterstützen wir auch Anfragen, für die noch keine thematisch passenden Peers identifiziert wurden. Diese werden entlang von Bootstrapping Shortcuts zur Peers weitergeleitet, die bereits gut vernetzt sind.

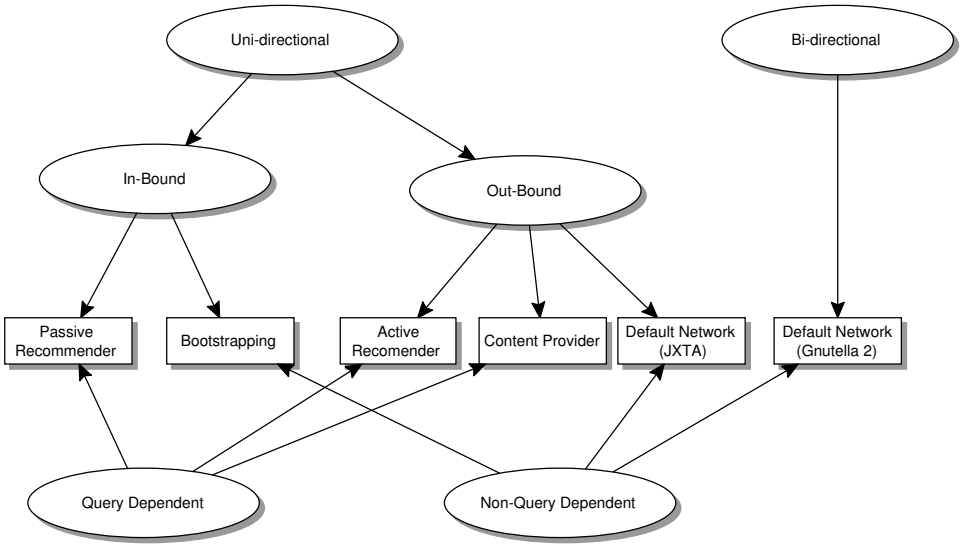


Abbildung 5.1: Klassifikation von Shortcuts

5.2 Anfrageabhängige Shortcuts

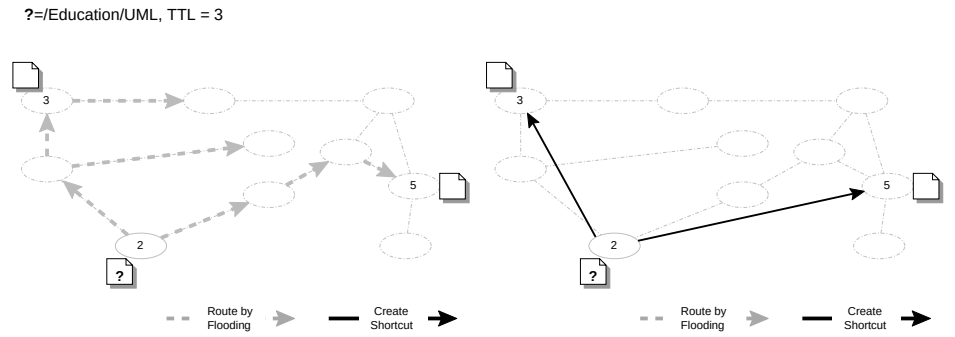
In diesem Abschnitt beschreiben wir, wie die Shortcut Management Komponente die Erstellung zweier neuer Typen von Shortcuts, *Content Provider Shortcuts* und *Recommender Peer Shortcuts*, durchführt. Derartige anfrageabhängige Shortcuts sind genau einem Thema und einem Peer zugeordnet. Wir stellen Einflussgrössen und Beispiele für ihre Erstellung vor, definieren die Form des jeweiligen Shortcut Index und präsentieren Verfahren für dessen Aktualisierung.

5.2.1 Content Provider Layer

Einflussgrössen. Das Design des Content Provider Shortcut Overlay wurde durch kürzlich veröffentlichte Arbeiten von [SMZ03] [TSW04] [Coo04] inspiriert und basiert auf dem einfachen, jedoch leistungsfähigen Prinzip der *Interest-based Locality*.

Definition 5.2.1 (Interest-based Locality für Content Provider Peers). *Kann ein Content Provider Peer eine oder mehrere Anfragen eines Peers erfolgreich beantworten, so ist es wahrscheinlich, das der Content Provider über weitere, für den anfragenden Peer interessante, Dokumente verfügt.* ♣

Wir bewerten die Fähigkeit eines entfernten Peers, passende Dokumente für unsere Anfrage zur Verfügung zu stellen. Der Peer fügt seinem lokalem Index für jede er-



(a) Fluten und Antworten (b) Erstellung

Abbildung 5.2: Content Provider Shortcut Erstellung

folgreiche Antwort Content Provider Shortcuts zu. Dieser Prozess erfolgt solange, bis der Index vollständig belegt ist, andernfalls werden existierende Shortcuts mit neu gefundenen Shortcuts ausgetauscht.

Erstellung. Wenn ein neuer Peer dem Netzwerk beitritt, verfügt er noch über keine Informationen über die Interessen anderer Peers. Durch das Stellen von Anfragen über den Default Network Overlay im INGA Netzwerk durch Fluten, versucht der Peer entfernte Peers für die Beantwortung der Anfrage zu finden. Eine erfolgreiche Anfrage ermittelt eine Menge von Peers, die potentielle Kandidaten für einen Eintrag im Content Provider Shortcut Index des anfragenden Peer darstellen. Wir definieren einen Eintrag für einen Content Provider Shortcut im Index als:

Definition 5.2.2 (Content Provider Shortcut Index). *Sei n_i ein anfragender Knoten, $q(t)$ eine Anfrage mit $t \in \mathcal{T}$, n_j ein antwortender Knoten, c die Kennung für Content Provider Shortcuts und ut die Zeit der Erstellung des Shortcuts, dann definieren wir die Struktur eines Eintrags im Content Provider Shortcut Index von n_i als einen Tuple*

$$co_{n_i} = (t, n_j, \mathcal{H}(q(t), n_j), c, ut) | co_{n_i} \in \mathcal{CO}_{n_i}$$

. ♣

Beispiel 5.2.1 (Erstellung eines Content Provider Shortcut). Grafik 5.2 zeigt die Erstellung von Content Provider Shortcuts am Beispiel von Peer 2. Zuerst flutet der Peer das Netzwerk mit einer Anfrage zum Thema $/Education/UML$ und

einer TTL von drei Hops. Peer 2 erhält Antworten von Peer 3 und 5 (siehe Grafik 5.2(a)). In einem zweiten Schritt fügt Peer 2 seinem lokalen Index für Content Provider Shortcuts zwei neue Einträge hinzu. Grafik 5.1 zeigt die Index Struktur in tabellarischer Form. Entsprechend Definition 5.2.2 enthält der Index das Thema der Anfrage, die PID des antwortenden Peers, die Anzahl der Ergebnisse für die Anfrage, den Typ des Shortcuts und den Zeitpunkt der Erstellung. Im Beispiel hat Peer drei 23 Übereinstimmungen für die Anfrage erzielt und wurde am 21. April 2005 um 10:38:56 erstellt. . ∗

Shortcut ϕ	**PID**	**Query Hits**	**Type**	**Update**
/Education/UML	3	23	c	210405103856
/Education/UML	5	20	c	210405103856

Tabelle 5.1: Index für Peer 2 mit Content Shortcuts

Nachfolgende Anfragen werden mit den bereits bekannten Themen und korrespondierenden Peers im Index verglichen. Die Anfragen können einerseits vom lokalen Peer selbst stammen, in diesem Fall würde der Peer eine Anfrage mehrfach stellen. Viel häufiger tritt der Fall auf, das ein entfernter Peer eine Anfrage mit dem selben Thema stellt. Wird die Anfrage des entfernte Peer zum Index des eigenen lokalen Peer weitergeleitet, profitiert der entfernte Peer vom bereits gesammelten 'Wissen' des eigenen lokalen Peers. Wenn ein Peer kein passendes Thema für die Anfrage in einem Content Provider Index findet, wird für die Anfrage ein Peer auf der Basis anderer lokaler Indizes für Overlays ausgewählt, beispielsweise des Recommender-, Bootstrapping- oder des Default Network Overlays. Geeignete Auswahl- und Routingverfahren stellen wir in Abschnitt 6.2 vor.

5.2.2 Recommender Layer

Einflussgrössen. Typischerweise werden einige Peers besonders erfolgreich im Finden geeigneter Content Provider sein. Durch eine hohe Anzahl von Anfragen zu einem Thema kann ein entfernter Peer ein Spezialwissen über passende Content Provider aufbauen. Ein anfragender Peer ist besonders an einem solchem Peer interessiert, wenn beide eine hohe Überdeckung in ihren Anfragen aufweisen, sie sich also beide für die selben Themen interessieren.

Definition 5.2.3 (Interest-based Locality für Recommender Peers). *Wenn ein entfernter Peer über passende Content Provider Shortcuts für eine Anfrage eines lokalen Peers verfügt, ist es wahrscheinlich, das der entfernte Peer auch über passende Content Provider Shortcuts für nachfolgende Anfragen verfügt bzw. Anfragen mit ähnlichen Themen stellt.* ♣

Ähnlich Content Provider Shortcuts repräsentieren Recommender Shortcuts zusätzliche Kanten in einem Overlay Netzwerk. Wir bewerten jedoch für Recommender Peers nicht die Anzahl der bereits an einem entfernten Peer gespeicherten Dokumente, sondern die Fähigkeit des Recommender Peers die 'richtigen' Anfragen zu stellen und Shortcuts mit für uns relevanten Themen zu anderen Peers im Netzwerk aufzubauen. Wird für eine Anfrage an einem Peer kein geeigneter Content Provider Shortcut gefunden, so wird versucht, für die Anfrage die 'besten' Recommender Shortcuts auszuwählen. Recommender Shortcuts sollen die Wahrscheinlichkeit erhöhen, das für eine Anfrage mit möglichst wenig Nachrichten die passenden Peers gefunden werden. Strategien zur Erstellung von Recommender Shortcuts erfolgen aktiv oder passiv:

Aktive Erstellung durch eigene Anfragen. Eine aktive Erstellung eines Recommender Peers erfolgt durch aktives Senden und Auswerten einer erfolgreichen Anfrage durch den eigenen Peer. Der Sender der Anfrage extrahiert aus dem *Message Pfad* der *Result Message* (siehe Definition 4.3.3) den vorletzten Peer. Dieser Peer hat die Anfrage unmittelbar zum Content Provider weitergeleitet.

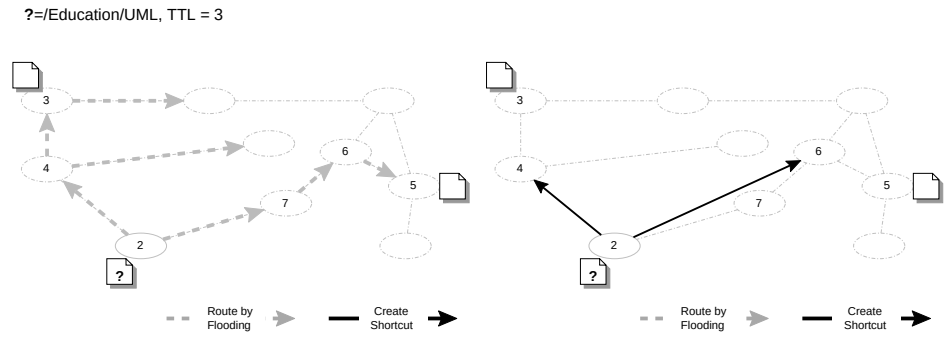
Definition 5.2.4 (Recommender Peer Shortcut Index (aktiv)). *Sei n_i ein anfragender Knoten, $q(t)$ eine Anfrage mit $t \in \mathcal{T}$, n_k der vorletzte Knoten im Message Pfad der Result Message, $\mathcal{H}(q(t), n_j)$ die durch den Shortcut erzielten Query Hits, r für den Typ Recommender Shortcut und ut den Zeitpunkt seiner Erstellung, dann definieren wir die Struktur eines Eintrags im Recommender Index von n_i als einen Tuple*

$$ro_{n_i} = (t, n_k, \mathcal{H}(q(t), n_j), r, ut) | ro_{n_i} \in \mathcal{RO}_{n_i}$$

. ♣

Beispiel 5.2.2 (Aktive Erstellung von Recommender Shortcuts). Grafik 5.3(a) zeigt die aktive Erstellung von Recommender Shortcuts am Beispiel von Peer 2. Basierend auf dem Fluten des Netzwerkes zur Identifikation von Content Provider Shortcuts identifiziert Peer 2 über den Pfad der *Result Message*, das Peer 4 und Peer 6 die Nachricht zu einem Content Provider Peer weitergeleitet haben. Zu diesen Peers erstellt Peer 2 jeweils einen Recommender Shortcut (Grafik 5.3(b) und Tabelle 5.2).

Passive Erstellung durch gezieltes 'Zuhören'. Um den Lernprozess für Recommender Shortcuts weiter zu beschleunigen, werden auch Recommender Shortcuts aufgrund von Anfragen gebildet, die ein lokaler Peer nicht selbst gestellt hat.



(a) Fluten und Antworten (b) Erstellung

Abbildung 5.3: Recommender Shortcut Erstellung (aktiv)

Shortcut	PID	Query Hits	Type	Update
/Education/UML	4	23	r	210405103856
/Education/UML	6	20	r	210405103856
/Education/UML	3	23	c	210405103856
/Education/UML	5	20	c	210405103856

Tabelle 5.2: Index für Peer 2 mit Content und Recommender Shortcuts

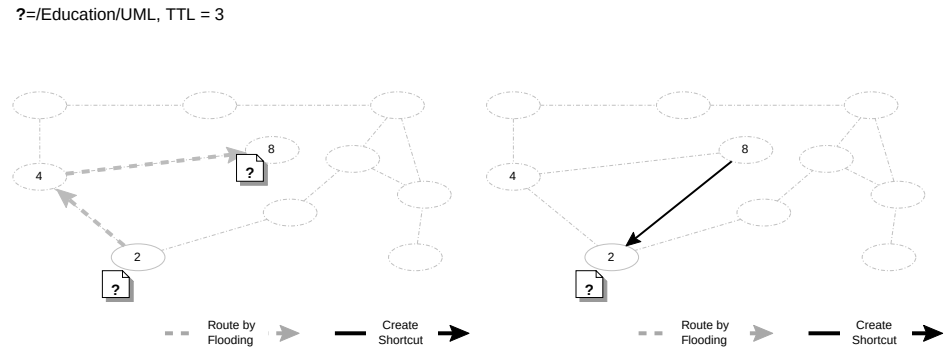
Dazu werden Anfragen anderer Peers ausgewertet, die durch den eigenen lokalen Peer weitergeleitet werden. Im Gegensatz zu der aktiven Strategie zur Erstellung von Recommender Shortcuts sammeln wir dadurch für populäre Themen schnell Informationen über passende Peers ohne eigene Aktivität.

Definition 5.2.5 (Recommender Peer Shortcut Index (passiv)). *Sei n_l ein anfragender Knoten, n_i ein Knoten durch den die Anfrage geleitet wird, $q(t)$ eine Anfrage mit $t \in \mathcal{T}$, r für den Type Recommender Shortcut und ut für den Zeitpunkt seiner Erstellung dann definieren wir die Struktur eines Eintrags im Recommender Index von n_i als einen Tuple*

$$ro_{n_i} = (t, n_l, \mathcal{H}(q(t), n_j), r, ut) | ro_{n_i} \in \mathcal{RO}_{n_i}$$

Da wir die Anzahl der erzielten Query Hits *nicht kennen, setzen wir $\mathcal{H}(q(t), n_j = 1$ in der optimistischen Annahme, das n_l für die Anfrage passende Peers findet.* ♣

Beispiel 5.2.3 (Passive Recommender Short Erstellung). Grafik 5.4 zeigt die Erstellung von Recommender Shortcuts mit einer passiven Strategie. Peer 8 erhält



(a) Fluten und Antworten (b) Erstellung

Abbildung 5.4: Recommender Shortcut Erstellung (passiv)

durch Fluten eine Anfrage zum Thema */UML/Education*. Anhand der *Query Message* (siehe Definition 4.3.2) wird Peer 2 als anfragender Peer ermittelt. Zu diesem Peer erstellt Peer 8 einen Recommender Shortcut (Grafik 5.4(b) und Tabelle 5.3).

Shortcut	**PID**	**Query Hits**	**Type**	**Update**
/Education/UML	2	1	r	210405103856

Tabelle 5.3: Index für Peer 8 mit passivem Recommender Shortcut

5.3 Anfrageunabhänige Shortcuts

Anfrageunabhängige Shortcuts erlauben das Weiterleiten einer Anfrage unabhängig von dem gewählten Thema der Anfrage. Verfügt ein Peer über keine Shortcuts, muss er eine initiale Menge von Verbindungen zu anderen Peers aufbauen. Im ersten Teil des Abschnittes beschreiben wir den Prozess zum Aufbau des *Default Network Overlays*. Ist dieser Overlay gebildet, können *Bootstrapping Shortcuts* zu besonders gut vernetzten Peers erstellt werden. Diese anfrageunabhängigen Shortcuts erlauben eine Reduzierung der Nachrichten gegenüber dem Default Overlay Network. Im zweiten Teil des Abschnittes stellen wir Strategien zur Erstellung und Aktualisierung dieses neuen Shortcut Typs vor.

5.3.1 Default Network Layer

Einflussgrössen. Um einem unstrukturierten Netzwerk beizutreten, muss ein Peer eine Menge Peers kennenlernen, die bereits mit dem Netzwerk verknüpft sind. Diese initiale Menge an Peers erlaubt das Auffinden weiterer Peers im Netzwerk. Wir bezeichnen sie als Default Network Layer. Historisch gesehen wurden diese Peers in den frühen Tagen von Gnutella noch mündlich oder in Newsgroups propagiert. Dadurch wurden häufig Peers miteinander vernetzt, die ähnliche Interessen aufwiesen. Aufgrund dieser Lokalität benötigten Anfragen wenige Nachrichten [Ora01]. Mit der Schliessung von Napster und der steigenden Popularität von Gnutella wurde dieser Prozess durch neue Technologien automatisiert. Dadurch wurden Peers 'zufällig' miteinander vernetzt, was in einem erhöhten Aufkommen von Nachrichten für eine Anfrage resultierte.

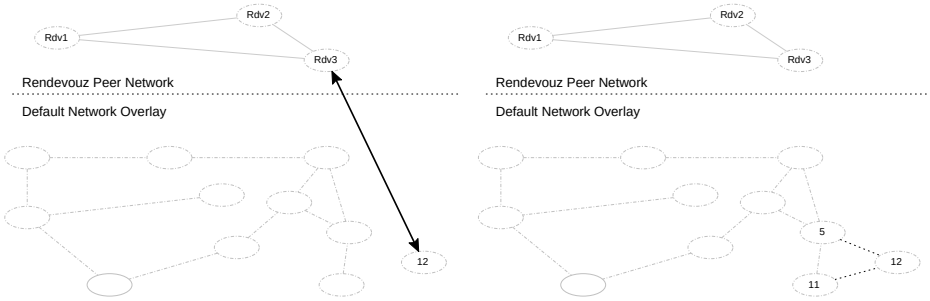
Definition 5.3.1 (Default Network Overlay). *Als Default Network Overlay eines Peers bezeichnen wir eine Menge von zufällig ausgewählten Peers, die aufgrund der initialen Netzwerk Struktur mit dem Peer benachbart sind.* ♣

Erstellung. Die Erstellung des Default Network Layers basiert auf dem Rendezvous Protokoll des JXTA Netzwerkes. Zusätzlich zum eigentlichen Peer-to-Peer Netzwerk existiert ein Netzwerk von Rendezvous Super-Peers. Diese speichern die PID von Peers im Netzwerk. Strategien zum Austausch und zur Aktualisierung von PIDs zwischen den Rendezvous Peers werden in [TAA+03] vorgestellt. Im Gnutella Netzwerk existiert ein ähnliches verteiltes Konzept, der *Gnutella Web Cache*. In ihm werden die IP-Adressen aktuell mit dem Netzwerk verbundener Peers über das HTTP Protokoll registriert, aktualisiert und publiziert [KAD+04].

Definition 5.3.2 (Default Network Overlay Index). *Sei n_i ein anfragender Knoten und n_j ein benachbarter Knoten, dann definieren wir die Struktur eines Eintrag im Default Network Overlay von n_i als*

$$do_{n_i} = (n_j) | do_{n_i} \in \mathcal{DO}_{n_i}$$

Zu Beginn verbindet sich ein Peer zu einem beliebigen Rendezvous über das HTTP oder TCP Protokoll. Der Peer erhält eine Liste von PIDs zufällig ausgewählter Peers, die bereits mit dem *Default Overlay Netzwerk* verbunden sind. Im JXTA Protokoll basiert die Liste der Peers auf den aktuell am Rendezvous Peer registrierten Peers. Zu einer Teilmenge dieser Peers werden neue Verbindungen erstellt. Auf der Basis dieser Verbindungen kann eine Suche und Erstellung weiterer Shortcuts erfolgen. Schliesslich registriert sich der Peer ebenfalls an einem Rendezvous Peer. Der Prozess wird wiederholt, wenn benachbarte Peer aus dem eigenen Default Network Overlay das Netzwerk verlassen.



