# Kontextbezug und Authentizität in Sozialen Netzen

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität München

vorgelegt von
Mirco Schönfeld
2016

# Kontextbezug und Authentizität in Sozialen Netzen

**Mirco Schönfeld**

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität München

vorgelegt von
Mirco Schönfeld


| | |
|---|---|
| 1. Berichterstatter: | Prof. Dr. Claudia Linnhoff-Popien |
| 2. Berichterstatter: | Prof. Dr. Alexander Schill |
| Tag der Einreichung: | 21. September 2016 |
| Tag der Disputation: | 22. November 2016 |

## Eidesstattliche Versicherung

(siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. 5)

Hiermit erkläre ich an Eides statt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Mirco Schönfeld

# Danksagung

Diese Arbeit ist während meiner Zeit am Lehrstuhl für Mobile und Verteilte Systeme der Ludwig-Maximilians-Universität München entstanden. In dieser Zeit habe ich viel Unterstützung erfahren, für die ich mich an dieser Stelle bedanken möchte.

An erster Stelle bedanke ich mich bei Prof. Dr. Claudia Linnhoff-Popien, die es mir ermöglicht hat, meine wissenschaftliche Tätigkeit an ihrem Lehrstuhl zu beginnen. Sie hat den Raum geschaffen für lehrreiche, motivierende Diskussionen, eine produktive, freundschaftliche Arbeitsatmosphäre und spannende, fruchtbare Kooperationen, in denen ich meine Interessen und mein Thema finden und verfeinern konnte.

Danken möchte ich auch Prof. Dr. Alexander Schill, der sich bereit erklärt hat, meine Arbeit als Zweitgutachter zu betreuen. Seine interessanten Fragen und wertvollen Hinweise werden mich auch über diese Arbeit hinaus beschäftigen.

Mein Dank gilt auch Prof. Dr. Heinrich Hußmann, den ich zu Beginn meines Studiums kennengelernt habe und der freundlicherweise den Vorsitz meiner Prüfungskommission übernommen hat.

Bedanken möchte ich mich auch bei allen Kollegen vom Lehrstuhl, die mich vom ersten Tag an herzlich aufgenommen und mich mit ihrem Humor, ihrem fachlichen Interesse und dem Spaß an der Zusammenarbeit begleitet haben. Dadurch sind einige Freundschaften entstanden, die auch über das Lehrstuhlleben hinaus bestehen bleiben werden. So geht mein herzlicher Dank unter anderem an Sebastian Feld, Dr. Martin Werner, Dr. Kevin Wiesner, Dr. Marco Maier, Florian Dorfmeister und Lorenz Schauer.

Durch die Höhen und Tiefen der Promotion haben mich auch meine Freunde außerhalb des Lehrstuhls begleitet und mit diversen gemeinsamen Freizeitaktivitäten einen hervorragenden Ausgleich geschaffen. Darüber hinaus danke ich Tobias Ott und Dominik Busching, die mich beide auf ihre Art stets unterstützt haben.

Mein besonderer Dank gilt meinen Eltern und meiner Schwester, die nie am positiven Ausgang meiner Promotion gezweifelt haben.

Und schließlich bedanke ich mich herzlich bei meiner Freundin Dr. Silvia Fernández Rodríguez, deren Vertrauen, Liebe und wundervolles Wesen ich nicht missen möchte.

# Zusammenfassung

Soziale Netze gehören zu den wichtigsten Anwendungen des World Wide Web. Freunde vernetzen und organisieren sich, tauschen Bilder aus und erzählen in Statusnachrichten aus ihrem Alltag. In heutigen Sozialen Netzen gehen diese Statusnachrichten meist an die gesamte Freundesliste – eine Einschränkung der Adressaten ist zwar möglich, aber so umständlich, dass es kaum genutzt wird. Ein typischer Nutzer pflegt in seiner Freundesliste allerdings mehrere soziale Gruppen gleichzeitig. Das können etwa verschiedene Freundeskreise, Arbeitskollegen und Bekannte aus Sportvereinen sein. Beim Verfassen seiner Statusnachrichten hat dieser Nutzer zwar einen bestimmten Adressatenkreis vor Augen; er versendet seine Mitteilung trotzdem an seine gesamte Freundesliste. Die Grenzen der sozialen Gruppen verschwimmen, man spricht vom sogenannten *context collapse*. Um daraus resultierende Konflikte und Spannungen in den Freundeskreisen zu verhindern, haben Nutzer eine einfache Strategie entwickelt: sie zensieren sich selbst; eine authentische Kommunikation ist so kaum möglich.