(a) Verbinden mit Rendezvous Peer (b) Erstellung

Abbildung 5.5: Erstellung des Default Network Overlay

Beispiel 5.3.1 (Default Overlay Network). Grafik 5.5 zeigt den zweistufigen Prozess der Registrierung im Default Network Overlay für $Peer$ 12. Der Peer verbindet sich mit dem zufällig ausgewählten Rendezvous Peer $Rdv3$ und erhält eine Liste bereits am Netzwerk teilnehmender Peers (Grafik 5.5(a)). Peer 12 verbindet sich zu Peer 5 und Peer 11 und registriert sich ebenfalls als Teilnehmer des Netzwerkes im Rendezvous Peer $Rdv3$ (Grafik 5.5(b)). $*$

5.3.2 Bootstrapping Layer

Einflussgrössen. Manchmal hat ein Peer für eine Anfrage kein passendes Thema in seinem lokalen Index registriert. In diesem Fall muss der Peer die Anfrage über einen Shortcut weiterleiten, der nicht einem spezifischen Thema zugeordnet ist. Bisher haben wir dazu nur das Weiterleiten über Shortcuts des Default Network Overlay betrachtet. Diese Strategie basiert auf dem Fluten des Netzwerkes und ist mit einer hohen Anzahl an Nachrichten verbunden. Adamic et.al. hat in [ALPH01] eine *High Out Degree* Strategie vorgestellt, bei der die Anzahl der Nachrichten in einem unstrukturierten Netzwerk im Vergleich zur naiven Strategie der Überflutung des Netzwerkes mit Nachrichten deutlich gesenkt wird. Wir wenden diese Erkenntnis für die Definition des Bootstrapping Overlay an.

Die Eigenschaft eines Peers für das Bootstrapping wird über seine *Bootstrapping Capability* definiert. Wir definieren *Bootstrapping Nodes* als Peers, die einerseits viele ausgehende Shortcuts zu disjunkten Peers registrieren konnten, also einen hohen *OutDegree* aufweisen und zu denen möglichst viele disjunkte Peers einen Shortcut aufgebaut haben, also Peers die einen hohen *Indegree* aufweisen.

Definition 5.3.3 (Bootstrapping Capability). *Sei* OutDegree$_{n_i}$ *die Anzahl disjunkter Peers zu denen* n_i *Shortcuts registriert hat und* InDegree$_{n_i}$ *die Anzahl unterschiedlicher Peers, die mit einen Shortcut auf* n_i *verweisen, dann definieren wir die Bootstrapping Capability* BC_{n_i} *von* n_i *wie folgt:*

$$BC_{n_i} = (Outdgree_{n_i} + 1) \times (InDegree_{n_i} + 1)$$

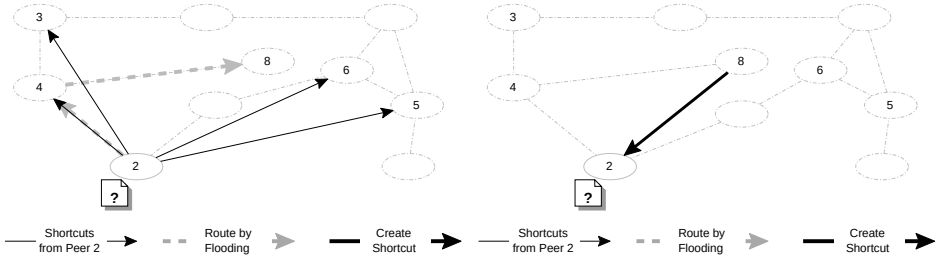
Zur Vermeidung von 'Null' Werten begrenzen wir beide Werte. Um eine Annährung des InDegree in einem verteilten Netzwerk ohne vorhandenes globales Wissen zu ermöglichen, registrieren wir für jeden gespeicherten Shortcut den vorletzten Peer aus dem Pfad der Anfrage, da diese Peers einen Shortcut auf den lokalen Peer gespeichert haben. Bei aktiven Recommender Shortcuts ist dieser Peer identisch mit dem Recommender Peer. Die Anzahl disjunkter Peers ergibt eine Annäherung des Indegree. Ein hoher Wert von BC_{n_i} beschreibt einen Peer, der mit vielen Peers über lokalen Shortcuts vernetzt ist und dessen Shortcuts für viele andere Peers relevant sind. Ähnlich [Kle98] verstehen wir derartige Peers als adaptive, themenunabhängige *Information Hubs* im Netzwerk. Sie sind besonders hilfreich, für Peers, die dem Netzwerk neu beigetreten sind und die noch keine Peers für ihre Anfragen sammeln konnten (*warm-up problem*) bzw. für Peers die eine Anfrage zu einem Thema stellen, für das sie keine lokalen Referenzen registriert haben.

Erstellung. Zur Gewinnung der Bootstrapping Information verwenden wir eine Strategie, bei der jeder Peer in der Anfrage die eigenen Bootstrapping Informationen in der *Query Message* (siehe auch Definition 4.3.2) sendet. Zu jeder Anfrage, die durch den Peer geleitet wird, speichern wir die Bootstrapping Capability des anfragenden Peers.

Definition 5.3.4 (Bootstrapping Index). *Sei* n_i *ein anfragender Knoten,* BC_{n_i} *seine Bootstrapping Capability und* n_j *ein Knoten, durch den eine Anfrage geleitet wird, dann definieren wir einen Index Eintrag in* n_j *als*

$$bo_{n_j} = (n_i, BC_{n_i}) | bo_{n_j} \in \mathcal{BO}_{n_j}$$

Beispiel 5.3.2 (Bootstrapping Index). In Grafik 5.6 sendet Peer 2 eine Anfrage und mit der *Query Message* auch seine Bootstrapping Capability. Wir nehmen an, das Peer 2 bisher vier Shortcuts zu vier Peers für ein Thema erstellt hat. Die Anfrage erhält auch Peer 8, der diese Information in seinem Bootstrapping Index speichert und einen Bootstrapping Shortcut zu Peer 2 aufbaut. Tabelle 5.4 erweitert das Beispiel aus Tabelle 5.3 und zeigt den vollständigen Index von Peer 8 mit Bootstrapping Informationen.



(a) Fluten (b) Erstellung

Abbildung 5.6: Bootstrapping Shortcut Erstellung

Shortcut	PID	Query Hits	Type	Update	BO
/Education/UML	2	1	r	210405103856	4

Tabelle 5.4: Index Peer 8 mit Bootstrapping Informationen

5.4 Verwandte Arbeiten

Die Verwendung von Shortcuts zur Verbesserung des Routings in unstrukturierten Peer-to-Peer Netzwerken ist noch wenig untersucht. Studien [FHKM04, GDS+03] haben gezeigt, das Peers sowohl geographische oder Interessen-basierte Lokalität aufweisen.

Die Autoren in [SMZ03, Coo04, BCAA04] definieren *Interest-based Locality* durch Shortcuts zu Peers, die eine Anfrage in der Vergangenheit exakt beantworten konnten. Sie zeigen, das durch den Einsatz dieser Shortcuts in einem unstrukturierten Netzwerk die Anzahl der Nachrichten gesenkt werden kann. Wir haben die Idee beider Arbeiten aufgegriffen und in die Erstellung von Content Provider Shortcuts einfließen lassen. Unser Modell der verschiedenen Shortcuts Overlays berücksichtigt ausserdem Peers, die sich durch erfolgreiche Anfragen und eine gute Vernetzung auszeichnen.

Arbeiten von [CW04, CMH+02] und neuere Arbeiten zum *Freenet Next Generation Routing Algorithm* untersuchen Shortcuts die auf der geographische Lokalität von anfragenden und antwortenden Peers basieren. Ihre Arbeiten beruhen auf der These, das geographisch nahe Peers mit höherer Wahrscheinlichkeit eine Anfrage beantworten können, als entfernte Peers. Ziel ist die Reduzierung der Latenzzeit und die Optimierung der verfügbaren Bandbreite im Netzwerk. Diese Arbeitens sind orthogonal zu unserer Arbeit. Sie beziehen sich auf die Optimierung der Netzwerk Struktur auf der Basis technischer Kriterien, während wir Peers

gruppieren, die an gleichen Themen interessiert sind.

5.5 Zusammenfassung

Dieses Kapitel zeigt Strategien zur Erstellung von Shortcuts in unstrukturierten Peer-to-Peer Netzwerken. Der Beitrag dieses Kapitels liegt in der Klassifikation von Shortcuts und in der Entwicklung neuer, jedoch einfacher, Strategien zur Erstellung von Shortcuts. Wir unterscheiden zwischen Shortcuts, die abhängig oder unabhängig von dem konkreten Thema der Anfrage erzeugt werden. Zur ersten Kategorie gehören Shortcuts zu Peers, die eine Antwort auf eine Anfrage geben konnten Shortcuts, Shortcuts zu Peers die eine Nachricht weitergeleitet haben und Shortcuts zu Peers, die eine Anfrage zu diesem Thema bereits in der Vergangenheit gestellt haben. Zur letzteren Kategorie gehören Shortcuts zu Peers, die gut vernetzt sind und Shortcuts zu zufällig ausgewählten Peers. Für den Aufbau von Shortcuts analysieren wir die Resultate einer Anfrage, den Weg unserer Anfrage über entfernte Peers und Anfragen, die durch einen lokalen Peer weitergeleitet wurden. Shortcuts werden in zwei lokalen Index gespeichert, ein Index Shortcuts zum Default Network Layer, der den 'Index' momentan existierender unstrukturierter Netzwerke abbildet und ein erweiterter Index, der Content-, Recommender- und Bootstrapping Shortcuts abbildet.

6
Index Verwaltung und Routing Modell

Dieses Kapitel ist der Auswahl von Peers für eine Anfrage auf der Basis lokal registrierter Shortcuts gewidmet. Wir zeigen den Ablauf des Routings von Anfragen in Shortcut Netzwerken mit den beiden Kernfunktionen Index Management und Shortcut Auswahl. Die Auswahl von registrierten Shortcuts für eine Anfrage und das Löschen bereits registrierter Shortcuts beeinflussen sich gegenseitig. Im ersten Teil dieses Kapitels präsentieren wir einen neuen Routing Algorithmus zur Auswahl von Shortcuts. Der Algorithmus berücksichtigt dynamisch, welche Shortcuts bereits in den verschiedenen lokalen Indizes vorhanden sind. Im zweiten Teil des Kapitels stellen wir eine Strategie zur Aktualisierung der Indizes vor.

6.1 Interaktionen und Ablaufmodell

Das INGA Netzwerk beinhaltet verschiedene Aktivitäten, die entweder von einem Peer lokal oder in Interaktion mit entfernten Peers ausgeführt werden. Grafik 6.1 zeigt eine Übersicht des Verhaltens lokaler Peers und entfernter Peers während des Anfrageprozesses.

Lokal: Senden der Anfrage. Der lokale Peer stellt zuerst die eigenen Informationen für die Query Message entsprechend Definition 4.3.2 zusammen (*Prepare Query*). Dann wählt der lokale Peer aufgrund seines zu diesem Zeitpunkt vorhandenen lokalen Wissens dynamisch passende Shortcuts für die Anfrage aus (*Se-*

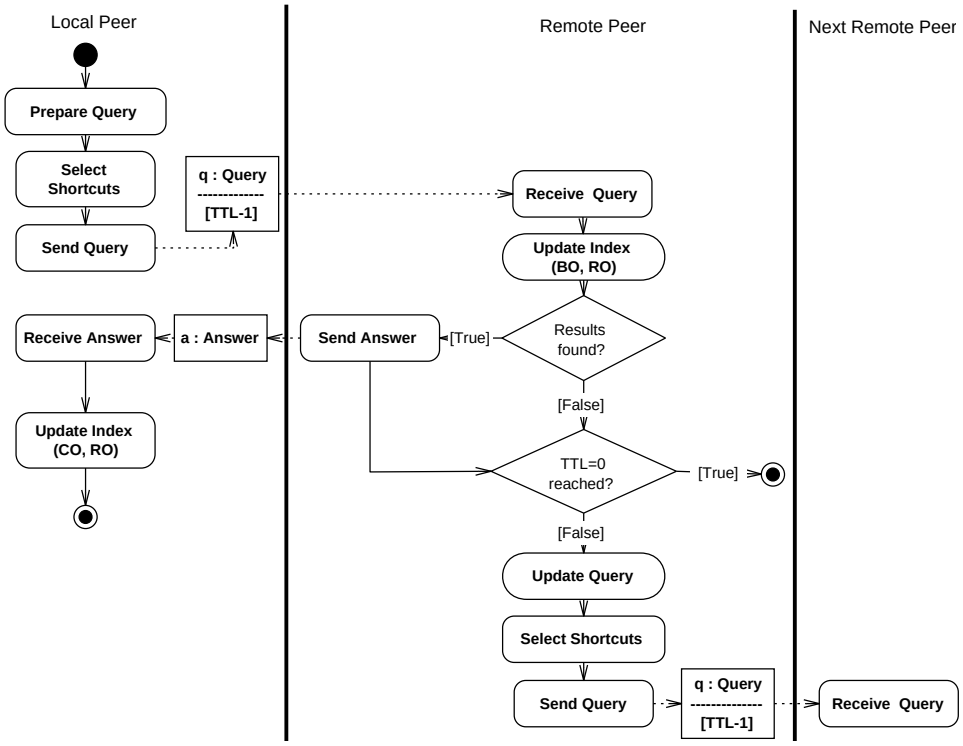


Abbildung 6.1: Routing und Aktualisierung

lect Shortcuts), beispielsweise im 'Idealfall' passende Content Provider Shortcuts. Schließlich wird die Anfrage über die ausgewählten Shortcuts an entfernte Peers gesendet (*Send Query*) und die Anzahl verbleibender Hops (*TTL*) reduziert.

Im Netzwerk: Beantworten der Anfrage. Erhält ein entfernter Peer eine Anfrage, werden die in ihr enthaltenen Informationen, wie anfragender Peer, Bootstrapping Capability oder das Thema der Anfrage extrahiert und der lokale Recommender und Bootstrapping Index aktualisiert (*Update Index (BO,RO)*). In Abschnitt 6.3 stellen wir dazu geeignete Algorithmen vor. Dann versucht der Peer passende Antworten für die Anfrage zu finden. Ist das der Fall, erzeugt der entfernte Peer eine Result Message entsprechend Definition 4.3.3 und sendet die Resultate zum anfragenden Peer zurück (*Send Answer*).

Im Netzwerk: Weiterleiten der Anfrage. Erhält ein Peer eine Anfrage in der die maximale Anzahl möglicher Hops noch nicht erreicht ist, wird die Anfrage

mit der ID des aktuellen Peers ergänzt (*Update Query*). Ebenfalls werden dynamisch weitere Shortcuts zur Weiterleitung der Anfrage ausgewählt (*Select Shortcuts*). Basierend auf der Ähnlichkeit der Anfrage mit bereits registrierten Shortcuts sind das entweder anfrageabhängige oder anfrageunabhängige Shortcuts. In Abschnitt 6.2 stellen wir einen Algorithmus zur dynamischen Auswahl der Shortcuts vor. Schließlich wird die Anfrage an weitere ausgewählte Peers gesandt und die Anzahl verbleibender Hops in der Anfrage reduziert (*Send Query*).

Lokal: Empfangen von Antworten. Wenn ein Peer eine Antwort empfängt (*Receive Answer*), werden aus der Result Message (siehe Definition 4.3.3) Informationen über die Anzahl der Antworten, den antwortenden Peer und weiter leitende Peers extrahiert und der lokale Recommender und Content Provider Index aktualisiert (*Update Index (Co, RO)*).

6.2 Dynamisches Routing Modell

In Abschnitt 5.1 definieren wir vier Typen von Shortcuts, die an einem Peer registriert werden können. Für jeden dieser Typen existieren in der Literatur bereits Routing Strategien, die mit unterschiedlicher Effizienz eine Anfrage weiterleiten. Wir stellen die Strategien kurz vor und adaptieren sie für das INGA Netzwerk. Im zweiten Teil stellen wir einen neuen Algorithmus vor, der die *top-k* Shortcuts für eine Anfrage an einem Peer lokal auswählt. Der Algorithmus untersucht, welche Shortcuts für das Thema Anfrage bereits bekannt sind und wählt dynamisch die passende(n) Routing Strategie(n) aus.

6.2.1 Greedy Routing

Strategie und Ziel. Analog zu den Arbeiten von Kleinberg [Kle00b] und Milgram verwenden wir eine Greedy Routing Strategie, die eine Nachricht über die *top-k* Shortcuts leiten und die größte Ähnlichkeit mit der Anfrage aufweisen. Das INGA Netzwerk verwendet eine Funktion zur Messung der semantischen Ähnlichkeit zwischen einer Anfrage und einem anfrageabhängigen Shortcut. Derartige Funktionen sind häufig domänenspezifisch und hängen von der verwendeten Ontologie ab. Wir benutzen die von Li et.al. [LBM03] vorgeschlagene Ähnlichkeitsfunktion für Themen-Hierarchien.

Definition 6.2.1 (Semantische Ähnlichkeit in einer Themen-Hierarchie). *Sei* $sim_{Topic} : Q \times \Phi \mapsto [0, 1]$ *eine Ähnlichkeitsfunktion zwischen einer Anfrage* $q \in Q$ *und einem anfrageabhängigen Shortcut* $\phi \in \Phi$. *Weiterhin seien beide Element*

derselben Themen -Hierarchie $q, \phi \in \mathcal{T}$. Wir definieren deren Ähnlichkeit als

$$sim_{Topic}(q, \phi) = \begin{cases} e^{-\alpha l} \cdot \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} & \text{if } q \neq \phi \\ 1 & \text{otherwise} \end{cases}$$

wobei l die Länge der kürzesten Verbindung zwischen q und ϕ und h die minimale Höhe entweder von q oder ϕ in der Themen-Hierarchie bildet. α und β sind Konstanten, die den Beitrag von h und l skalieren. Folgend den Experimenten von [LBM03] verwenden wir α=0.2 und β=0.6 . ♣

Die Funktion sim_{Topic} weist für eine hohe Ähnlichkeit einen Wert nahe 1 und für eine geringe Ähnlichkeit einen Wert nahe 0 auf.

Algorithmus. Algorithmus 1 zeigt den *TopGreedy* Algorithmus. Er wählt für eine Anfrage die *top-k* anfrageabhängigen Shortcuts aus, die über einem Grenzwert liegen. Der Algorithmus sucht die ähnlichsten Shortcuts für die Anfrage

Algorithm 1 TopGreedy

Require: **Query** q, **Set** *index*, **int** k, **int** t_{greedy}
1: *topShortcuts* $\leftarrow \{\}$
2: *s_tmp* $\leftarrow$ *index*
3: **while** (*s_tmp is* not empty) $\wedge$ ($k > 0$) **do**
4: *Next* $\leftarrow maxSim_{Topic}(q, s\_tmp)$
5: **if** sim_{Topic} (*q*,*Next*) $> t_{greedy}$ **then**
6: *s_tmp* $\leftarrow$ *s_tmp* $-$ (*Next*)
7: **if** (*Next* routes not to a peer in *topShortcuts*) **then**
8: *topShortcuts* $\leftarrow$ *topShortcuts* + *Next*
9: $k \leftarrow k - 1$
10: **end if**
11: **else**
12: break
13: **end if**
14: **end while**
15: **Return** *topShortcuts*

(Zeile 4-6), beginnend mit dem Shortcut mit der höchsten Ähnlichkeit. Wir verwenden einen Schwellwert t_{greedy} um Peers mit einer geringen Ähnlichkeit nicht zu berücksichtigen. Der Algorithmus vermeidet überlappende Shortcuts, beispielsweise zwei gewählte Shortcuts, die zu dem gleichen Peer führen (Zeile 7-9). Der Algorithmus bricht ab, wenn bereits alle Shortcuts geprüft oder *top-k* Shortcuts

ausgewählt wurden (Zeile 3) und wenn keine Shortcuts mehr vorliegen, die einen gewählten Grenzwert überschreiten (Zeile 11-12).

6.2.2 High Out-Degree Routing

Strategie und Ziel. Im INGA Netzwerk wählen wir die Knoten aus, die viele Peers zu unterschiedlichen Themen kennen. Dazu verwenden wir die *Bootstrapping Capability* entsprechend Definition 5.3.3. Für eine Anfrage wählen wir die lokal registrierten Bootstrapping Shortcuts mit der höchsten Bootstrapping Capability aus. Zusätzlich müssen diese Knoten über eine höhere Bootstrapping Capability als der lokale Peer verfügen. Damit verhindern wir ein Weiterleiten einer Anfrage an Peers mit einer geringen Vernetzung, ausgehend von der Vernetzung des eigenen Peers.

Algorithmus. Algorithmus 2 zeigt den *TopBootstrapping* Algorithmus. Analog zum *TopGreedy* Algorithmus wählt er für eine Anfrage die *top-k* Bootstrapping Shortcuts aus. Der Algorithmus beginnt mit der Auswahl von Shortcut mit der

Algorithm 2 TopBootstrapping

Require: **Set** *index*, **int** k, **int** *local_bo*

```
 1: topShortcuts ← {}
 2: s_tmp ← index
 3: while (s_tmp is not empty) ∧ (k > 0) do
 4:    Next ← maxBC(s_tmp)
 5:    s_tmp ← s_tmp − (Next)
 6:    if (Next routes not to a peer in topShortcuts) then
 7:       if (Next.GetBo() > local_bo) then
 8:          topShortcuts ← topShortcuts + Next
 9:          k ← k − 1
10:       end if
11:    end if
12: end while
13: Return topShortcuts
```

höchsten Bootstrapping Capability, deren Bootstrapping Capability grösser ist, als der lokale Wert (Zeile 4). Dabei vermeidet der Algorithmus überlappende Shortcuts, beispielsweise zwei gewählte Shortcuts, die zu dem gleichen Peer führen (Zeile 6-8). Der Algorithmus bricht ab, wenn bereits alle Shortcuts geprüft oder *top-k* Shortcuts ausgewählt wurden (Zeile 3).

6.2.3 Firework Query Routing

Strategie und Ziel. Die Autoren in [KNS04, NS02] verwenden in ihren Arbeiten das Fireworks Query Model. Eine Anfrage wird mit einer *top-k* Strategie an ausgewählte Peers durch das Netzwerk geleitet, bis Peers gefunden werden, deren Shortcuts exakt zur Anfrage passen. In diesem Fall wird die Anfrage an alle Peers geleitet. Die Autoren folgen der Idee, das eine Anfrage mit hoher Wahrscheinlichkeit einen Cluster erreicht hat, der viele Peers umfasst, die Interesse am Thema der Anfrage haben. Im INGA Netzwerk wird eine Anfrage über alle Content Provider Shortcuts geleitet, die eine exakte Übereinstimmung mit der Anfrage aufweisen. Andernfalls wird die Anfrage über eine Greedy-, Bootstrapping- oder Serendipity Routing Strategie geleitet.

Algorithmus. Algorithmus 3 zeigt den *FireWorks* Algorithmus, der alle Content Provider auswählt, die exakt zur Anfrage passen (Zeile 4). Analog zu *TopGreedy*

Algorithm 3 Fireworks

Require: **Query** q, **Set** *index*
1: *fwShortcuts* $\leftarrow \{\}$
2: *s_tmp* $\leftarrow$ *index.getContentProviderShortcuts()*
3: **while** (*s_tmp is* not empty) **do**
4: *Next* $\leftarrow exactSim_{Topic}(q, s\_tmp)$
5: *s_tmp* $\leftarrow$ *s_tmp* $-$ (*Next*)
6: **if** (*Next* routes not to a peer in *fwShortcuts*) **then**
7: *fwShortcuts* $\leftarrow$ *fwShortcuts* + *Next*
8: **end if**
9: **end while**
10: **Return** *fwShortcuts*

und *TopBootstrapping* vermeidet der Algorithmus überlappende Shortcuts (Zeile 6-8). Der Algorithmus bricht ab, wenn bereits alle Content Provider Shortcuts geprüft wurden (Zeile 3).

6.2.4 Serendipity Routing

Strategie und Ziel. Analog zu den lokalen Suchalgorithmen für die *Ant Colonization Optimization* tritt bei der Verwendung einer Greedy Routing Strategie das *Shortcut Problem* [SHB97, DG89] auf:

Definition 6.2.2 (Shortcut Problem). *Wird eine Anfrage nur über bereits registrierte Shortcuts geleitet, werden Peers die dem Netzwerk neu beitreten und bereits registrierte Peers nicht berücksichtigt.* ♣

Wir implementieren eine adaptive Anpassung der Shortcuts über zwei Funktionalitäten:

1. Ein Peer löscht Shortcuts, die in der Vergangenheit eine schlechte Performanz aufwiesen. Dazu verwenden wir die in Abschnitt 6.3 vorgestellten Index Management Strategien.

2. Wir leiten eine bestimmten Anteil der Anfragen über zufällig ausgewählte Verbindungen. Im INGA Netzwerk erfolgt ein Routing mit einer Wahrscheinlichkeit von $(1 - f)$ über einen Shortcut aus dem Bootstrapping, Content Provider oder Recommender Overlay und einer Wahrscheinlichkeit von f über den Default Network Overlay. [MSZ03, TSW04] haben den Effekt eines unerwarteten Zufalls (Serendipity) in ihren Versuchen in Shortcut Netzwerken nachgewiesen.

Algorithmus. Algorithmus 4 zeigt den *Serendipity* Algorithmus, der eine bestimmte, durch den Parameter $f|f \in [0, 1] \wedge f \in \mathcal{R}$ definierte, Anzahl zufälliger Shortcuts mit einer Menge bereits ausgewählten Shortcuts austauscht. Zuerst prüft der Algorithmus, ob der Auswahl von Shortcuts noch mehr zufällige Shortcuts hinzugefügt werden müssen, als durch den Parameter f bestimmt wurde (Zeile 2). Ist das nicht der Fall entscheidet der Algorithmus durch eine gleichförmige Zufallsverteilung, ob ein bereits ausgewählter Shortcut durch einen Shortcut aus dem Default Network Overlay ausgetauscht wird. Fehlende Shortcuts werden durch Shortcuts aus dem Default Overlay Network ergänzt (Zeile 12-15).

6.2.5 Dynamische Auswahl der Strategie

Strategie und Ziel. Die Aufgabe der Routing Komponente eines INGA Peers ist die besten Shortcuts für eine Anfrage zu finden. Dieser Prozess erfolgt dynamisch in Abhängigkeit von dem im lokalen Index für die Anfrage verfügbaren anfrageabhängigen und anfrageunabhängigen Shortcuts. Entsprechend den in Abschnitt 4.2.2 vorgestellten sozialen Metaphern definieren wir folgende Heuristiken für die Auswahl von Routing Algorithmen:

1. Zuerst sucht ein Peer nach Content Provider Shortcuts, die mit der Anfrage exakt übereinstimmen und die Anfrage unmittelbar zu einem Peer schicken, der passende Dokumente hat.

Algorithm 4 Serendipity

Require: **Set** *preSelected*, **Set** *defaultNetWorkShortcuts*, **int** f, **int** k

Set *postSelected* $\leftarrow \{\}$
if ($\frac{k-|preSelected|}{k} < f$) **then**
  while (*preSelected is* not empty) **do**
    Next $\leftarrow$ *next(preSelected)*
    preSelected $\leftarrow$ *preSelected* $-$ (*Next*)
    if (*rand*(0,1) $> f$) **then**
      postSelected $\leftarrow$ *postSelected* + *Next*
    end if
  end while
end if
$k \leftarrow\ k - |postselected|$
while $k > 0$ **do**
  postSelected $\leftarrow$ *postSelected* + *next(defaultNetworkShortcuts*)
  $k \leftarrow k - 1$
end while
Return *postSelected*

2. Werden nur ungenügend exakt passende Content Provider Shortcuts gefunden, versucht ein Peer zu einer Anfrage ähnliche Content Provider- und Recommender Shortcuts zu finden.

3. Hat ein Peer ungenügend anfrageabhängige Shortcuts gefunden, die eine minimale Ähnlichkeit mit der Anfrage aufweisen, versucht der Peer die Anfrage an Peers mit einer besonders guten Vernetzung weiter zu leiten.

4. Hat ein Peer keine Shortcuts mit den vorhergehenden Strategien finden können, sendet er die Anfrage an seine durch die initiale Netzwerk Struktur vorgegebenen Nachbarn. Um für eine Anfrage zusätzliche Peers kennenzulernen und ein Verharren in einem möglichen lokalen Maximum bei der Verwendung einer Greedy Strategie vorzubeugen, tauschen wir bereits bewußt gewählte Shortcuts mit einigen zufällig gewählten Shortcuts aus.

Algorithmus. Algorithmus *Dynamic* wird in Algorithmus 5 gezeigt. Er wählt für eine Anfrage die *top-k* Shortcuts aus. Der Algorithmus prüft, ob eine Anfrage bereits die maximale Anzahl von Hops erreicht hat. Entsprechend unseren Heuristiken ruft der Algorithmus zuerst eine Fireworks Strategie auf. Vor dem Aufruf weiterer Routing Strategien prüft der Algorithmus, ob die gewünschte Anzahl von

Algorithm 5 Dynamic

Require: **Query** q, **int** k, **int** t_{Greedy}
Ensure: $TTL_q < maxTTL$
1: $s \leftarrow \{\}$
2: $s \leftarrow$ *Fireworks(q,index)*
3: **if** $(|s| < k)$ **then**
4: $s \leftarrow$ *TopGreedy(q,index,*$(k - |s|)$*,*t_{Greedy}*)*
5: **end if**
6: **if** $(|s| < k)$ **then**
7: $s \leftarrow$ *TopBootStrapping(index,*$(k - |s|)$*)*
8: **end if**
9: $s \leftarrow$ *Serendipity(s,defaultNetworkShortcuts,f,k)*
10: **Return** s.

Shortcuts bereits erreicht wurde. Zuletzt werden einige bereits gewählte Shortcuts durch zufällige Shortcuts ausgetauscht.

6.3 Index Management

Mit zunehmender Anzahl der eigenen Anfragen und der weitergeleiteten Anfragen anderer Peers, wird der Content Provider, Recommender und Bootstrapping Index wachsen. Entsprechend den in Abschnitt 4.2.2 definierten sozialen Metaphern und in Abschnitt 4.1 vorgestellten Designentscheidungen wird ein Peer nur eine begrenzte Anzahl von Shortcuts verwalten. Für die Begrenzung des Index identifizieren wir folgende Ursachen:

1. Die Anzahl der registrierten Peers ist durch den lokal verfügbaren Plattenplatz, Hauptspeicher und die Prozessorleistung begrenzt.

2. Ein Peer verwaltet 'egoistisch' nur die Peers, die für *seine eigenen* Anfragen relevante Dokumente publizieren (Content Provider), relevante Peers empfehlen können (Recommender) oder gut vernetzt sind (Bootstrapping Peers).

3. Um ein Überfluten des Netzwerkes zu vermeiden, wird eine Anfrage nur an die 'besten' lokal bekannten Peers weitergeleitet. Es ist nur nötig, die besten Peers für ein Thema im Index zu verwalten.

Um zu entscheiden, welcher Shortcut nicht mehr im Index verbleibt, ordnen wir lokal die Content und Recommender Shortcuts. Dazu berechnen wir für jeden Shortcut eine Relevanz. Die Shortcuts mit der geringsten Relevanz werden gelöscht.

Strategie und Ziel. Unsere Strategie basiert darauf, das ein Peer der in der Vergangenheit viele unserer Anfragen erfolgreich beantworten konnte, auch in der Zukunft über passende Shortcuts bzw. Dokumente für unsere Anfragen verfügt. Weiterhin berücksichtigen wir die semantische Nähe der uns bekannten Anfragen eines entfernten Peers zu unseren eigenen Anfragen. Schließlich berücksichtigen wir noch die wechselnde Interessen der Peers. Dazu führen wir folgende drei Lokalitäten ein:

Community-Lokalität. Wir untersuchen, wie nahe ein lokal gespeicherter Recommender oder Content Provider Shortcut uns zu dem gewünschten Dokument führt und welches Wissen wir über die Erstellung des Shortcuts haben. Dazu führen wir einen Wichtungsfaktor $type$ für den Shortcut Typ ein. Content Provider Shortcuts, die im Index mit einem *'c'* markiert sind, benötigen einen Hop um eine Anfrage zu einem Peer weiter zuleiten, der ein passendes Dokument speichert. Wir vermuten weiterhin, das diese Peers besonders an diesem Thema interessiert sind und das auch ihre Bereitschaft Plattenplatz und Bandbreite für bestimmte Themen zur Verfügung zu stellen höher ist, als bei anderen Peers, wir setzen $type = 1$. Bei einem Recommender Shortcut beträgt die Entfernung zu einem Dokument zwei Hops. Zusätzlich wissen wir bei aktiven Recommender Shortcuts nicht, von welchem Typ der Shortcut vom Recommender zum Content Provider ist, beispielsweise kann der Recommender den Shortcut nur zufällig aufgebaut haben und nicht auf der Basis eigener Anfragen. Wir setzen $type = 0.5$

Zeitliche Lokalität. Interessen eines Peers können 'langfristig' wechseln [All96]. Wir verwenden eine *Least Recently Used (LRU)* Strategie, die bereits im Buffer Management von Datenbanken [ADU71] und im Cache Management für Webbrowser [MT02] erfolgreich zur Modellierung der Interessen der Nutzer und der Popularität verfügbarer Objekte verwendet wird. Sie ersetzt die Objekte im Cache, die am längsten nicht benutzt wurden. Analog verfahren wir mit Shortcuts im Index: Jeder Shortcut verfügt über das Attribut *Update*, das den Zeitpunkt seiner Erzeugung beschreibt. Wird ein Shortcut erfolgreich von einem lokalen Peer benutzt, so aktualisieren wir *Update* mit der aktuellen Zeit. Dadurch werden nicht benutze Shortcuts gelöscht, während häufig benutze Shortcuts im Index verbleiben. Wir berechnen die Relevanz eines Shortcuts sc abhängig von der Zeit entsprechend Gleichung 6.1:

$$update_{norm}(sc) = 1 - \frac{CurrentTime - Update(sc)}{CurrentTime - OldestUpdate} \tag{6.1}$$

wobei $CurrentTime$ die aktuelle Zeit, $Update(sc)$ den Zeitpunkt der Aktualisierung für einen Shortcut sc und $OldestUpdate$ den Zeitpunkt der Aktualisierung für den Shortcut mit der längsten Differenz zwischen seiner letzten Aktualisierung und der aktuellen Zeit bezeichnet.

Beispiel 6.3.1 (Zeitliche Lokalität der Shortcuts). Wir zeigen die Aktualisierung von $update_{normalized}$ am Beispiel eines Index mit zwei Content Provider Shortcuts. Zu Vereinfachung nehmen wir an, das alle Shortcuts in der selben Stunde erstellt wurden:

Shortcut	**PID**	**QH**	**Type**	**Update**	$update_{norm}$
/Education/UML	4	23	c	210405103800	$1 - \frac{39-38}{39-12}$=0.963
/Education/UML	6	20	c	210405101200	$1 - \frac{39-12}{39-12}$=0

Tabelle 6.1: Berechnung von $update_{norm}$ zum Zeitpunkt 210405103900

Semantische Lokalität. Ein Peer speichert in seinem Index alle passiven Recommender Shortcuts, die er durch Anfragen entfernter Peers kennenlernt. Durch diese Strategie lernt ein Peer schnell neue Peers über deren Anfragen kennen. Häufig stimmt jedoch nur ein kleiner Teil der Shortcuts mit den Interessen des eigenen lokalen Peers überein. Wir untersuchen die semantische Nähe passive Recommender Shortcuts zu unserem eigenen Interessen-Profil. In dieser Arbeit nehmen wir an, das ein solches Profil bereits gegeben ist. Alternativ kann das Interessen-Profil aus den Anfragen des lokalen Nutzers modelliert, aus den publizierten Dokumenten abgeleitet oder durch den Nutzer durch Metadaten manuell beschrieben werden. Zur Messung der semantischen Ähnlichkeit verwenden wir Gleichung 6.2.1. Wir vergleichen jeden Shortcut mit dem Interessenprofil und speichern die maximal erreichte Ähnlichkeit in $maxsim(sc)$. Shortcuts, die mit unseren Interessen ähnlich oder gleich sind, verbleiben im Index.

Wir vermuten, das die Lokalitäten unterschiedlich stark Einfluss auf den Rang eines Shortcuts im Index ausüben und gewichten jede der Lokalitäten mit einem Faktor. Gleichung 6.2 beschreibt die Relevanz $relevance(sc) \in [0, 1]$ eines Shortcuts im Index. Zur Gewichtung der unterschiedlichen Lokalitäten verwenden wir den *Weighted moving average*.

$$relevance(sc) = \frac{a * maxsim(sc) + +b * update_{norm}(sc) + c * type(sc)}{a + b + c} \quad (6.2)$$

In Abschnitt 7.3 untersuchen wir den Einfluss der Gewichtungsfaktoren auf die Effizienz des Netzwerkes.

Algorithmus. Algorithmus 6 wird aufgerufen, wenn ein neuer Recommender oder Content Provider Shortcut erstellt wird. Zuerst fügen wir den neuen Shortcut dem Index hinzu und berechnen seinen Relevanz entsprechend Gleichung 6.2. Hat der Index mit dem Shortcut seine maximale Größe überschritten(Zeile 1) prüfen wir, ob der Shortcut sc eine höheren Relevanz aufweist, als der Shortcut $lowest\_sc$ im Index mit dem geringsten Relevanz (Zeile 3). Ist das der Fall, löschen wir $lowest\_sc$ und fügen sc in den Index ein (Zeile 4,5).

Algorithm 6 InsertQueryDependentShortcut

Require: Shortcut *sc*, LocalIndex *index*, **int** maxIndexSize
1: **if** *index*.getSize() = *maxIndexSize* **then**
2: Shortcut lowest_sc=*index*.getShortcutWithLowestRelevance()
3: **if** (relevance(lowest_sc)<relevance(sc)) **then**
4: *index*.remove(lowest_sc)
5: *index*.insert(sc,relevance(sc))
6: **end if**
7: **end if**

6.4 Verwandte Arbeiten

Ähnliche Arbeiten zu Index- und Routing Strategien wurden im Bereich der Datenbanken, der Analyse sozialer Netzwerke und im Bereich der Suche in Peer-to-Peer Netzwerken durchgeführt.

6.4.1 Index Management

Folgende Ansätze beinhalten Index Strategien, die ein Clustern von Peers auf der Basis gemeinsamer Interessen unterstützen:

- Die Autoren in [SMZ03] verwenden eine Metrik zur Messung des Erfolgs von Content Provider Shortcuts. Die Metrik definiert die *success rate* als das Verhältnis zwischen der Anzahl der erfolgreichen Versuche und der Anzahl aller Versuche über einen Shortcut eine Anfrage zu leiten. Der Ansatz gewichtet stark bereits gefundene Shortcuts und gewichtet schwach die zu einem späteren Zeitpunkt gefundene Shortcuts.

- Die Autoren von [VKMvS04] untersuchen in ihren Arbeiten neben der LRU Strategie zwei weitere Verfahren für das Index Management von Content Provider Shortcuts:

History Based Indexing. Dieser Ansatz bewertet Peers höher, die gleiche Interessen wie der anfragende Peer aufweisen. Dazu enthält der Index alle Peers mit denen ein lokaler Peer in Kontakt getreten ist. Jeder Peer verfügt zusätzlich über einen Zähler. Kann ein Peer eine Anfrage erfolgreich beantworten, so wird der Zähler erhöht. Der Ansatz hat zwei Nachteile: Erstens kann der Index kann beliebig gross werden. Zweitens, passt sich der Index nur langsam an neue Interessen der Nutzer an. Die Autoren verwenden den Ansatz als Vergleichswert in ihren Arbeiten.

Popularity based Indexing. Dieser Ansatz bewertet, wie häufig ein Peer innerhalb eines Zeitintervalls erfolgreich eine Anfrage beantworten konnte. Peers verbleiben im Index, die innerhalb eines Zeitintervalls die meisten Anfragen beantworten konnten. Dadurch werden Peers höher bewertet, die populäre Anfragen beantworten konnten. Leider geben die Autoren keine Hinweise für die Wahl des Zeitintervalls an.

Die Ergebnisse der Autoren zeigen, das bei allen drei Ansätzen Peers mit gleichen Interessen im Index des anfragenden Peers gespeichert werden. Unterschiede ergeben sich in der Anzahl der Verwendung des Index, die bei der 'History' Strategie am höchsten ist.

- Die Autoren in [CFB04] vergleichen die LRU Strategie mit der *Most Often Used (MOU)* Strategie. Peers werden höher bewertet, die entweder eine Anfrage beantwortet haben oder die eine Anfrage zu einem antwortenden Peer weitergeleitet haben. Jeder Peer im Pfad zum antwortenden Peer erhält einen Wert. Der antwortende Peer erhält den höchsten Wert, für nachfolgende Peers sinkt der Wert exponentiell. Beide Strategien benötigen ungefähr die gleiche Anzahl von Hops. Die Vorteile der MOU Strategie liegen in der Bildung einer stabileren Netzwerk Struktur nach einer geringen Anzahl von Anfragen.

6.4.2 Routing Ansätze

Soziale Netzwerke. Suchstrategien für soziale Netzwerke wurden kürzlich von verschiedenen Autoren für Peer-to-Peer Systeme untersucht. Die Autoren von [CFB04] schlagen eine Strategie vor, die eine hohe Überlappung in ihren Anfragen haben und einem gemeinsamen Cluster zugeordnet werden. Zur Speicherung der erfolgreichen Anfragen analysieren sie verschiedene Indexstrategien. Im Gegensatz zu INGA berücksichtigen sie für die Bildung der Cluster keine Themen und keine semantischen Beziehungen zwischen

den Themen. Die Autoren von [TSW04] untersuchen Strategien für das semantische Weiterleiten von Anfragen. Peers und Anfragen sind durch eine Themen-Hierarchie klassifiziert. Die Autoren verwenden jedoch nur Content Provider Shortcuts, die in einem unbegrenzten gespeichert werden und optimieren den Ansatz nicht in einem dynamischen Netzwerk. In [AA04] untersuchen die Autoren verschiedene Strategien für die Suche in sozialen Netzwerken, wie dem E-Mail Verkehr in einem Forschungslabor. Sie zeigen, das eine High Degree Strategie sich bedingt eignet und mit einer Greedy Strategie effizient eine Suche erfolgt. Sie kombinieren nicht die Strategien und zeigen keine Strategien für das Index Management und die Bildung von Shortcuts auf.

Gossiping. *Gossiping* ist ein Verfahren, um in einem Netzwerk eine Information an alle Peers eines Netzwerkes zu verschicken. Hierzu werden Nachrichten an (zufällig) ermittelte Adressaten gesandt [DGH$^+$87]. Im Gegensatz zu einer Suche in einem Peer-to-Peer Netzwerk liegt der Schwerpunkt auf einer effizienten Verteilung von Informationen an möglichst alle Peers, ohne das eine zentrale Instanz vorliegt und mit einer möglichst gleichmäßiger Belastung aller Peers [CAPMN02, GBL$^+$03].

Die Autoren von [ACMH02] bezeichnen *Semantic Gossiping* als das Erlernen von Shortcuts zwischen entfernten Peers mit ähnlichen Schemata in Mapping Netzwerken. Zur Messung der Ähnlichkeit analysieren sie die syntaktische Ähnlichkeit einer Anfrage, nachdem sie über mehrere Peers weitergeleitet wurde. Zur Feststellung der semantische Ähnlichkeit nutzen die Autoren Zyklen in Mapping-Netzwerken und untersuchen den Informationsverlust mit Hilfe der Maximum Likely-Hood Approximation. Zusätzlich wird überprüft, ob die Resultate mit semantisch ähnlichen Metadaten annotiert wurden. Die Autoren zeigen in Versuchen, das ein Peer nach einer Lernphase des Systems nur noch Anfragen an Peers stellt, die eine hohe semantische Ähnlichkeit aufweisen. Wir vermuten, das der Semantic Gossiping Ansatz mit dem INGA Ansatz kombiniert werden kann. Ein gemeinsamer Ansatz profitiert vom INGA Netzwerk von Konzepten zur Erstellung von Shortcuts und der dynamischen Auswahl von Routingstrategien und vom Semantic Gossiping Ansatz durch die Überbrückung semantischer Heterogenitäten durch die Analyse der Anfrage und Resultate. Wir erwarten, das die Community sich diesem Thema in ihrer zukünftigen Forschung annehmen wird.

6.5 Zusammenfassung

Dieses Kapitel stellt drei Beiträge vor: Ein Ablaufmodell für die Interaktionen eines Peers in einem Shortcut Netzwerk, Index Strategien für die Aktualisierung bereits registrierter Shortcuts und die Kombination von Routing Strategien abhängig von den bereits an einem Peer registrierten Shortcuts.

Wir erörtern den Zusammenhang zwischen Index- und Routing Strategien in unserem Ablaufmodell. Durch jede von einem Peer selbst gestellte oder weitergeleitete Anfrage lernt der Peer neue Shortcuts zu anderen Peers kennen. Da der Shortcut Index an einem Peer begrenzt ist, werden beim Registrieren neuer Shortcuts bereits existierende Shortcuts zu Peers gelöscht, die in der Vergangenheit nur wenige Dokumente für eine Anfrage geliefert haben oder in der Vergangenheit nur auf wenige Anfragen antworten konnten oder zu Peers, die schlechter vernetzt sind.

Ausgehend von den registrierten Shortcuts im Index wählt ein Peer die besten Shortcuts für eine Anfrage aus. Wir präsentieren einen Algorithmus, der zwischen einer Fireworks, Greedy und High Out-Degree Strategie entscheidet. Zusätzlich berücksichtigt der Algorithmus in einer Serendipity Routing Strategie zufällig gewählte Peers, damit eine Anfrage auch an neu im Netzwerk registrierte Peers gesendet werden kann.

7 Simulation und Evaluierung

Zur empirischen Auswertung der Algorithmen simulieren wir in einem rundenbasierten Simulator das Verhalten von INGA . Als realen Datensatz für die Interessen-Profile der Nutzer verwenden wir das Open Directory (DMOZ). Wir untersuchen das Verhalten der einzelnen Shortcut Overlays und der Parameter des Index in einem statischen und dynamischen Netzwerk und vergleichen INGA gegenüber der naiven Strategie von Gnutella und einem aktuellen Ansatz auf der Basis von Shortcuts.

7.1 Modellierung des Systems

In diesem Abschnitt stellen wir kurz die Eigenschaften unser Simulationsumgebung, wie die Verteilung und Dynamik der Themen und Anfragen, sowie den Aufbau und die Dynamik des Default Netzwerkes, vor.

7.1.1 Initiale Struktur des Netzwerkes

Ähnlich wie Gnutella modellieren wir das *Default Netzwerk* (siehe Abschnitt 4.2.3), indem wir die Peers in einem Overlay Netzwerk anordnen. Jeder Peer verfügt über eine Menge von Nachbarn, mit denen er Nachrichten austauschen kann. Die Verbindungen eines Peers zu seinen Nachbarn sind uni-direktional. Aufgrund der Ausrichtung der Arbeit auf die Analyse von Anfragen, die gegenüber den eigentlichen Dokumenten ein geringes Volumen aufweisen, berücksichtigen wir nicht

die verfügbare Bandbreite für eine Verbindung bzw. die Netzwerk Verzögerung (Latency). Stattdessen beschränken wir uns auf das Messen der Anzahl der Nachrichten für eine Anfrage und der Anzahl der dafür benötigten Hops.

Zu Beginn einer Simulation verfügt jeder Peer nur über die Nachbarn des Default Networks, die wir für die Simulation eines statischen Netzwerkes nicht verändern. Verschiedene Studien [IRF04, ALPH01] haben gezeigt, das auch die initiale Verteilung der Verbindungen zwischen Peers in File Sharing Netzwerken einer Power Law Verteilung folgt und Small World Eigenschaften aufweist. In der Simulation verwenden wir zur Generierung des Default Netzwerkes einen Small World Generator aus dem *Java Universal Network/Graph Framework (JUNG)* [1]. Er generiert ein Small World Netzwerk entsprechend dem Modell von Kleinberg [Kle00a], bei der jeder Peer vier 'Short Range' Verbindungen und einer 'Long Range' Verbindung aufweist. Die Wahrscheinlichkeit einer solchen Verbindung wird durch den Clustering Exponent α ausgedrückt, in unseren Versuchen verwenden wir $\alpha = 2.1$. Wir generieren 1024 Peers, von denen jeder über genau fünf ausgehende Verbindungen zu benachbarten Peers und 'unbegrenzt' eingehende Verbindungen verfügt.

7.1.2 Verteilung der Anfragen und Themen

Analog zu [SCK03, CGM02b, CFB04] nehmen wir an, das jeder Peer nur an bestimmten Themen ein Interesse aufweist. Zur möglichst realistischen Modellierung der Anfragen und Themen pro Peer verwenden wir das *Open Directory (ODP)*[DMO]. Es verfügt über eine Sammlung und Kategorisierung von ca. fünf Millionen Webseiten in ca. 500.000 Kategorien, die Kategorisierung erfolgt durch ca. 50.000 menschliche Autoren.[2] Wir verwenden für unsere Simulation einen Ausschnitt des Open Directories mit 1648 Kategorien der oberen fünf Hierarchie-Ebenen. Aus den 1624 unterschiedlichen Autoren wählen wir zufällig 1024 Autoren aus, die jeweils einen Peer repräsentieren. Für unsere Simulation weist das ODP folgende interessante Eigenschaften auf:

- *Semantische Beziehungen.* Das ODP enthält ein kontrolliertes Vokabular, d.h. eindeutige Benennungen (Deskriptoren über eine Pfad-Angabe) für jeden Thema. Themen stehen in hierarchischen Relationen zueinander in Bezug. Landesspezifische Schreibweisen, Synonyme bzw. als gleichbedeutend behandelte Quasi-Synonyme, Abkürzungen, Übersetzungen etc. werden durch Äquivalenzrelationen miteinander in Beziehung gesetzt. Themen werden außerdem durch Assoziationsrelationen und hierarchische Relationen vernetzt.

[1] Java Klasse: edu.uci.ics.jung.random.generators.KleinbergSmallWorldGenerator
[2] Stand April 2004

Autoren	1	2	3	4	5	6	7	8	9	10	11	12	13	14	20	25	Σ
Themen	1217	274	80	30	14	10	3	8	1	2	2	1	1	1	1	1	1648

Abbildung 7.1: Verteilung der Autoren über die Themen (Popularität der Themen).

Themen	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	18	20	22	Σ
Autoren	991	295	128	96	45	21	18	9	5	6	3	1	1	1	1	1	1	1	1624

Abbildung 7.2: Verteilung der Themen über die Autoren

Für diese Arbeit verwenden wir ausschließlich die hierarchische Relationen der Form Generalisierung/Spezialiserung.

- *Zuordnung von Interessen zu Personen und Gruppen.* Die Zuweisung von Dokumenten zu Kategorien erfolgt durch menschliche Autoren, die über einen eindeutigen Schlüssel identifiziert werden können. Wir modellieren die Interessen eines Peers, indem wir die Themen eines Autors einem Peer gleichsetzen.

Verteilung der Themen und Autoren. Die Popularität einzelner Themen ist im Open Directory stark Zipf verteilt. Grafik 7.1 verdeutlicht diesen Zusammenhang: 1217 Themen werden jeweils von einem Autor bearbeitet, 274 von zwei Autoren ... 25 Autoren bearbeiten zusammen ein Thema. Ähnlich verhält sich die Anzahl der Themen, die pro Autor bearbeitet werden, siehe auch Grafik 7.2: 991 Autoren bearbeiten genau ein Thema, 295 Autoren bearbeiten zwei Themen, 128 Autoren drei Themen ... ein Autor bearbeitet 22 Themen.

Ähnlich den Projektgruppen in einem Unternehmen entstehen Gruppen von Peers, die sich gemeinsam für ein Thema interessieren. Die maximale Anzahl der Teilnehmer einer Gruppe beträgt 24 Peers. Vergleichbar den Sachbearbeitern in einem Unternehmen ist der Mehrzahl der Themen genau einem Peer zugeordnet. Ähnlich verhält es sich mit der Anzahl der Themen an einem Peer. Wenige Peers interessieren sich für viele Themen, währen die Mehrzahl der Peers sich für genau ein Thema interessiert.

Verteilung der Anfragen. Im INGA Netzwerk werden Recommender Shortcuts und der Bootstrapping Index in Abhängigkeit von den Anfragen aufgebaut und verwendet. Um die Leistungsfähigkeit der Algorithmen besser beurteilen zu können, untersuchen wir das Verhalten bei einer Gleichverteilung von Anfragen über alle Themen. Für die Indexing- und Routing Strategien ist eine gleiche Verteilung der Popularität aller Themen besonders ungünstig, da Anfragen für ein Thema sich nur selten wiederholen und bereits erstellte Shortcuts im Gegensatz zu einer Zipf

Verteilung, besonders wenig für Anfragen von anderen Peers verwendet werden können.

Eine weitere Verteilung der Anfragen wird in [VKMvS04] vorgeschlagen. Ein Peer fragt überwiegend Themen an, für die er sich selbst interessiert. Zusätzlich fragt er zu einem kleineren Anteil zufällige Themen. Mit dem vorliegenden Datensatz ergibt eine solche Verteilung einen unrealistisch hohen Recall, da für viele Themen genau nur der anfragende Peer Dokumente publiziert. Wir haben auf die Darstellung dieses Szenarios verzichtet.

7.1.3 Modellierung von Dynamik

Verschiedene Studien [SGG03, CLL04, Mar02] zeigen, das Peer-to-Peer Netzwerke eine hohe Volatilität aufweisen. Überraschenderweise gibt es jedoch bisher keine Untersuchungen des Verhaltens von Shortcut-Ansätzen in einem dynamischen Netzwerk. Zur Simulation eines dynamischen Netzwerkes haben wir die Verfügbarkeit der Peers auf der Basis der Studie von [SGG03] implementiert. Die Studie untersucht das Verhalten von Peers in einem Gnutella und Napster Netzwerk über einen Zeitraum von zwölf Stunden. Während dieser Zeit beträgt die durchschnittliche Verfügbarkeit eines Peers im Netzwerk ca. 60 Minuten. Ein Peer startet alle 240 Sekunden eine Anfrage, was nach [LCC+02] einer *QueryJoinRatio* von ca 15 entspricht. In unserer Simulation zeigen wir einen Zeitraum von 30 Anfragen pro Peer, was einer Simulation von zwei Stunden entspricht. Grafik 7.3 zeigt die kummulierte Verteilung (CDF) der Verfügbarkeit für diesen Zeitraum. Das Netzwerk ist durch eine hohe Anzahl von relativ dynamisch verfügbaren Peers gekennzeichnet, 40% der Peers weisen eine maximale Verfügbarkeit von ca. 28% auf. Aufgrund der durchschnittlichen Verfügbarkeit von einer Stunde weisen 26% der Peers eine Verfügbarkeit von mehr als 90% auf. Wird ein längerer Zeitraum als 30 Anfragen oder zwei Stunden betrachtet, verringert sich der Anteil der Peers, die eine hohe Verfügbarkeit aufweisen, weiter.

7.2 Methodik der Simulation

Wir simulieren das Verhalten der Peers auf der Basis eines runden-basierten Models [SCK03, CFB04]. Tabelle 7.4 zeigt die Parameter der Simulation. Für jede Runde werden 42 Peers zufällig ausgewählt. Um eine Anfrage zu formulieren, wird für jeden gewählten Peer das Thema der Anfrage bestimmt. Analog zu Gnutella sendet der Peer die Anfrage an k Nachbarn mit $k = 2$ über bereits existierende Shortcuts mit einer begrenzten TTL. Erhält ein Peer eine Anfrage zum ersten Mal, vermindern wir die TTL der Anfrage und leiten die Anfrage an dessen ausgewählte

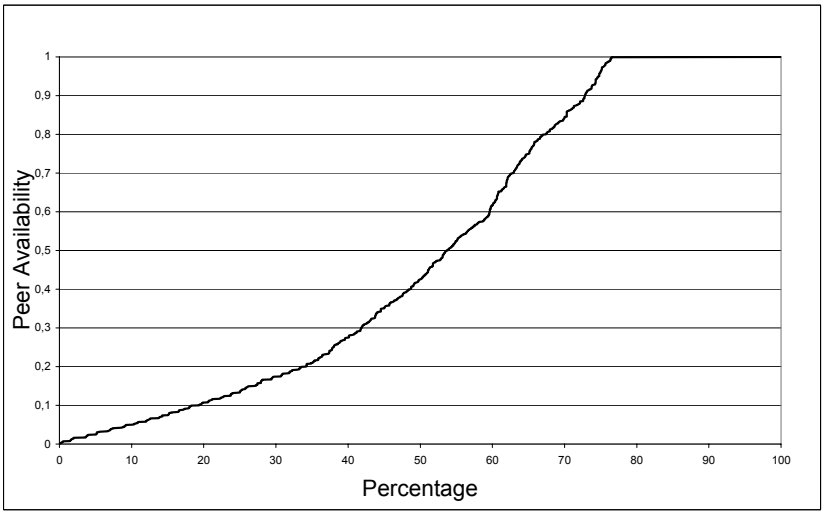


Abbildung 7.3: Verteilung der Verfügbarkeit

Nachbarn weiter.

Die Auswahl der Shortcuts für eine Anfrage erfolgt dynamisch auf Basis der bereits im lokalen Index gespeicherten Shortcuts (siehe auch Algorithmus 5 in Abschnitt 6.2). Ist noch kein Shortcut im Index vorhanden, sendet ein Peer eine Anfrage, indem er aus den verfügbaren fünf benachbarten Peers des Default Network Layers zwei Peers zufällig auswählt. Ist der gewählte Peer bereits im Message Pfad der Anfrage enthalten, tauschen wir diese mit einem anderen der fünf Peers aus.

Wir simulieren insgesamt einen Zeitraum von 30.000 Anfragen über 715 Runden, die für insgesamt 1646 Themen gestellt werden. Um auch eine Dynamik der Interessen zu simulieren, wird den Peers in der ersten Phase eine Hälfte der Themen und in der zweiten Phase die zweiten Hälfte der Themen zugewiesen. Dadurch beobachten wir, in wieweit die Aktualisierung der Indizes im Netzwerk nach der Einführung neuer Themen erfolgt.

7.2.1 Metriken

Durch die Simulation wollen wir Erkenntnisse sammeln, wie sich INGA im Vergleich zu anderen Ansätzen verhält, welchen Einfluss die Parameter für die Ge-

Parameter	Wert
Simulierte Peers	1024
Durchläufe der Simulation	6
Auswahl Peers Pro Runde	42
Simulierte Runden	715
Themen	1646
Erste Phase	823
Zweite Phase	823
Anfragen	30.000
Anfragen pro Peer (gleichverteilt)	29
TTL der Anfrage	6
Ausgewählte Peers pro Anfrage Peers (k)	2
Schranke für Greedy Suche (t_{Greedy})	15 %
Zufällige Shortcuts (f)	20 %
Maximale Anzahl von Shortcuts im Index	40

Abbildung 7.4: In der Simulation verwendete Parameter

wichtung des Index und dessen Größe und welchen Beitrag die einzelnen Layer haben. Zur Messung der der Effizienz unserer Algorithmen verwenden wir folgende Metriken:

- **Query Recall.** Wir verwenden den Recall um die Leistungsfähigkeit der Algorithmen zu messen, z. B., in welchem Mass ein Peer Dokumente nur auf der Basis von lokalem Wissen über andere Peers finden kann.

  Definition 7.2.1 (Query Recall). *Der Query Recall definiert das Verhältnis der für die Anfrage gefundenen, relevanten Dokumente gegenüber der Anzahl aller für die Anfrage relevanten Dokumente.* ♣

- **Messages.** Wir repräsentieren die benötigten Kosten durch die Anzahl der dafür benötigten Nachrichten. Die Anzahl der Nachrichten definiert ebenfalls indirekt ein Mass zur Ableitung der Skalierbarkeit des Systems.

  Definition 7.2.2 (Messages). *Wir zählen alle* Query Messages *(siehe Definition 4.3.2) und alle* Result Messages *(siehe Definition 4.3.3). Deren Summe bestimmt die Kosten für die Anfrage.* ♣

- **Message Gain.** Ähnlich dem Return on Investment (ROI) führen wir den *Message Gain* ein. Er beschreibt das Verhältnis zwischen dem Recall für eine Anfrage und der dafür benötigen Anzahl von Nachrichten.

Definition 7.2.3 (Message Gain). *Wir definieren den Message Gain als den Anteil der benötigten Nachrichten einer Anfrage im Verhältnis zum erreichten Recall.*

$$MessageGain = \frac{\#gefundene\ relevante Dokumente}{\#alle\ relevanten Dokumente \times \#Messages} \quad (7.1)$$

. ♣

- **Average Path Length.** Analog zu Definition 4.2.1 messen wir die durchschnittlich kürzeste Entfernung zwischen zwei Peers im Shortcut Overlay Netzwerk. Eine kurze *Average Path Length* bedeutet einen schnellen, möglichst direkten Informationsfluss. Sie ermöglicht ein gezieltes Verteilen einer Anfrage mit einer geringen Anzahl von Hops, und damit geringere Netzwerkkosten, in einer geringen Zeit.

  Definition 7.2.4 (Average Path Length). *Sei $\mathcal{G}$= (SC,P) ein Graph, der ein Shortcut Netzwerk repräsentiert. In $\mathcal{G}$ gibt es N Peers P mit $|P| = N$, die über Shortcuts SC miteinander verbunden sind. $D(i,j)$ repräsentiert die Länge in Hops des* kürzesten Pfades *zwischen zwei Peers i und j mit $i,j \in P$. Die durchschnittliche kürzeste Anzahl von Hops in $\mathcal{G}$, bezeichnet als $AveragePathLength(\mathcal{G})$, wird definiert als:*

$$AveragePathLength(\mathcal{G}) = \frac{1}{\binom{N}{2}} \sum_{i \neq j} D(i,j) \quad (7.2)$$ ♣

7.2.2 Implementierung ähnlicher Systeme

In unseren Experimenten wollen wir uns ebenfalls mit existierenden Ansätzen vergleichen. Wir haben in der Simulation folgende zwei Strategien für das Routing implementiert:

Gnutella. Die Simulation verhält sich analog der Flutungsstrategie von Gnutella 0.4 [Gnu04]. Jede Anfrage enthält die maximale Anzahl der Hops, die sie weitergeleitet werden kann. Eine Anfrage wird an die, durch das Default Netzwerk definierten, zufällig ausgesuchten Nachbarn weitergeleitet. Die Simulation beinhaltet ebenfalls die *Gnutella Duplicate Detection* zur Erkennung von Peers, die bereits die Anfrage erhalten haben, so das diese die Anfrage nicht erneut erhalten. Dazu wird einerseits der *Query Message Path* analysiert und es werden in einem lokalem Cache die letzten Anfragen zwischengespeichert.

Interest-based Locality (IBL). Die Autoren von [SMZ03] verwenden ebenfalls das Konzept der *Interest-based locality* um logische Overlays und Interessenbasierte Shortcuts zu erzeugen. Ein Shortcut wird nach jeder erfolgreichen Anfrage an einem Peer lokal gespeichert. Das Konzept ist analog zu *Content Provider Shortcuts*. Die IBL-Suchstrategie unterscheidet sich jedoch von INGA , da ein Shortcut nur für das Routing ausgewählt wird, wenn dieser exakt mit der Anfrage übereinstimmt. Andernfalls wird die Nachricht im Netzwerk mit dem naiven Ansatz weitergeleitet. Als Index Strategie verwendet der Ansatz eine LRU Strategie [ADU71].

7.3 Evaluierung

7.3.1 Verhalten in einem statischen Netzwerk

Zur Simulation von *Gnutella*, *IBL* und INGA verwenden wir die in Tabelle 7.4 aufgeführten Parameter. Wir simulieren zuerst ein statisches Netzwerk, bei dem die Anfragen gleich verteilt sind.

Recall. Wir beobachten in Grafik 7.5 das INGA einen höheren Recall als die beiden anderen Ansätze aufweist. Während der Recall der naiven Strategie weitestgehend konstant bleibt, steigt der Recall für *IBL* und für INGA , da ein Peer Informationen über anderen Peers in seinem lokalen Index speichert und für weitere Anfragen verwendet. Der gesteigerte Recall von INGA gegenüber *IBL* kann darauf zurückgeführt werden, das einerseits mehr Shortcuts innerhalb kürzerer Zeit gespeichert werden und durch die dynamische Auswahl der Routing Strategie deutlich weniger auf den Default Network Layer zurückgegriffen werden muss. Nachdem zur Hälfte der Simulation die Themen ausgetauscht werden, sinkt der Recall bei beiden Ansätzen, da die bereits gefunden Shortcuts nicht mehr den neuen Themen eines Peers entsprechen. Während der Recall von *IBL* fast auf das Niveau des naiven Ansatzes zurückfällt, profitiert INGA vom Einsatz der Bootstrapping Peers.

Average Path Length. Grafik 7.6 zeigt die durchschnittliche Anzahl von Hops zwischen zwei beliebigen Peers. Während der naive Ansatz eine statische Netzwerk Struktur besitzt und über eine konstante Anzahl von ca. sieben Hops verfügt, verringert sich bei den beiden Shortcut Ansätzen die Anzahl der Hops mit der Anzahl der im Cache registrierten Shortcuts und stabilisiert sich dann. Durch INGAs vier verschiedenen Möglichkeiten, Shortcuts (Content, Aktive und passive Recommender, Bootstrapping und Default Shortcuts) im lokalen Index zu registrieren, werden einerseits deutlich *schneller*

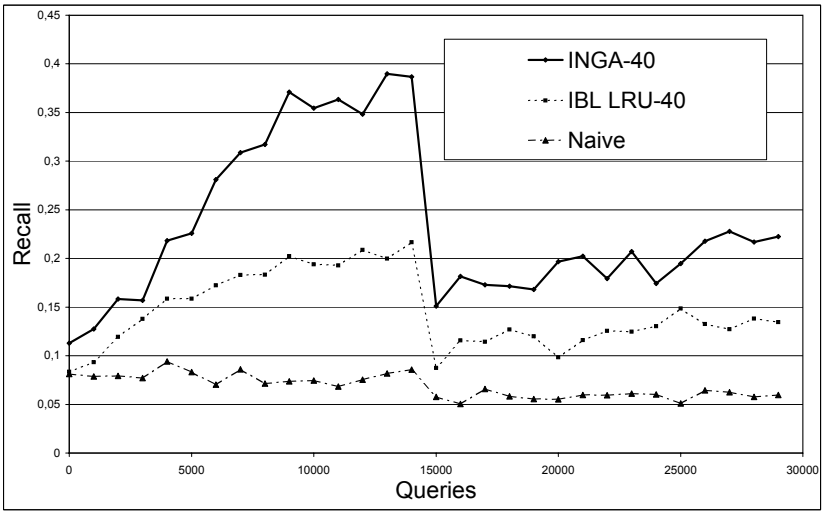


Abbildung 7.5: Recall: Statisches Netzwerk

und auch deutlich *mehr* Shortcuts zu unterschiedlichen Peers gewonnen, als im Ansatz von *IBL*. Das ist darauf zurückzuführen, das die Anzahl der benachbarten Peers in INGA größer ist als in *IBL*. Sind die bei *IBL* ausgewählten besten Peers bereits im Message Pfad der Anfrage enthalten, sendet der Ansatz die Anfrage immer an die gleichen Nachbarn. Im Gegensatz sucht INGA in diesem Fall zusätzlich Peers aus dem Recommender Layer und dem Bootstrapping Layer aus. Dadurch stabilisiert sich die Anzahl der Hops in INGA nach bereits fünf Anfragen pro Peer auf durchschnittlich 35% und in *IBL* nach 15 Anfragen auf ca 55 % des des naiven Ansatzes.

Messages. Die Verringerung der Pfadlänge lässt jedoch nur indirekte Rückschlüsse auf die Anzahl der Nachrichten pro Anfrage zu. Die Anzahl der Nachrichten ist bestimmt durch die Anzahl k der Peers an die eine Anfrage lokal weitergeleitet wird und durch die TTL der Nachricht. Beide Parameter sind in allen Ansätzen mit $k = 2$ und $TTL = 6$ identisch.

Die Anzahl der Nachrichten wird weiterhin bestimmt durch die entstehende Netzwerk Struktur. Stehen weniger als zwei Peers für die Auswahl zur Verfügung wird im naiven Ansatz die Nachricht nur an einen oder gar keinen

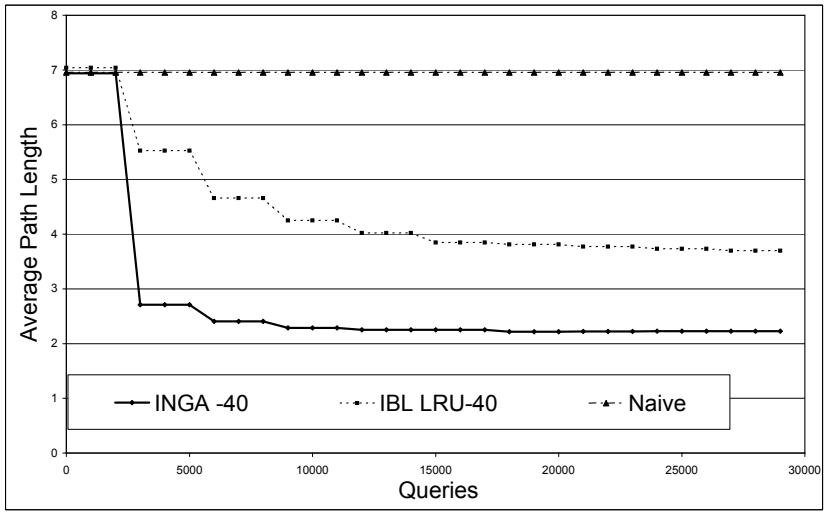


Abbildung 7.6: Kürzester Pfad: Statisches Netzwerk

Peer gesendet. In *IBL* wird eine Nachricht sowohl über den Content Layer als auch über den Default Layer weitergeleitet, wodurch potentiell eine höhere Wahrscheinlichkeit existiert, das zwei Peers für eine Nachricht ausgewählt werden können. In INGA wird diese Wahrscheinlichkeit durch den *Recommender Layer*, die *Similarity Selection* und den *Bootstrapping Layer* noch weiter erhöht. Bei beiden Shortcut Ansätzen steigt die Anzahl der Nachrichten zunächst über den naiven Ansatz an.

Grafik 7.7 zeigt das Verhalten der Nachrichten für alle drei Ansätze. Ein Weiterleiten der Nachricht an benachbarte Peers wird in allen Ansätzen begrenzt, da die Nachricht nicht an Peers weitergeleitet wird, die bereits im Pfad der *Query Message* enthalten sind und die Nachricht bereits erhalten haben. Die Wahrscheinlichkeit für einen noch nicht angefragten Peer steigt mit der Anzahl der im lokalen Index gespeicherten Shortcuts. Aufgrund der verschiedenen Strategien für die Erzeugung von Shortcuts werden im INGA Netzwerk schneller Shortcuts gewonnen. Das bedeutet, dass eine Anfrage mit einer hohen Wahrscheinlichkeit bereits zu den für die Anfrage relevanten Peers geleitet wurde. Der INGA Ansatz versucht in diesem Fall Boot-

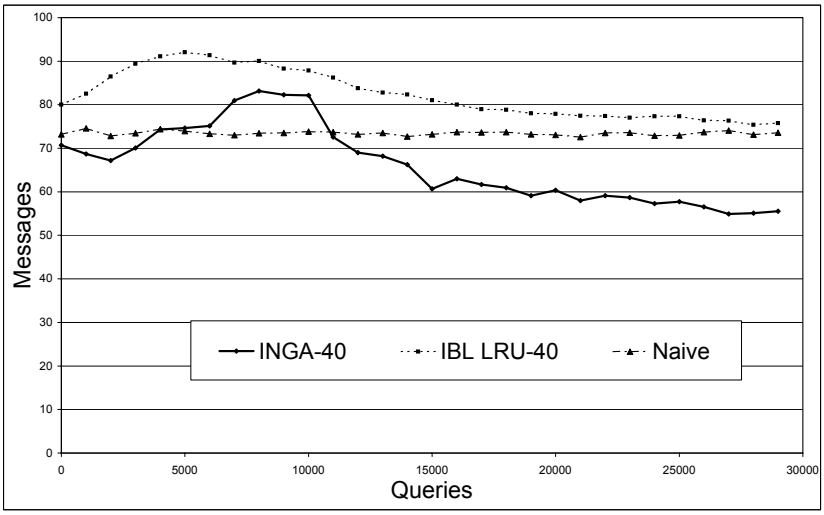


Abbildung 7.7: Nachrichten: Statisches Netzwerk

strapping Peers auszuwählen. Dadurch fokussieren sich die Anfragen auf eine kleine Anzahl von Bootstrapping Peers, wodurch die Nachrichten auf ca 75% des Niveaus der anderen beiden Ansätze reduziert werden.

Die Strategie von *IBL* erzeugt zunächst mehr Nachrichten, da die einzelnen Peers untereinander Shortcuts aufbauen müssen. Da jeder Peer Shortcuts zu verschiedenen Themen speichert, hat eine Anfrage eine geringe Wahrscheinlichkeit zu einem Peer weitergeleitet zu werden, der die Anfrage bereits erhalten hat. Nachdem jedoch genügend Shortcuts gesammelt wurden, stabilisiert sich die Anzahl der Nachrichten bei *IBL* auf das Niveau des naiven Ansatzes. Das bedeutet, dass nach ca. 10 Anfragen pro Peer INGA geringere Kosten für eine Anfrage benötigt als *IBL*.

Message Gain. INGA verringert die Anzahl der Nachrichten und ermöglicht einen deutlichen höheren Recall. Grafik 7.8 quantifiziert die Leistungsfähigkeit des Ansatzes gegenüber der naiven und der Strategie von *IBL* durch das Verhältnis der aufgewendeten Nachrichten für die Erhöhung des Recalls (Message Gain). Aufgrund der statischen Struktur des Netzwerkes bleibt der Message Gain der naiven Strategie über alle Anfragen auf dem gleichen Niveau,

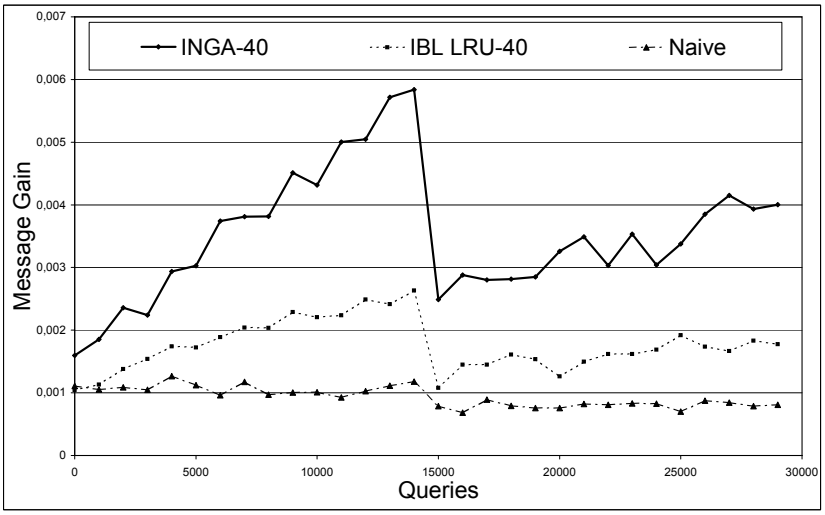


Abbildung 7.8: Message Gain: Statisches Netzwerk

während er für die beiden Ansätze mit Shortcuts steigt. Nach der Hälfte der Anfragen steigt INGA aufgrund der reduzierten Nachrichten und des gesteigerten Recalls auf das sechsfache Niveau der naiven Strategie, während *IBL* nur etwas mehr als das zweifache Niveau erreicht. Beide Ansätze fallen bei der Aktualisierung der Themen stark ab und beginnen neue Shortcuts zu sammeln. Dieser Prozess verhält sich ähnlich der ersten Phase, jedoch werden neue Shortcuts langsamer gewonnen, da nicht mehr passende Shortcuts lokal identifiziert und aus dem Index gelöscht werden müssen.

7.3.2 Verhalten in einem dynamischen Netzwerk

Analog zum statischen Netzwerk vergleichen wir INGA mit der Strategie von [SMZ03] und der naiven Strategie in einem dynamischen Netzwerk. Wir verwenden eine Index Größe von 40 Einträgen und eine LRU Strategie. Ebenfalls vergleichen wir uns mit dem naiven Ansatz. Wir verwenden die bereits für die statische Simulation beschriebenen Parameter aus Tabelle 7.4 und verteilen die Anfragen uniform.

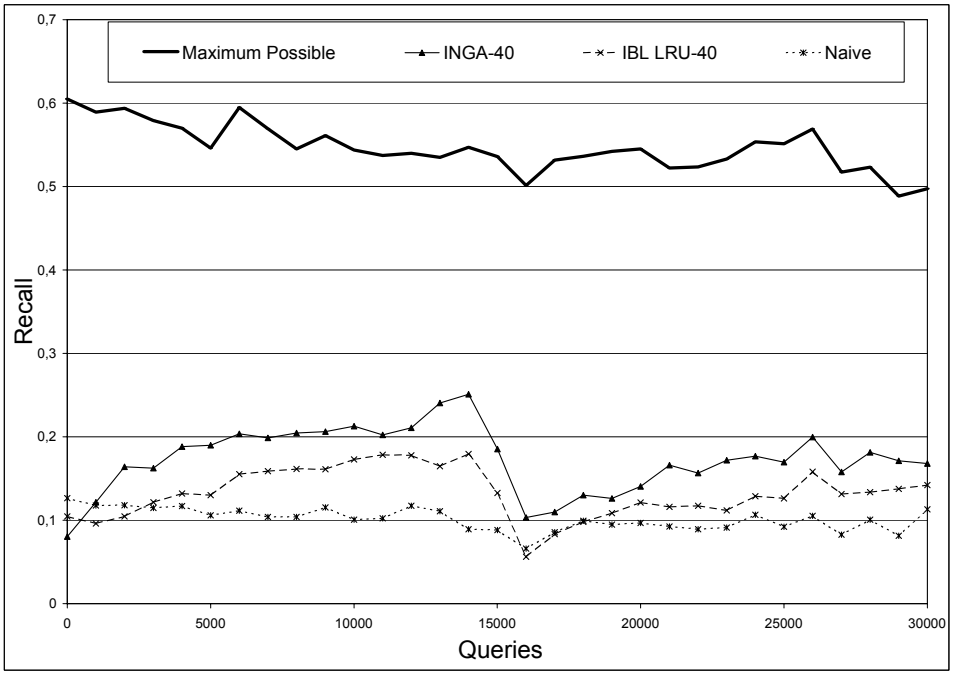


Abbildung 7.9: Recall: Dynamisches Netzwerk

Recall. In einem dynamischen Netzwerk ist der maximal erreichbare Recall abhängig von den zum Zeitpunkt der Anfrage verfügbaren Dokumenten. In Grafik 7.9 bezeichnet der Graph *Maximum Possible* den maximal erreichbaren Recall zum Zeitpunkt des Sendens einer Anfrage. Er liegt bei ca 55%. Während die naiven Strategie sich nach 15 Anfragen pro Peer auf einen Recall von ca. 10% stabilisiert, wächst bei *IBL LRU-40* der Recall auf 17% und bei INGA auf ca 25%. Der maximal erreichbare Recall zu diesem Zeitpunkt beträgt 54%. Nach Einführen neuer Anfragen für alle Peers sinkt *IBL LRU-40* auf 6% Recall ab, während INGA auf 10% abfällt. Zum Ende der zweiten Phase der Simulation erreicht INGA wieder einen Recall von ca. 18%, *IBL LRU-40* von ca 13% und die naive Strategie 11 %. Das Anwachsen des Recalls in der zweiten Phase der Simulation bei INGA wird durch den vollen Index zu Beginn der zweiten Phase gebremst. Im Gegensatz zu Beginn der ersten Phase muss jeder Peer in seinem Index unnötige Einträge für Shortcuts löschen bevor neue Shortcuts gewonnen werden können.

Nachrichten. In Grafik 7.10 können wir beobachten, das der naiven Ansatz sich bei 108 Nachrichten im dynamischen Netzwerk stabilisiert, während im sta-

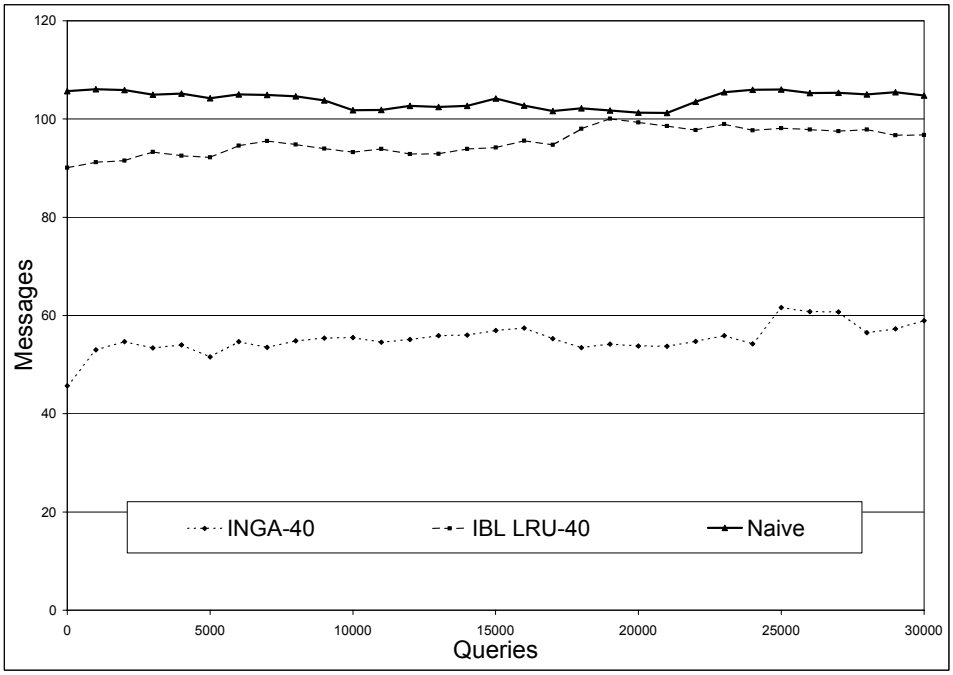


Abbildung 7.10: Nachrichten: Dynamisches Netzwerk

tischen Netzwerk (Grafik 7.7) nur 73 Nachrichten benötigt wurden. Analog dem statischen Netzwerk ist auch im dynamischen Netzwerk die Anzahl der Nachrichten durch die TTL und die Anzahl k der ausgewählten Nachbarn für das Weiterleiten einer Nachricht bestimmt. Für $TTL = 6$ und $k = 2$ ergibt sich eine theoretisch mögliche Anzahl von $\sum_{i=1}^{6} 2^i = 2^7 - 1 = 127$ Nachrichten. Ebenfalls analog zum statischen Ansatz wird eine Anfrage nicht mehr an Nachbarn weitergeleitet, die bereits im Pfad der Anfrage enthalten sind und die Anfrage bereits erhalten haben. Im Gegensatz zum statischen Netzwerk verändern sich jedoch die Nachbarn eines Peers im naiven Ansatz und im *Default Netzwerk* durch die Volatilität der Peers im dynamischen Netzwerk. Dadurch sind potentiell mehr Peers für das Weiterleiten einer Anfrage lokal wählbar, wodurch die Anzahl der Nachrichten steigt.

Wir beobachten auch bei *IBL LRU-40* einen Anstieg der Nachrichten im dynamischen (96 Nachrichten) gegenüber dem statischen Netzwerk (72 Nachrichten). Wir erklären diese Steigerung mit demselben Effekt wie beim naiven Ansatz. *IBL LRU-40* baut jedoch zusätzlich adaptive Shortcuts auf, wodurch Duplikate im Pfad der Anfrage auftreten können und eine Anfrage

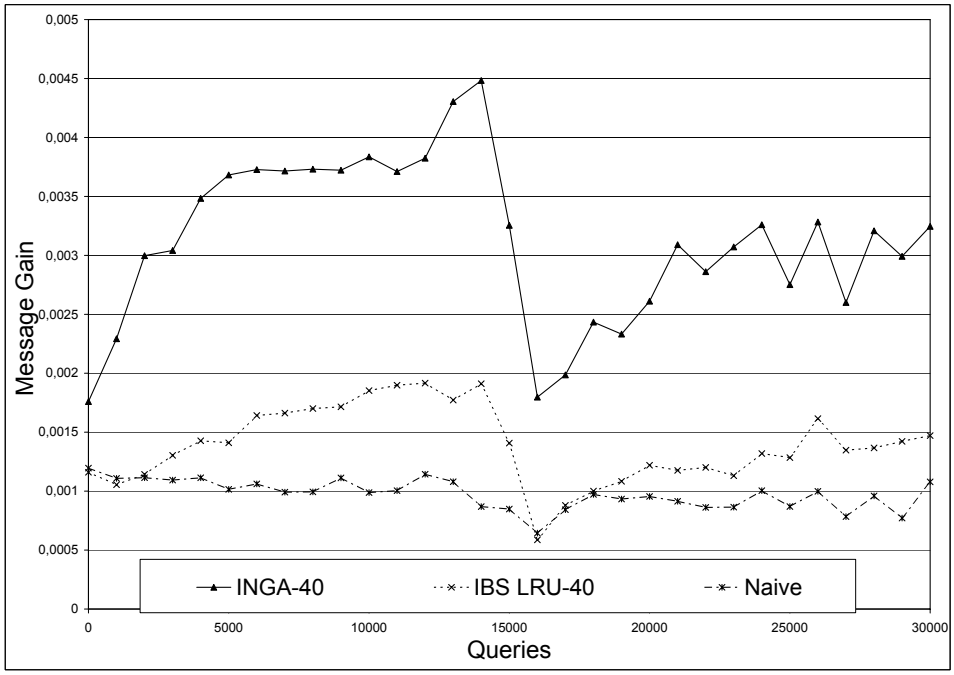


Abbildung 7.11: Message Gain: Dynamisches Netzwerk

nicht mehr weitergeleitet wird. Dadurch reduzieren sich die Nachrichten geringfügig gegenüber dem naiven Ansatz.

Im Gegensatz zu dem naiven Ansatz und *IBL LRU-40* reduziert INGA deutlich die Anzahl der Nachrichten, sowohl im statischen (58 Nachrichten) und im dynamischen Netzwerk (59 Nachrichten). Diesen Effekt führen wir auf die unterschiedlichen Overlays, besonders auf den Bootstrapping Layer, zurück. Durch die Fokussierung auf wenige Peers bei der Weiterleitung der Anfragen benötigt INGA deutlich weniger Nachrichten, und damit Netzwerkkosten, gegenüber dem naiven Ansatz und *IBL LRU-40*.

Message Gain. Ähnlich dem statischen Netzwerk verringert INGA deutlich die Anzahl der Nachrichten und erhöht den Recall gegenüber dem naiven Ansatz und *IBL LRU-40*. Grafik 7.11 zeigt den Leistungsgewinn für alle drei Ansätze. Während für den naiven Ansatz der Message Gain weitestgehend konstant ist, steigt er für *IBL LRU-40* und INGA mit zunehmender Anzahl von Anfragen an. INGA erreicht dabei etwas mehr als den zweifachen Wert von *IBL LRU-40* und eine ca. vierfache Steigerung gegenüber dem naiven Ansatz.

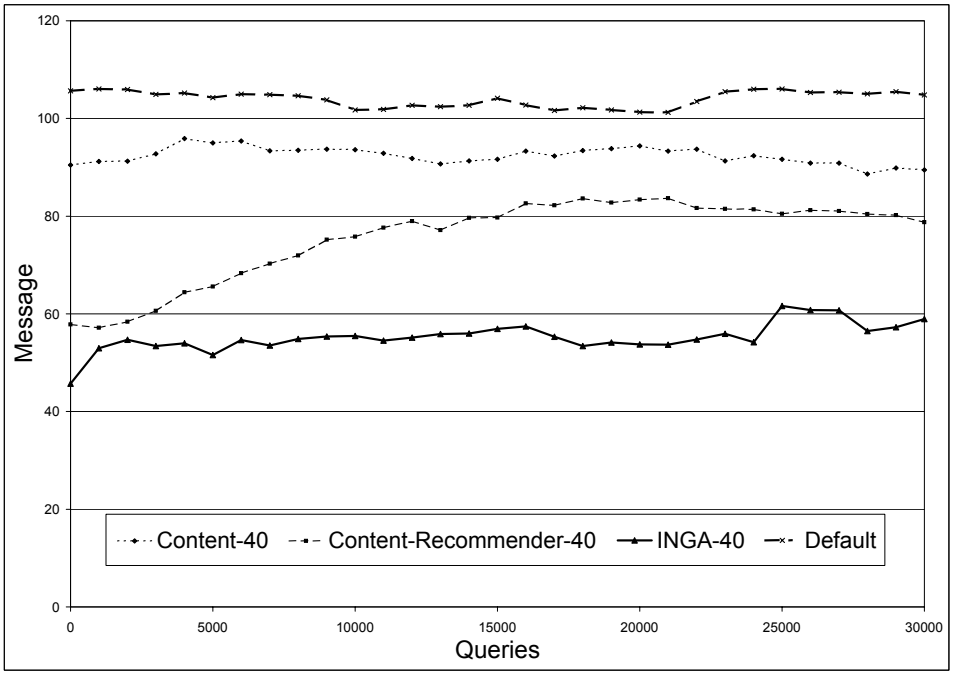


Abbildung 7.12: Message: Beitrag einzelner Layer im dynamischen Netzwerk

7.3.3 Beitrag der Layer und Einfluss der Index Parameter

Wir vermuten, das die Leistung von INGA von den verwendeten Overlays und von der Index-Größe und den Gewichtungen der einzelnen Index-Parameter abhängt. Im eine optimale Kombination zu bestimmen, untersuchen wir in diesem Abschnitt das Verhalten von INGA in Simulationen mit verschiedenen Parametern.

Layer tragen unterschiedlich zur Leistungssteigerung bei. Im ersten Experiment betrachten wir die Leistungsfähigkeit der einzelnen Overlays. Grafik 7.12 zeigt den Verbrauch von Nachrichten im Vergleich zum Default Layer. *Content-40* verwendet dieselbe Strategie zum Erzeugen von Shortcuts wie *IBL LRU-40* und benötigt die meisten Nachrichten (ca. 92 Nachrichten). *Content-Recommender 40* benötigt nur 63 Nachrichten, der Verbrauch steigt im Verlauf der Simulation auf 79 Nachrichten an. Wir erklären diesen Effekt dadurch, das zuerst wenige Peers, die besonders lange online sind, Recommender und Content Provider Shortcuts aufbauen. Dadurch fokussieren sich die Anfragen auf wenige Peers. Durch die Duplikaterkennung im Pfad der Anfrage wird eine Anfrage nicht mehr weitergeleitet. Mit weiteren Anfragen bauen auch Peers mit einer kurzen Sessiondauer Content-

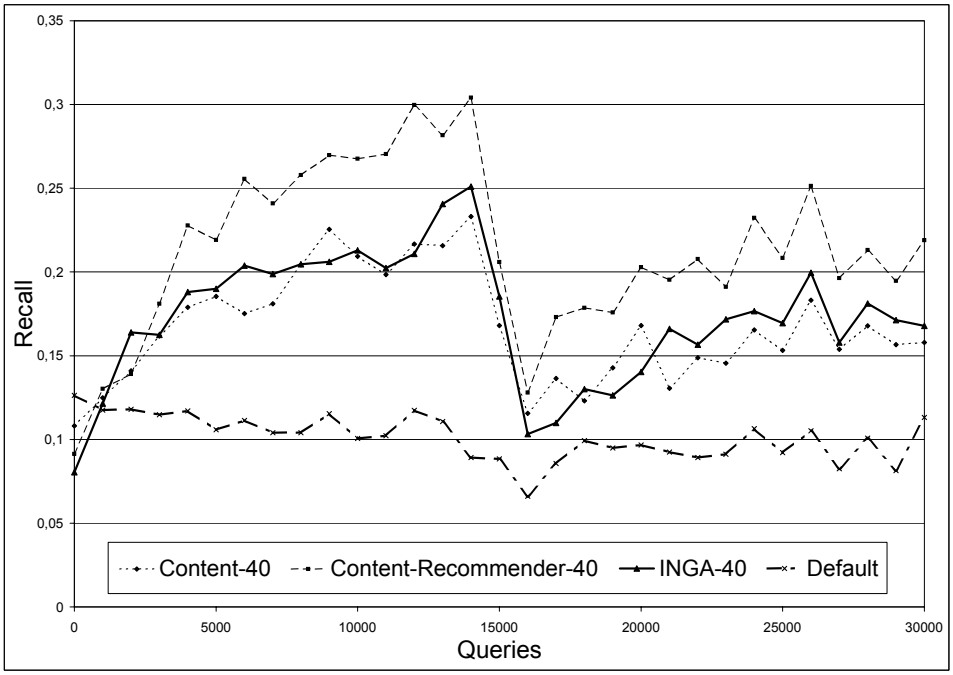


Abbildung 7.13: Recall: Beitrag einzelner Layer im dynamischen Netzwerk

und Recommender Shortcuts auf. Dadurch sinkt die Wahrscheinlichkeit, das ein lokal ausgewählter Peer bereits im Pfad der Anfrage enthalten ist und die Anzahl der verbrauchten Nachrichten stabilisiert sich. In *INGA -40* werden die Anfragen, für die keine Content oder Recommender Shortcuts gefunden werden können, auf wenige Bootstrapping Peers fokussiert. Dadurch werden oft lokal Peers ausgewählt, die bereits im Pfad der Anfrage enthalten sind. Die Anfrage wird in diesem Fall nicht weitergeleitet.

Grafik 7.13 zeigt den Recall für jeden einzelnen Overlay. Für die Erhöhung des Recalls sind zwei Einflussgrössen ausschlaggebend: Die Fokussierung der Anfragen auf wenige Peers senkt den Recall und die Menge von Shortcuts, die ein Peer pro Anfrage ermittelt. *Content-Recommender 40* hat den höchsten Recall. In diesem Layer tritt nur zu Anfang eine Fokussierung der Anfragen zu Peers mit einer hohen Verfügbarkeit auf. Ein Peer ermittelt pro aktiver Anfrage zwei Shortcuts zu zwei unterschiedlichen Peers und ermittelt zusätzlich aus den Anfragen der anderen Peers weitere Recommender Shortcuts. *INGA -40* verwendet auch die passive Recommender Strategie zur Registrierung von Shortcuts im Index. Einem höheren Recall steht die Fokussierung der Anfragen auf Peers im Bootstrapping Netzwerk

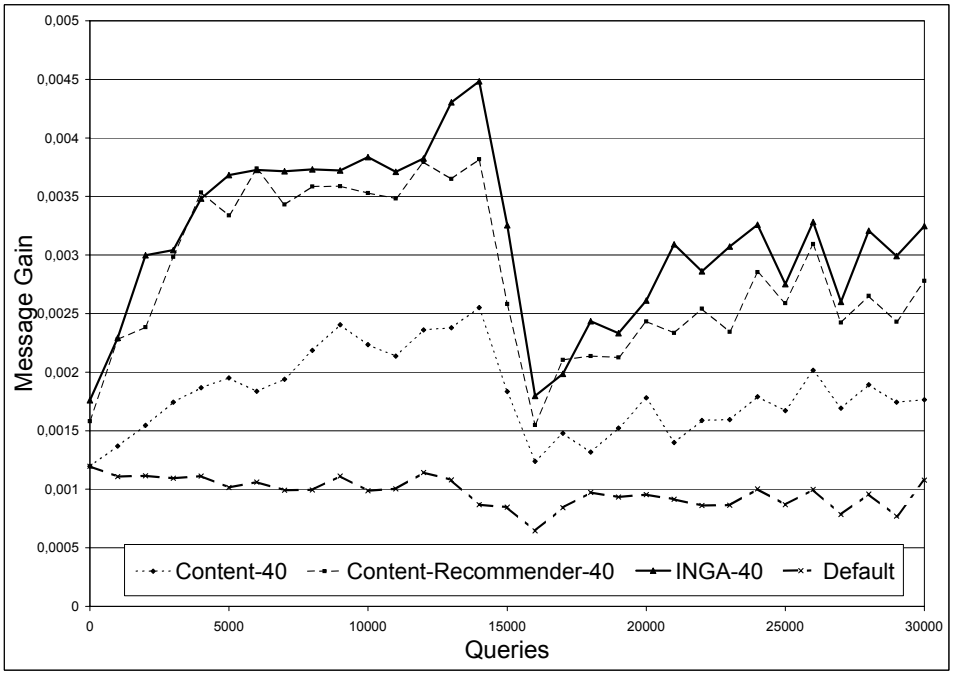


Abbildung 7.14: Message Gain: Beitrag einzelner Layer im dynamischen Netzwerk

entgegen. Dadurch werden potentiell weniger Nachrichten verschickt und auch weniger Peers angefragt. *Content-40* verwendet keine Recommender Shortcuts. Ein Peer registriert nur bei einer eigenen Anfrage Shortcuts zu Content Providern im Index. Der Ansatz hat den Nachteil, das der Peer einerseits weniger Shortcuts über die Zeit ermittelt und der Ansatz träger auf eine Volatilität im Netzwerk regiert, da bereits registrierte Shortcuts erst identifiziert werden müssen und und dann gelöscht werden können.

Zwischen dem zu erreichenden Recall und dem Verbrauch von Nachrichten muss ein Kompromiss eingegangen werden. Grafik 7.14 zeigt den Message Gain als Mass für den Kompromiss für den Content Provider Overlay, den Content Provider und Recommender Overlay und *INGA-40* im Vergleich mit dem naiven Ansatz, der durch den Default Layer dargestellt wird. Alle drei Shortcut Layer verbessern deutlich den Message Gain mit der Anzahl der Anfragen. Deutlich erzielt INGA die höchste Leistungssteigerung gegenüber allen anderen Ansätzen.

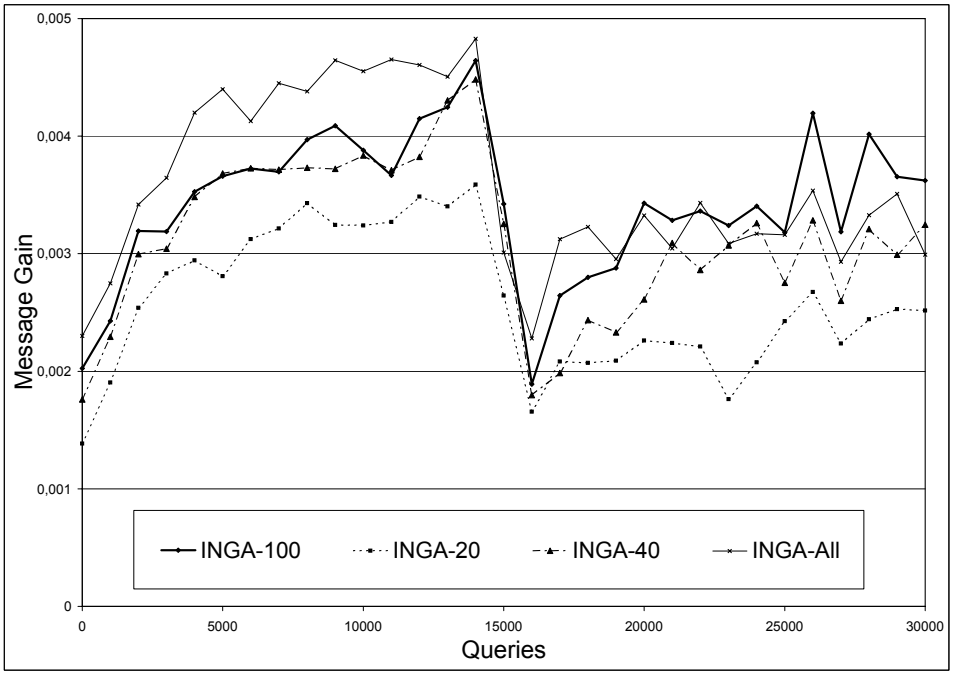


Abbildung 7.15: Vergleich der Indexgrösse im dynamischen Netzwerk

INGA arbeitet effizient mit einem kleiner Index-Größe. Im folgenden Experiment interessiert uns der Einfluss der Größe des Index auf den Message Gain. Dazu simulieren wir INGA mit einem unbegrenzten Index und mit einem Index von 100, 40, 20 Shortcuts. Unsere Experimente zeigen, das ein begrenzter Index sich analog einen unbegrenzten Index verhält. In Grafik 7.15 können wir beobachten, das ein unbegrenzter Index zunächst deutlich schneller einen hohen Message Gain erreicht als ein begrenzter Index. Nach ca. 15 Anfragen erreichen der unbegrenzte Index und der Index mit 100 bzw. 40 Shortcuts ungefähr denselben Recall, während der zu stark begrenzte Index mit 20 Shortcuts nur ca. 75% des Message Gains der anderen Graphen erreicht. In der zweiten Phase der Simulation steigt der Message Gain des auf 100 und 40 Shortcuts begrenzten Index analog dem unbegrenzten Index an. Alle drei Indizes erreichen ungefähr den selben Message Gain nach 30 Anfragen pro Peer.

Kombinierte Gewichtung der Index-Parameter ist ideal. In einer weiteren Simulation untersuchen wir den Einfluss der Parameter für den Index. Dazu haben wir mehrere Simulationen mit verschiedenen, teilweise extremen, Gewichtungen

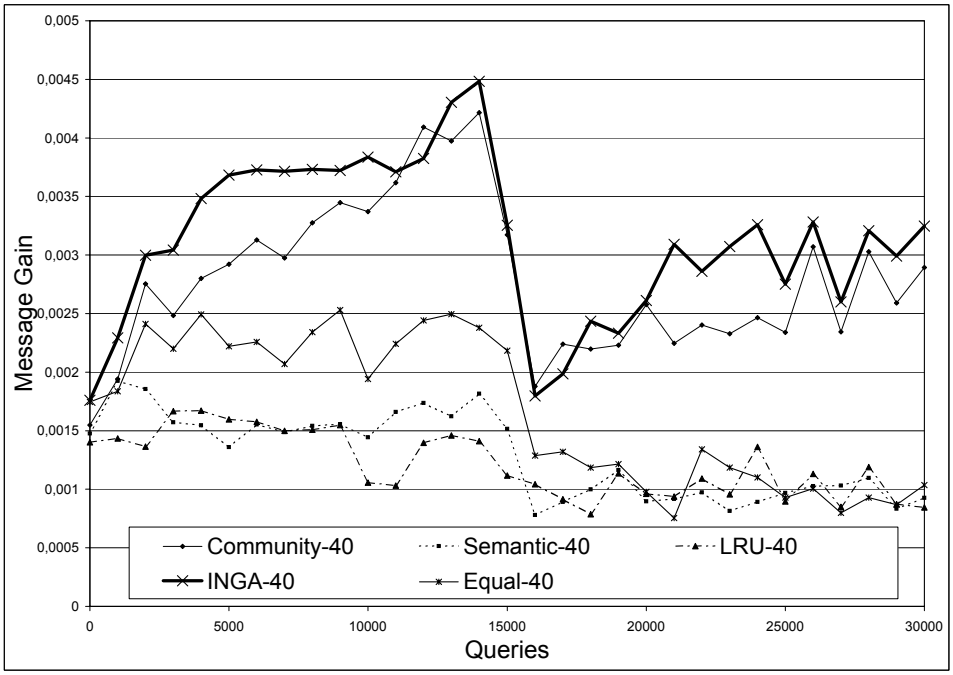


Abbildung 7.16: Vergleich der Indexewichtung

für die in Abschnitt 6.3 vorgestellten Lokalitäten durchgeführt. In allen Versuchen wurde eine Indexgrösse von 40 Einträgen verwendet. Tabelle 7.1 zeigt die Index-Parameter der einzelnen Simulationen. In Grafik 7.16 beobachten wir, dass nur

Simulation ID	Semantisch	Zeitlich	Community
Community-40	0	0	10
Semantic-40	10	0	0
LRU-40	0	10	0
Equal-40	3	3	4
INGA -40	1	1	8

Tabelle 7.1: Index-Parameter und zugeordnete Simulationen

bei *Community-40* und *INGA -40* eine Steigerung des Message Gains erfolgt. Die Autoren von [SMZ03] und [VKMvS04] schlagen den Einsatz einer LRU Strategie für Shortcut Netzwerke vor. Wir konnten diese Hypothse nicht bestätigen. Unsere Versuche zeigen, dass bei ausschließlicher Gewichtung der zeitlichen Lokalität

LRU-40, der semantischen Lokalität *Semantic-40* und bei gleicher Gewichtung aller Lokalitäten *Equal-40* keine Steigerung des Message Gains zu beobachten ist. Nur bei einer starken Gewichtung der Community Lokalität und einer geringfügige Gewichtung der semantischen und zeitlichen Lokalität steigt der Message Gain an.

7.4 Zusammenfassung

Unser Hauptinteresse in diesem Kapitel galt der Identifikation einer effizienten Routing Strategie auf der Basis lokalen Wissens und dem Vergleich unseres Ansatzes mit ähnlichen Arbeiten. Unsere Versuche zeigen, dass die Anwendung von sozialen Metaphern auf das Routing in semantischen Peer-to-Peer Netzwerken erfolgversprechend ist. Der Einsatz verschiedener Shortcut Indizes und Routing Strategien in INGA ermöglicht ein effizienteres Routing von Nachrichten gegenüber der naiven Strategie und dem *IBL* Ansatz von [SMZ03]. Wir konnten in einem statischen und einem dynamischen Netzwerk sowohl den Recall deutlich steigern, als auch die Anzahl der Nachrichten drastisch senken.

Ein weiterer wichtiger Aspekt unserer Versuche war die Identifikation von folgenden Prinzipien für das Software Engineering von Shortcut Netzwerken:

Bootstrapping senkt Messages. Durch den Einsatz von Bootstrapping Shortcuts werden Anfragen auf wenige Peers fokussiert. Dadurch sinkt der Verbrauch von Nachrichten deutlich, aber auch geringfügig der Recall. Typische Szenerien für den Einsatz von Bootstrapping Shortcuts weisen eine hohe Redundanz der gesuchten Objekte und eine geringe verfügbare Bandbreite, wie in Musik-Sharing Netzwerken, auf. Für die Desktop Suche innerhalb eines Unternehmens oder zwischen virtuellen Organisationen mit genügend vorhandener Bandbreite sollten Bootstrapping Shortcuts nur bedingt eingesetzt werden, da in diesen Szenarien ein hoher Recall benötigt wird und eine hohe verfügbare Bandbreite vorausgesetzt werden kann.

Passive Recommender Shortcuts. Ein Peer sollte alle eingehenden Anfragen untersuchen und passive Recommender Shortcuts speichern, auch wenn diese nicht zu den zentralen Interessen des Peers gehören. Dadurch bildet sich schnell ein Shortcut Netzwerk, von dem auch andere Peers profitieren. Durch die Index Strategie werden zu einem späteren Zeitpunkt unpassende Shortcuts entfernt.

Analyse der Message Pfade. Aus dem Pfad der Anfrage gewinnen wir Informationen, wer die Anfrage gestellt hat und welche Peers bereits die Anfrage

weitergeleitet haben. Kann ein lokaler Peer nur noch Shortcuts zu Peer aussuchen, die eine Anfrage bereits erhalten haben, wird die Anfrage nicht mehr weitergeleitet. Dadurch sinkt der Verbrauch von Nachrichten im Netzwerk drastisch.

Anfragespezifisch Shortcuts wählen. Entsprechend [Mil67, Kle00a] wählen wir Shortcuts aus, die der Anfrage exakt entsprechen oder Shortcuts, die der Anfrage ähnlich sind. Finden wir in unserem lokalen Index keine passenden anfragespezifischen Shortcuts, wählen wir Shortcuts zu Peers, die gut vernetzt sind.

Community Lokalität zur Begrenzung des Index. Wir speichern vorwiegend Recommender Shortcuts zu einer breiten Anzahl von Themen, bis der Index seine maximale Größe erreicht hat. Dann konzentrieren wir uns vorwiegend auf Shortcuts für unsere eigenen Anfragen (Content Provider Shortcuts) und entfernen Shortcuts zu entfernten Themen oder Recommender Peers. Die Community eines Peers, seine direkten Nachbarn, bilden zu einem großen Anteil die Peers, die in der Vergangenheit erfolgreich Dokumente für eine Anfrage liefern konnten.

Zusammenfassend können wir feststellen, das Shortcut zusätzliche Overlays über dem Default Netzwerk Overlay abbilden. In diesen Overlay Netzwerken werden Peers mit gleichen oder ähnlichen Interessen als Nachbarn gruppiert. Die entstehenden Overlay Netzwerke sind adaptiv, das heißt Nachbarn werden bei wechselnden eigenen Interessen und veränderter Verfügbarkeit durch den INGA Algorithmus ausgetauscht. Wir konnten zeigen, das Anfragen entlang der interessenspezifischen und adaptiven Overlays effizient in statischen und dynamischen Netzwerken weitergeleitet werden können.

Teil III

Diskussion

8 Zusammenfassende Diskussion

In der vorliegenden Arbeit entwickelten wir drei Ansätze, die eine effiziente Suche mittels adaptiver Overlay Strukturen in Peer-to-Peer Netzwerken ermöglichen und technologisch praktikabel sind. Nach einer Zusammenfassung der Beiträge dieser Arbeit schließen wir mit einem Ausblick auf weitere interessante Forschungsprobleme.

8.1 Zusammenfassung

Der Beitrag dieser Arbeit war die Entwicklung einer Technologie für die effiziente Suche von Dokumenten in Peer-to-Peer Netzwerken. Die Charakteristiken dieser Zielplattform erforderten die Entwicklung einer vollständig verteilten Technologie, die wechselnden Interessen und die Volatilität der Teilnehmer des Netzwerkes toleriert und trotzdem mit hoher Effizienz und möglichst geringem Aufwand für den einzelnen Nutzer die besten Peers für eine Anfrage auswählt. Wir greifen kurz die in der Einleitung formulierten Probleme auf und wiederholen ihre Lösungen, wie sie in dieser Arbeit dargestellt wurden.

Architekturmodelle für Overlay Netzwerken. Wir identifizierten in Kapitel 2 typische Charakteristiken potentieller Anwendungen der Informationssuche in Peer-to-Peer Netzwerken. Exemplarisch analysierten wir die Anwendungsgebiete *Legale Musiktauschbörse* und *Semantisch vernetzte Arbeitsplätze* anhand der gefundenen Merkmale. Wir klassifizierten Peer-to-Peer Architek-

turen für die Informationssuche nach dem *Grad der Zentralisierung* und der *Struktur der Overlays*. Wir ordneten die in dieser Arbeit vorgestellten Systeme Edutella und TOPICS den Super-Peer basierten Systemen mit einem strukturierten, globalen Index und das System INGA den semi-strukturierten Systemen mit lokalen Indizes zu.

Ansätze für globale Indexstrukturen. In Kapitel 3 stellen wir Edutella und TOPICS als Beispiele für strukturierte Overlay Netzwerke vor. Beide Ansätze verwenden einen globalen, vollständig verteilten Index zur Registrierung der Profile einzelner Peers. Zur Verringerung der Nachrichten für die Registrierung und für Anfragen von Peers identifizierten wir in Edutella drei Konzepte: statische Super-Peers, den HyperCup Broadcast Algorithmus und einen zweigeteilten Routing-Index. Der TOPICS Ansatz verwendet zur Reduzierung der Nachrichten ebenfalls eine Super-Peer basierte Index Struktur. Zur Abbildung und effizienten Anfrage der semantischen Beziehungen einer Themen-Hierarchie speichern wir *Konzepte als Schlüssel* und *Beziehungen als Objekte* in einer *verteilten Hashtabelle*. Abschließend diskutieren wir Nachteile strukturierter Overlays und stellen aktuelle Verfahren zur Lastverteilung vor.

Metaphern für Interessen-basierte Overlays. In Overlay-Netzwerken mit einem globalen Index wird das Profil des Nutzers an einem entfernten Peer registriert. Der Nutzer verliert dadurch die Kontrolle über sein Profil. Zusätzlich entstehen in volatilen Netzwerken hohe Kosten für eine Aktualisierung des Profils. Wir lösten dieses Problem durch die *Einführung lokaler Shortcut Indizes*. Zur Erstellung der Shortcuts definierten wir in Kapitel 4 folgende *soziale Metaphern*: Eine Anfrage wird an eine Person gestellt, die diese Anfrage in der Vergangenheit bereits korrekt beantwortet hat (*Content Provider*). Kennen wir keine Person, die eine ähnliche Anfrage in der Vergangenheit beantwortet hat, suchen wir nach Personen, die eine ähnliche Anfrage in der Vergangenheit gestellt hat (*Recommender*). Kennen wir keine solchen Personen, schicken wir die Anfrage zu einer Person, die bereits ein 'breites' soziales Netzwerk zu anderen Personen entwickelt hat (*Bootstrapping Peer*). Um neue Personen für bereits bekannte Anfragen zu ermitteln, leiten wir eine Anfrage zu einem geringen Anteil auch an die durch die initiale Netzwerk Struktur vorgegebenen Nachbarn (*Default Network*) weiter. Wir verwendeten diese Nachbarn ebenfalls, wenn wir noch keine Shortcuts aufgebaut haben.

Peer-to-Peer Architektur für lokale Indexstrukturen. Auf der Basis unstrukturierter Overlay Netzwerke definierten wir in Kapitel 4 eine *neue Infrastruk-*

tur:INGA. In ihr repräsentiert jeder Peer eine Person. Entsprechend den sozialen Metaphern speichert jeder Peer lokal Shortcuts, die *abhängig* oder *unabhängig* von dem *konkreten Thema der Anfrage* erzeugt werden. Zur ersten Kategorie gehören Shortcuts zu Peers, die eine Antwort auf eine Anfrage geben konnten, Shortcuts zu Peers, die eine Nachricht weitergeleitet haben und Shortcuts zu Peers, die eine Anfrage zu diesem Thema bereits in der Vergangenheit gestellt haben. Zur letzteren Kategorie gehören Shortcuts zu Peers, die gut vernetzt sind, als auch Shortcuts zu zufällig ausgewählten Peers.

Die verschiedenen Shortcuts in INGA repräsentieren vier unterschiedliche Typen von Overlay Netzwerken. Für jedes Overlay Netzwerk definierten wir *formal* in Kapitel 4 die Struktur eines Shortcuts und die korrespondierende Index Struktur.

Erzeugen von Overlay Netzwerken auf der Basis von Shortcuts. In Kapitel 5 entwickelten wir neue Konzepte zur *Erstellung von Recommender und Bootstrapping Shortcuts*. Unser Ziel war das *vollständig automatisierte* Finden von relevanten Shortcuts ohne zusätzlichen Aufwand für den Nutzer. Wir lösten dieses Problem durch die Analyse der Resultate eigener Anfragen, des transitiven Weges dieser Anfragen über entfernte Peers sowie durch beobachten von Anfragen entfernter Peers, die durch den lokalen Peer weitergeleitet wurden.

Shortcuts werden in zwei lokalen Indizes gespeichert: ein Index für Shortcuts zum Default Network Layer, der den 'Index' momentan existierender unstrukturierter Netzwerke abbildet und ein erweiterter Index, der Content-, Recommender- und Bootstrapping Shortcuts abbildet.

Algorithmen zur Auswahl der besten Shortcuts. Zur vollständig automatischen Auswahl der besten Shortcuts für eine Anfrage entwickelten wir in Kapitel 6 eine völlig neue Familie von Algorithmen. Für jeden der vier Typen von Overlay Netzwerken entwickelten wir einen eigenen Algorithmus zur Auswahl der besten Shortcuts für eine Anfrage: *Fireworks, TopGreedy, Top Boot, Serendipity*.

Zur Berechnung des *Bootstrapping Rank* entwickelten wir eine völlig neue Metrik. Dazu berücksichtigten wir die aktuelle Vernetzung eines Peers und ermittelten den Bootstrapping Wert aus dem lokalen Shortcut Index.

Basierend auf den sozialen Metaphern entwickelten wir den ersten Algorithmus, der für eine Anfrage dynamisch den passenden Routing Algorithmus auswählt. Der Algorithmus *Dynamic* bevorzugt Routing Strategien die, gezielt für das Thema der Anfrage, exakt übereinstimmende oder ähnliche

Content Provider und Recommender Shortcuts auswählen. Wird kein passender Shortcut gefunden, greift der Algorithmus auf Bootstrapping Shortcuts und zuletzt auf die Nachbarn des initialen Netzwerkes zurück.

Algorithmen für die Pflege der lokalen Indizes. Die Auswahl von Shortcuts für eine Anfrage und das Löschen bereits registrierter Shortcuts beeinflussen sich gegenseitig. Um zu entscheiden, welcher Shortcut nicht mehr im Index verbleibt, entwickelten wir in Kapitel 6 eine *neue Index-Strategie für volatile Shortcut Netzwerke*. Sie ordnet die Content und Recommender Shortcuts und berechnet für jeden Shortcut eine Relevanz. Die Shortcuts mit der geringsten Relevanz werden gelöscht. Für die Berechnung der Relevanz verwendeten wir drei Parameter: Wir untersuchten den Typ des Shortcuts und gewichteten Content Provider Shortcuts doppelt so hoch wie Recommender Shortcuts. Weiterhin berücksichtigten wir die semantische Ähnlichkeit der gefundenen Recommender Shortcuts zu unseren eigenen Anfragen. Schließlich verwendeten wir den Zeitpunkt der letzten Aktualisierung eines Shortcuts. Wir aktualisierten einen Shortcut, wenn wir ihn erfolgreich für unsere eigene Anfrage verwendeten.

Quantifizierung der Effizienz und Simulation volatiler Netzwerke. In Kapitel 7 simulierten wir INGA in einem dynamischen und einem statischen Netzwerk. Wir erzeugten *Volatilität* sowohl durch wechselnde Interessen als auch durch die unterschiedliche Verfügbarkeit der einzelnen Peers. Wir konnten zeigen, dass der Einsatz verschiedener Shortcut Indizes und Routing Strategien in INGA sowohl in einem statischen als auch einem dynamischen Netzwerk eine *drastische Steigerung des Recall* und eine *deutliche Senkung der Anzahl der Nachrichten* gegenüber State-of-the-Art Ansätzen ermöglichte.

Durch zahlreiche Simulationen identifizierten wir für das Software Engineering zukünftiger Applikationen auf der Basis von Shortcut Netzwerken folgende grundlegende Prinzipien:

- Bootstrapping Shortcuts fokussieren die Anfragen und senken die Nachrichten.
- Passive Recommender Shortcuts erlauben einen schnellen Aufbau des Index und vermeiden dadurch einen *cold start effect*.
- Ein dynamisches Auswählen von Routing Strategien reduziert deutlich Nachrichten.
- Eine LRU Index-Strategie allein ist ungeeignet für Shortcut Netzwerke und muss mit anderen Strategien wie *Similarity* oder *Community*

kombiniert werden. Der Index sollte langfristig deutlich mehr Content Provider Shortcuts als Recommender Shortcuts beinhalten.

Zusammenfassend können wir feststellen, das unsere Lösung der beschriebenen Probleme eine einfache und robuste Technologie für eine effiziente Suche in Peer-to-Peer Netzwerken ermöglicht. Unser Konzept der adaptiven Shortcut Overlays erlaubt eine vollständig automatische Identifikation von Nachbarn im Netzwerk mit ähnlichen oder gleichen Interessen und erfolgt nur auf Basis beobachteter Anfragen. Der Vorteil gegenüber existierenden Ansätzen mit einem globalen Index ist, dass kein zusätzlicher Aufwand zur Publikation des Profils eines Peers benötigt wird, da die Profile lokal beim Nutzer in einem Index gespeichert werden. Die Profile entfernter Peers werden durch Beobachtung identifiziert und Peers mit ähnlichen Profilen im Overlay Netzwerk als Nachbarn angeordnet. Die von uns entwickelten Algorithmen und Technologien tolerieren wechselnde Interessen und eine hohe Volatilität der einzelnen Peers. Trotzdem ermöglichen sie ein effizientes Routing von Anfragen. Obwohl das Problem bereits in einigen anderen Arbeiten identifiziert wurde, stellt diese Arbeit die erste umfassende Lösung für die Identifikation von adaptiven Shortcut Overlays, deren Erstellung, deren Management und dem effizienten Routing entlang von Shortcut Overlays zur Verfügung.

8.2 Perspektiven zukünftiger Forschung

Die in dieser Arbeit vorgestellten Methoden und Algorithmen bilden eine leistungsstarke Basis zur Entwicklung von Peer-to-Peer Applikationen auf der Basis von adaptiven Shortcut Overlays. Dennoch konnten wir in dieser Arbeit nicht alle Probleme dieser Applikationsdomäne adressieren. Wir stellen kurz ausgewählte zusätzliche Problemstellungen vor und skizzieren potentielle Ansatzpunkte für die weitere Forschung.

Ausdrucksmächtigkeit der Anfragen. Zukünftige Shortcut Overlays sollten die Auswahl der 'besten' Peers auch für komplexe Anfragen unterstützen. Beispielsweise müssen für eine konjunktive Anfrage mit mehreren Termen Peers ausgewählt werden, die möglichst alle Terme der Anfrage unterstützen. Lokale Indexstrukturen verfügen jedoch nur in Ausnahmefällen über alle dafür benötigten Informationen. Fehlt in der lokalen Indexstruktur die Zuordnung eines Terms zu einem Peer bedeutet das entweder, dass der Peer keine Dokumente für diesen Term bereithält oder aber, die wahrscheinlichere Annahme, dass bisher für diesen Term an den Peer keine Anfrage gestellt wurde. Es wird eine probabilistische Funktion benötigt, die trotz unzureichender lokaler Informationen die besten Peers für eine Anfrage auswählt. Derarti-

ge Funktionen wurden bisher unzureichend für Shortcut Netzwerke untersucht, und sind Schwerpunkt aktueller Forschung. Arbeiten von [BMWZ05, BMT+05, Coo04, CAPMN02] sammeln lokal Daten über die Gewichtung und Korrelation der Terme im Netzwerk und wenden Maße aus dem Information Retrieval, wie CORI und Gloss, an.

Emergent Semantics. Zukünftige Peer-to-Peer Systemen werden entweder ausschließlich über lokale oder über globale und lokale Interpretationen von Anfragen und Dokumenten verfügen. Durch unterschiedliche lokalen Interpretationen müssen jedoch zusätzliches semantische Vereinbarungen zwischen dem anfragenden und dem antwortenden Peer vereinbart werden. Durch die hohe Volatilität in einem Peer-to-Peer Netzwerk und der Autonomie der einzelnen Peers ändern sich diese Vereinbarungen zudem häufig. Dadurch unterscheiden sich diese Systeme grundsätzlich von Systemen mit vordefinierten Semantiken, bei denen die Interoperabilität durch eine statische Komponente, wie eine globale Ontologie, fest beschrieben wird. Stattdessen werden neu auftauchende semantische Vereinbarungen (*emergent semantics*) [ACMO+04] in einem Bottom-Up Konsens zwischen einzelnen Peers gebildet. Dieser Prozess basiert auf den Interaktionen des Menschen mit der Maschine und definiert iterativ neue semantische Vereinbarungen. Aktuelle Ansätze dieses hochaktuellen Forschungsgebiets, wie [ACMH02], nutzen den Pfad der Anfrage aus und beobachten die Resultate und Antworten für eine Anfrage.

Personalisierung der Suchstrategien. Aktuelle Applikationen für die Peer-to-Peer Suche verwenden eine fest programmierte Strategie zur Suche. Viele Nutzer profitieren von der vordefinierten Prozess-Logik der Suche. Ein Nachteil dieses Ansatzes ist jedoch die schlechte *Personalisierbarkeit des Suchprozesses*. Erfahrenere Nutzer wenden, zur Erhöhung des Recalls oder der Precision für eine Anfrage, komplexe Sucheprozesse an, die mehrere unterschiedliche Anfragen beinhalten. Eine interessante Weiterentwicklung wäre eine Komponente, die das Suchverhalten des Nutzers beobachtet und erkennen kann. In einem zweiten Schritt simuliert die Komponente zu einer Anfrage den nutzerspezifischen Suchprozess und schlägt bei unzureichenden Resultaten neue Anfragen vor, bzw. stellt diese automatisiert. Alternativ soll die Komponente besonders erfolgreiche Suchstrategien anderer Peers dem Nutzer vorschlagen bzw. anwenden.

Expertensuche in virtuellen Organisationen. INGA erlaubt die Identifizierung 'passender' Peers für eine Anfrage auf der Basis bereits beobachteter Interaktionen. Aufgrund der Unübersichtlichkeit und Unüberschaubarkeit glo-

bal agierender Unternehmen und virtueller Organisationen existiert dort das Problem der Suche von geeigneten Personen für eine Anfrage. In INGA wird jeder Peer einer Person zugeordnet. Eine interessante Erweiterung wäre die Optimierung der bestehender Methoden zur Identifikation von Peers, der Auswahl von Peers bzw. der Verwaltung des Index mit dem Ziel einer Anwendung, die Dokumente und passende Experten für eine Anfrage identifiziert.

Privacy Enhancement Technologien. Der Ansatz von INGA basiert auf einem Netzwerk, bei dem jeder Peer Daten über andere Peers sammelt. So beobachtet ein Peer sowohl die eigenen, als auch Anfragen und Resultate entfernter Peers. Diese Informationen werden in einem lokalen Index gespeichert, der wiederum von entfernten Peers gelesen werden kann. Sie ermöglichen eine Zuordnung von Peers zu Interessen und verbessern die Effizienz des Routings. Auf der anderen Seite lassen sich diese persönlichen Daten auch missbräuchlich verwenden, beispielsweise zur Profilbildung der Peers und deren Auswertung für Spam Mails. Die Akzeptanz von Shortcut Netzwerken in Unternehmen hängt unter anderem von der Wahrung der Privatsphäre ab. Zukünftige Technologien sollten einerseits die Privatsphäre eines Peers bewahren und trotzdem die Effizienz des Netzwerkes nicht deutlich senken, bzw. einen Kompromiss erlauben. INGA weist bereits mit der Lokalität der Daten ein nützliches Designmerkmal auf. Der Ansatz erlaubt jedem Nutzer lokal die Sicherheit der eigenen Shortcuts und Dokumente zu gewährleisten und erlaubt deren Löschung aus dem Index ohne auf einen zentralen Server zugreifen zu müssen. Zukünftige Technologien können diesen Prozess automatisieren, die Identität des Peers anonymisieren oder lokale Shortcuts und Dokumente nur ausgewählten Peers zugänglich machen.

Literaturverzeichnis

[AA04] Lada Adamic and Eytan Adar. How to search a social network. Technical report, HP Labs, 1501 Page Mill RoadPalo Alto, CA 94304, 2004.

[Abe01] Karl Aberer. P-grid: A self-organizing access structure for p2p information systems. In *Sixth International Conference on Cooperative Information Systems (CoopIS)*, Trento, Italy, 2001.

[ACMH02] Karl Aberer, Philippe Cudre-Mauroux, and Manfred Hauswirth. A framework for semantic gossiping. *SIGMOD Rec.*, 31(4):48–53, 2002.

[ACMO+04] Karl Aberer, Philippe Cudré-Mauroux, Aris M. Ouksel, Tiziana Catarci Mohand-Said Hacid, Arantza Illarramendi, Vipul Kashyap, Massimo Mecella, Eduardo Mena, Erich J. Neuhold, Olga De Troyer, Thomas Risse, Monica Scannapieco, Fèlix Saltor, Luca de Santis, Stefano Spaccapietra, Steffen Staab, and Rudi Studer. Emergent semantics principles and issues. In *9th International Conference on Database Systems for Advanced Applications (DASFAA)*, 2004.

[ADU71] Alfred V. Aho, Peter J. Denning, and Jeffrey D. Ullman. Principles of optimal page replacement. *J. ACM*, 18(1):80–93, 1971.

[AGS02] Roberto J. Bayardo Jr.and Rakesh Agrawal, Daniel Gruhl, and Amit Somani. Youserv: a web-hosting and content sharing tool for the masses. In *Proc. of the World Wide Web Conference*, pages 345–354, 2002.

[All96] James Allan. Incremental relevance feedback for information filtering. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 270–278, New York, NY, USA, 1996. ACM Press.

[ALPH01] Lada A. Adamic, Rajan M. Lukose, Amit R. Puniyani, and Bernardo A. Huberman. Search in Power-law Networks. In *Physical Review E, 64 46135*, 2001.

[ATS04] Stephanos Androutsellis-Theotokis and Diomidis Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Comput. Surv.*, 36(4):335–371, 2004.

[BBK02] Magdalena Balazinska, Hari Balakrishnan, and David Karger. Ins/twine: A scalable peer-to-peer architecture for intentional resource discovery. In *M. Balazinska, H. Balakrishnan, and D. Karger. In Proceedings of Pervasive 2002.*, 2002.

[BCAA04] Marcelo Werneck Barbosa, Melissa Morgado Costa, Jussara M. Almeida, and Virglio A. F. Almeida. Using locality of reference to improve performance of peer-to-peer applications. In *WOSP '04: Proceedings of the fourth international workshop on Software and performance*, pages 216–227. ACM Press, 2004.

[BG03] D. Brickley and R. V. Guha. Resource Description Framework (RDF) Schema Specification 1.0, 2003. `http://www.w3.org/TR/rdf-schema`.

[BGK+02] Philip A. Bernstein, Fausto Giunchiglia, Anastasios Kementsietsidis, John Mylopoulos, Luciano Serafini, and Ilya Zaihrayeu. Data management for peer-to-peer computing: A vision. In *Fifth International Workshop on the Web and Databases*, Madison, Wisconsin, June 2002.

[BKvH02] J. Broekstra, A. Kampman, and F. van Harmelen. Sesame: A generic architecture for storing and querying RDF and RDF schema. In *The Semantic Web - ISWC 2002*, volume 2342 of *LNCS*, pages 54–64. Springer, 2002.

[Blo70] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970.

[BMB02] Dave Beckett, Eric Miller, and Dan Brickley. Expressing simple dublin core in RDF/XML. Technical report, Dublin Core Metadata Initiative, 2002. `http://dublincore.org/documents/2002/07/31/dcmes-xml/`.

[BMT+05] Matthias Bender, Sebastian Michel, Peter Triantafillou, Gerhard Weikum, and Christian Zimmer. Improving collection selection with overlap-awareness. In *28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR05)*, pages –, Salvador, Brazil, 2005. ACM.

[BMWZ05] Matthias Bender, Sebastian Michel, Gerhard Weikum, and Christian Zimmer. The MINERVA project: Database selection in the context of P2P search. In Gottfried Vossen, Frank Leymann, Peter Lockemann, and Wolffried Stucky, editors, *Datenbanksysteme in Business, Technologie und Web (BTW2005; 11. Fachtagung des GI-Fachbereichs Datenbanken und Informationssysteme (DBIS))*, volume P-65 of *Lecture Notes in Informatics*, pages 125–144, Karlsruhe, Germany, March 2005. Gesellschaft für Informatik.

[CAPMN02] Francisco Matias Cuenca-Acuna, Christopher Peery, Richard P. Martin, and Thu D. Nguyen. PlanetP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities. Technical Report DCS-TR-487, Department of Computer Science, Rutgers University, September 2002.

[CFB04] Vicent Cholvi, Pascal Felber, and Ernst Biersack. Efficient search in unstructured peer-to-peer networks. *European Transactions on Telecommunications: Special Issue on P2P Networking and P2P Services*, (15):535–548, November 2004.

[CGM02a] Arturo Crespo and Hector Garcia-Molina. Routing indices for peer-to-peer systems. In *International Conference on Distributed Computing Systems*, july 2002.

[CGM02b] Arturo Crespo and Hector Garcia-Molina. Semantic Overlay Networks for P2P Systems. Technical report, Computer Science Department, Stanford University, 2002.

[CK01] E. Cohen and H. Kaplan. Refreshment policies for web content caches. In *Proceedings of IEEE Infocom*, 2001.

[CLL04] Jacky Chu, Kevin Labonte, and Brian Neil Levine. Availability and popularity measurements of peer-to-peer file systems, technical report. Technical report, University of Massachusetts, June 2004.

[CMH+02] Ian Clarke, Scott G. Miller, Theodore W. Hong, Oskar Sandberg, and Brandon Wiley. Protecting Free Expression Online with Freenet. *IEEE Internet Computing*, 6(1):40–49, 2002.

[Coo04] Brian Cooper. Guiding queries to information sources with infobeacons. In *ACM/IFIP/USENIX 5th International Middleware Conference*, Toronto, 2004.

[Coo05] Microsft Cooperation. Microsoft Solutions for Small and Medium Business- Peer-to-Peer Networking with Windows XP. Technical report, 02/2005.

[CRB+03] Yatin Chawathe, Sylvia Ratnasamy, Lee Breslau, Nick Lanham, and Scott Shenker. Making gnutella-like P2P systems scalable. In *SIGCOMM '03: Conference on Applications, technologies, architectures, and protocols for computer communications*, pages 407–418. ACM Press, 2003.

[CW04] Hailong Cai and Jun Wang. Foreseer: A Novel, Locality-Aware Peer-to-Peer System Architecture for Keyword Searches. In *ACM/IFIP/USENIX International Middleware Conference*, volume 3231 of *LNCS*, 2004.

[DADA96] R. Dolin, D. Agrawal, L. Dillon, and A. El Abbadi. Pharos: A scalable distributed architecture for locating heterogeneous information sources. Technical Report TRCS96-05, 16, 1996.

[DF04] Stefan Decker and Martin Frank. The networked semantic desktop. In *WWW2004 Workshop Application Design, Development and Implementation Issues in the Semantic Web*, 2004.

[DFA00] R. DingleDine, M. Freedman, and D. Andmolnar. The FreeHaven project: Distributed anonymous storage service. In *International Workshop on Design Issues in Anonymity and Unobservability. 67-95*, volume 2009 of *Lecture Notes in Computer Science*, 2000.

[DG89] J. L. Deneubourg and C. Gross. Collective patterns and decision making. *Ethology, Ecology, and Evolution*, 1, 1989.

[DGH+87] Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. In *PODC '87: Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, pages 1–12. ACM Press, 1987.

[DMO] The Open Directory Catalog. `http://www.dmoz.org`.

[DZW04] B. D. Davison, W. Zhang, and B. Wu. Lessons from a gnutella-web gateway. In *Thirteenth International World Wide Web Conference Alternate Track Papers and Posters*, pages 502–503, New York, 2004. ACM Press.

[Exc04] Care Date Exchange. Company webseite. http://www.carescience.com, 2004.

[FFK04] André Köhler Frank Fuchs-Kittwoski. Knowledge creating communities in the context of work processes. *ACM SIGGROUP Bulletin*, 23(3), 2004.

[FHKM04] F. Le Fessant, S. Handurukande, A.-M. Kermarrec, and L. Massoulie. Clustering in Peer-to-Peer File Sharing Workloads. In *3rd. IPTPS Workshop*, 2004.

[FPV+98] James C. French, Allison L. Powell, Charles L. Viles, Travis Emmitt, and Kevin J. Prey. Evaluating database selection techniques: A testbed and experiment. In *Research and Development in Information Retrieval*, pages 121–129, 1998.

[GBL+03] Indranil Gupta, Ken Birman, Prakash Linga, Al Demers, and Robbert van Renesse. Kelips: Building an efficient and stable P2P DHT through increased memory and background overhead. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, 2003.

[GBL+04] Godfrey, Brighten, Karthik Lakshminarayanan, Sonesh Surana, Richard Karp, and Ion Stoica. Load balancing in dynamic structured p2p systems. In *Proceedings of IEEE Infocom*, Hong Kong, 2004.

[GDS+03] Krishna P. Gummadi, Richard J. Dunn, Stefan Saroiu, Steven D. Gribble, Henry M. Levy, and John Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *SOSP '03: Nineteenth ACM symposium on Operating systems principles*, pages 314–329. ACM Press, 2003.

[GGM95] Luis Gravano and Héctor García-Molina. Generalizing GlOSS to vector-space databases and broker hierarchies. In *International Conference on Very Large Databases, VLDB*, pages 78–89, 1995.

[GMY02] Hector Garcia-Molina and Bervely Yang. Efficient search in peer-to-peer networks. In *International Conference on Distributed Compting Systems (ICDCS)*, 2002.

[Gnu04] The gnutella developer forum, http://rfc-gnutella.sourceforge.net, 2004.

[Gon01] Li Gong. Project jxta: A technology overview. Technical report, Sun Microsystems Inc., 2001.

[GWJD03] Leonidas Galanis, Yuan Wang, Shawn R. Jeffery, and David J. DeWitt. Locating Data Sources in Large Distributed Systems. In *29th Conference on Very Large Data Bases (VLDB)*, 2003.

[HBM+04] P. Haase, J. Broekstra, M.Ehrig, M. Menken, P.Mika, M. Plechawski, P. Pyszlak, B. Schnizler, R. Siebes, S. Staab, and C. Tempich. Bibster - a semantics-based bibliographic peer-to-peer system. In *Proceedings of the International Semantic Web Conference (ISWC)*, 2004.

[HIMT03] Alon Y. Halevy, Zachary G. Ives, Peter Mork, and Igor Tatarinov. Piazza: Data management infrastructure for semantic web applications. In *Twelfth International World Wide Web Conference (WWW2003)*, Budapest, Hungary, May 2003.

[HMSB87] Michael N. Huhns, Uttam Mukhopadhyay, Larry M. Stephens, and Ronald D. Bonnell. *Distributed Artificial Intelligence*, chapter DAI for document retrieval: The MINDS project, pages 249–283. Pitman/Morgan Kaufmann, 1987.

[IEE02] IEEE P1484.12 Learning Object Metadata Working Group. Draft standard for learning object metadata. Technical report, IEEE Learning Technology Standards Committee (LTSC), 2002. `http://ltsc.ieee.org/doc/wg12/LOM_1484_12_1_v1_Final_Draft.pdf`.

[IF99] C. Kesselmann I. Foster. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufman Publishers, Inc. San Francisco, California, 1999.

[Ini04] Dublin Core Meta Data Initiative. Dcmi metadata terms. Technical Report 2004-12-20, DCMI Usage Board, 2004.

[IRF04] Adriana Iamnitchi, Matei Ripeanu, and Ian Foster. Small-World File-Sharing Communities. In *23th. IEEE InfoCom HongKong*, 2004.

[KAD+04] Pradnya Karbhari, Mostafa Ammar, Amogh Dhamdhere, Himanshu Raj, George Riley, and El len Zegura. Bootstrapping in Gnutella: A Measurement Study. In *The 5th anuual Passive and Active Measurement Workshop (PAM)*, LNCS, 2004.

[Kan99] Gene Kan. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, chapter Gnutella, pages 94–122. O'Reilly, 1999.

[KBS02] Peter J. Keleher, Bobby Bhattacharjee, and Bujor D. Silaghi. Are virtualized overlay networks too much of a good thing? In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 225–231. Springer-Verlag, 2002.

[Kle98] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. In *SODA '98: Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, pages 668–677, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics.

[Kle00a] John Kleinberg. Navigation in a small world. *Nature*, (406):845, 2000.

[Kle00b] Jon Kleinberg. The Small-World Phenomenon: An Algorithmic Perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, 2000.

[KNS04] Irwin King, Cheuk Hang Ng, and Ka Cheung Sia. Distributed content-based visual information retrieval system on peer-to-peer networks. *ACM Trans. Inf. Syst.*, 22(3):477–501, 2004.

[KSS97] Henry Kautz, Bart Selman, and Mehul Shah. Referral Web: Combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, 1997.

[LBM03] Y. Li, Z.A. Bandar, and D. McLean. An Approach for messuring semantic similarity between words using semantic multiple information sources. In *IEEE Transactions on Knowledge and Data Engineering*, volume 15, 2003.

[LCC+02] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker. Search and replication in unstructured peer-to-peer networks. In *ICS '02: 16th international conference on Supercomputing*, pages 84–95. ACM Press, 2002.

[LGH02] Alexander Löser, Christian Grune, and Marcus Hoffmann. A didactic model, definition of learning objects and selection of metadata for an online curriculum. In *International Workshop on Interactice Computer Aided Learning (ICT)*, Villach, Austria, September 2002.

[LHH+04] B. Loo, J. Hellerstein, R. Huebsch, S. Shenker, and I. Stoica. Enhancing p2p file-sharing with an internet-scale query processor. In *In Proc. of Int. Conf. on Very Large Databases (VLDB)*, Toronto, 2004.

[LNS+03] Alexander Löser, Felix Naumann, Wolf Siberski, Wolfgang Nejdl, and Uwe Thaden. Semantic Overlay Clusters in Super-Peer Networks. In *International Workshop on Databases, Information Systems and Peer-to-Peer Computing in Conjunction with the 29th VLDB*, Berlin, Germany, 2003.

[LNWS03] Alexander Löser, Wolfgang Nejdl, Martin Wolpers, and Wolf Siberski. Information Integration in Schema-Based Peer-To-Peer Networks. In *15th International Conference of Advanced Information Systems Engieering (CAiSE 03)*, Klagenfurt, June 2003.

[LS99] O. Lassila and R. Swick. W3C Resource Description framework (RDF) Model and Syntax Specification, 1999. `http://www.w3.org/TR/REC-rdf-syntax/`.

[Lös04a] Alexander Löser. Data Source Discovery in Educational P2P Networks. In Wladimir Uskov, editor, *Proceedings of the IASTED Intl. Conference on Web-Based Education (WBE)*. ACTA Press, 2004.

[Lös04b] Alexander Löser. Towards Taxonomy-based Routing in P2P Networks. In *The Second Workshop on Semantics in Peer-to-Peer and Grid Computing at the 13. WWW Conference*, New York, 2004.

[LST05] Alexander Löser, Steffen Staab, and Christoph Tempich. Semantic Methods for P2P Query Routing. In *3rd. German Conference on Multi Agent Technologies (MATES)*, volume 3550 of *LNAI*. Springer, 2005.

[LSZ04] Alexander Löser, Kai Schubert, and Frederik Zimmer. Taxonomy-Based Routing Overlays in P2P Networks. In *8th International Database Engineering and Applications Symposium (IDEAS 2004)*, IEEE Computer Society, 2004.

[LT05] Alexander Löser and Christoph Tempich. On Ranking Peers in Semantic Overlay Networks. In *3rd Conference on Professional Knowledge Management, PAIKM'05 Workshop*, 2005.

[LTQ+05] Alexander Löser, Christoph Tempich, Bastian Quilitz, Wolf-Tilo Balke, Steffen Staab, and Wolfgang Nejdl. Searching dynamic communities with personal indexes. In *3rd. International Semantic Web Conference (ISWC) Galway*, 2005.

[LWSN03] Alexander Löser, Martin Wolpers, Wolf Siberski, and Wolfgang Nejdl. Efficient data store discovery in a scientific P2P network. In *Interntl. Workshop on Semantic Web Technologies for Searching and Retrieving Scientific Data, International Semantic Web Conference (ISWC 2003)*, Sunibel Island, Florida, USA, October 2003.

[Mar02] Evangelos P. Markatos. Tracing a large-scale peer to peer system: an hour in the life of gnutella. In *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2002.

[MCR03] Manuel Costa Miguel Castro and Antony Rowstron. Should we build Gnutella on a structured overlay? In *2nd Workshop on Hot Topics in Networks (HotNets-II) Workshop Program*. ACM SIGCOMM, 2003.

[Mil67] Stanlay Milgram. The small world problem. *Psychology Today*, 67(1), 1967.

[MKL+02] Dejan S. Milojicic, Vana Kalogeraki, Rajan Lukose, Kiran Nagaraja, Jim Pruyne, Bruno Richard, Sami Rollins, and Zhichen Xu. Peer-to-peer computing. Technical report, HP Laboratories Palo Alto, Technical Report HPL-2002-57, 2002.

[MSZ03] Shashidhar Merugu, Sridhar Srinivasan, and Ellen Zegura. Adding Structure to Unstructured Peer-to-Peer Networks: The Role of Overlay Topology. In *Group Communications and Charges: Technology and Business Models. ICQT 2003 Proceedings*, volume 2816. Springer Verlag, 2003.

[MT02] Vijay S. Mookerjee and Yong Tan. Analysis of a least recently used cache management policy for web browsers. *Oper. Res.*, 50(2):345–357, 2002.

[Nap03] Napster. Napster company website. http://www.napster.com, 2003.

[Net04] Groove Networks. Groove Company Website. http://www.groove.net, 2004.

[NS02] C. Ng and K. Sia. Peer clustering and firework query model. In *Poster Proceedings of 11th World Wide Web Conference, May 2002.* ACM Press, 2002.

[NT95] I. Nonaka and H. Takuechi. *The Knowledge Creating Company*. Oxford University Press, 1995.

[NWQ+02] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér, and T. Risch. EDUTELLA: a P2P Networking Infrastructure based on RDF. In *Eleventh International World Wide Web Conference (WWW2002)*, Hawaii, USA, May 2002.

[NWS+03] Wolfgang Nejdl, Martin Wolpers, Wolf Siberski, Alexander Löser, Ingo Bruckhorst, Mario Schlosser, and Christoph Schmitz. Super-Peer-Based Routing and Clustering Strategies for RDF-Based Peer-To-Peer Networks. In *Twelfth International World Wide Web Conference (WWW2003)*, Budapest, Hungary, May 2003.

[NWS+04] Wolfgang Nejdl, Martin Wolpers, Wolf Siberski, Chrispoph Schmitz, Mario Schlosser, Ingo Brunkhorst, and Alexander Löser. Super-Peer-Based Routing Strategies for RDF-Based Peer-to-Peer Networks. *In Elsevier's Journal of Web Semantics*, 2004.

[OCW03] MIT OpenCourseWare a free, open, publication of MIT Course Materials., 2003. `http://ocw.mit.edu/index.html`.

[OMA04] Open Mobile Alliance OMA. DRM Architecture Version 2.0. Technical report, July 2004.

[Ora01] Andy Oram, editor. *Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology*. O´Reilly, Sebastopol (CA), 2001.

[RB03] M. Roussopoulos and M. Baker. Cup: Controlled update propagation in peer to peer networks. In *Proceedings of the 2003 USENIX Annual Technical Conference*, June 2003.

[RBR+04] Mema Roussopoulos, Mary Baker, David S. H. Rosenthal, Petros Maniatis TJ Giuli an, and Jeff Mogul. 2 p2p or not 2 p2p? In *Proceedings of the Third International Workshop on Peer-to-Peer Systems*, , San Diego, CA, USA, February 2004.

[RD01] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. *Lecture Notes in Computer Science*, 2218:329–350, 2001.

[RFH+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content addressable network. In *Conference on applications, technologies, architectures, and protocols for computer communications*. ACM Press New York, NY, USA, 2001.

[Rit01] Jordan Ritter. Why gnutella can't scale. Technical report, Darkridge, Inc., 2001.

[RKCD01] Antony I. T. Rowstron, Anne-Marie Kermarrec, Miguel Castro, and Peter Druschel. SCRIBE: The design of a large-scale event notification infrastructure. In *Networked Group Communication*, pages 30–43, 2001.

[RLS+03] Ananth Rao, Karthik Lakshminarayanan, Sonesh Surana, Richard Karp, and Ion Stoica. Load balancing in structured p2p systems. In *International Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.

[Sau03] Leo Sauermann. The Gnowsis, Using Semantic Web Technologies to build a Semantic Desktop. Diplomarbeit, Technische Univeristät Wien, 2003.

[SCK03] Mario T. Schlosser, Tyson E. Condie, and Sepandar D. Kamvar. Simulating a File-Sharing P2P Network. In *First Workshop on Semantics in P2P and Grid Computing, 12th. WWW Conference, Budapest*, 2003.

[SGG03] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. *Multimedia Systems*, 9(2), 2003.

[SHB97] Ruud Schoonderwoerd, Owen Holland, and Janet Bruten. Ant-like agents for load balancing in telecommunications networks. In *AGENTS '97: Proceedings of the first international conference on Autonomous agents*, pages 209–216. ACM Press, 1997.

[SMK+01] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM '01: Conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.

[SMZ03] Kunwadee Sripanidkulchai, Bruc Maggs, and Hui Zhang. Efficient Content Location Using Interest Based Locality in Peer-to-Peer System. In *Infocom*. IEEE, 2003.

[Sri01] Kunwadee Sripanidkulchai. The popularity of gnutella queries and its implications on scalability. Technical report, Technical Report, Carnegie Mellon University, February 2001.

[SSDN02] Mario Schlosser, Michael Sintek, Stefan Decker, and Wolfgang Nejdl. HyperCuP—Hypercubes, Ontologies and Efficient Search on P2P Networks. In *International Workshop on Agents and Peer-to-Peer Computing*, Bologna, Italy, July 2002.

[TAA+03] Bernard Traversat, Ahkil Arora, Mohamed Abdelaziz, Mike Duigou, Carl Haywood, Jean-Christophe Hugly, Eric Pouyoul, and Bill Yeager. Project JXTA 2.0 Super-Peer Virtual Network. Technical report, Sun Microsystems Inc., 2003.

[TEF+04] Christoph Tempich, Marc Ehrig, Christiaan Fluit, Peter Haase, Esteve Llado Marti, Michal Plechawski, and Steffen Staab. Xarop: A midterm report in introducing a decentralized semantics-based knowledge sharing application. In *Practical Aspects of Knowledge Management: 5th International Conference, PAKM 2004, Vienna, Austria*, number 3336 in Lecture Notes in Computer Science, 2004.

[tPW05] SnoCap Peer to Peer Website. http://www.snocap.com/, 2005.

[Tra04] Bernard Traversat. JXTA Technology: Creating Connected Communities. Technical report, Sun Microsystems, Inc., 2004.

[TSW04] Christoph Tempich, Steffen Staab, and Adrian Wranik. REMINDIN:Semantic Query Routing in Peer-to-Peer Networks based on Social Metaphers. In *Proceedings of the 13th WWW Conference New York*. ACM, 2004.

[VKMvS04] Spyros Voulgaris, Anne-Marie Kermarrec, Laurent Massoulie, and Maarten van Steen. Exploiting semantic proximity in peer-to-peer

content searching. In *International Workshop on Future Trends in Distributed Computing Systems (FTDCS)*, 2004.

[Vz99] P. Valduriez and M. Özsu. *Principles of distributed database systems*. Prentice Hall, 2nd edition edition, 1999.

[W3C03] W3C. Xpath definition, technical report. http://www.w3c.org/TR/xpath, 2003.

[Wat03] Duncan J. Watts. *Six Degrees - The Science of a Connected Age.* Norton and Company, 2003.

[Web04] Herbert Weber. Future net. *Informatik Spektrum*, 27(1), 2004.

[YGM03] Beverly Yang and Hector Garcia-Molina. Designing a super-peer network. In *International Conference on Data Engineering (ICDE)*, March 2003.

[YRS02] Y.Chawathe, S. Ratnasamy, and S. Shenker. Can heterogeneity make gnutella scale? http://research.att.com/ yatin/publications/, May 2002.

[Zim04] Frederick Zimmer. Effziente Routingstrategien für Semantic Overlay Cluster. Master's thesis, CIS, Technische Universität Berlin, 2004.

[Zip49] G. K. Zipf. *Human Behaviour and the Principle of Least Effort*. Addison-Wesley Press, 1949.

[ZKJ01] B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, Computer Science Division, U. C. Berkeley, April 2001.