Soziale Netze mit Kontextbezug widmen sich diesem zentralen Missstand und schaffen einen Rahmen für Kommunikation, der der Unterhaltung von Angesicht zu Angesicht ähnlich ist und der von heutigen sozialen Netzen so nicht abgebildet werden kann. Es handelt sich dabei um soziale Netze, deren sozialer Graph nur auf der Basis von Kontextinformationen entsteht. Profile können beispielsweise nur dann Informationen austauschen, wenn sie ähnliche Kontextdeskriptoren hatten oder haben. Solche kontextbezogenen sozialen Netze stellen zunächst eine spezielle Form von informationszentrischen Netzwerken dar. Informationsobjekte werden darin an Kontexte adressiert; diejenigen Profile werden zu Empfängern, die sich in dem entsprechenden Kontext befinden oder befunden haben. Diese kontextabhängige Adressierung erlaubt wiederum einen vollständig verteilten Aufbau eines solchen Sozialen Netzwerks. In der vorliegenden Arbeit wird dieses neuartige Paradigma detailliert beschrieben.

Damit eine Kommunikation aufgrund gemeinsamer Kontexte überhaupt möglich ist, müssen Kontexte adressierbar werden. Darüber hinaus sollten die Adressen einen Vergleich der zugrunde liegenden Kontexte zulassen. Denn der praktische Nutzen wäre erheblich eingeschränkt, wenn Informationen nur innerhalb exakt gleicher Kontexte ausgetauscht werden könnten. Das hieße, dass genau definiert werden müsste, welche Informationen zur Beschreibung des aktuellen Kontextes eines Profils zulässig wären und welche nicht.

In diesem Zusammenhang wird deshalb ein generisches Adressierungsschema entwickelt, das ohne diese Einschränkung auskommt. Es wird gezeigt, wie

Kontexte mit Hilfe von Bloom-Filtern adressiert werden können und wie über Bloom-Filter ein Vergleich der beschriebenen Kontexte möglich ist. Das so konzipierte Matching von Kontexten wird umfangreich evaluiert. Dazu wird auf der Datenbasis eines öffentlich zugänglichen sozialen Netzes aus eMail-Kommunikationen ein kontextzentrisches soziales Netz simuliert. Die zugrundeliegenden sozialen Graphen des eMail- und des kontextzentrischen Netzwerks werden mit Methoden der sozialen Netzwerkanalyse verglichen. Es zeigt sich, dass mit kontextzentrischer Adressierung vergleichbare Strukturen im sozialen Graphen entstehen.

Weitreichende Implikationen für die Anwendbarkeit von kontextbezogenen sozialen Netzen ergeben sich, wenn Kontextinformationen zwischen Nutzern verwendet werden können, ohne diese tatsächlich gegenseitig zu offenbaren. Dann werden neben Beweisen der Kontextgleichheit auch kooperative statistische Erhebungen über den Datenbestand mit Garantien an die Privatsphäre möglich. Dies wäre einfach, wenn man eine vertrauenswürdige dritte Partei (Trusted Third Party) zuließe. Um diese Schwachstelle zu vermeiden, werden zum einen Algorithmen vorgeschlagen, die einen netzweiten Konsens über Werte der einzelnen Teilnehmer herstellen können, ohne deren Eingabedaten zu offenbaren. Zum anderen können Ansätze zum privatsphäreschonenden Ähnlichkeitsbeweis von Kontextinformationen verwendet werden.

# Abstract

Online Social Networks (OSNs) are amongst the most important applications in the World Wide Web. Friends connect, share photos and keep themselves up-to-date about their daily lifes via status messages. These messages are most likely to be broadcasted to all people in a user's friendlist – selecting a specific audience is possible, indeed, but cumbersome. Broadcasting status messages, however, leads to a so called *context collapse* – a collapse of different circles of friends users usually collect in their friendlists, simultaneously. To prevent conflict and tension, users have developed a simple strategy to cope with context collapse: they bowdlerize and censor their status messages; authentic communication among friends is barely possible.

Context-centric OSNs focus on this exact drawback. They create a setting for communication that is comparable to face-to-face-communications with respect to authenticity of communicated content. These novel kind of OSNs modify the underlying social graph based on context information. Profiles are able to exchange information if and only if they share some common context descriptions. Thereby, context-centric OSNs represent a special kind of fully distributed information-centric networks where information objects are addressed to contexts. This novel addressing scheme and paradigma for OSNs is presented in depth.

A main requirement for this scheme to be practical is making contexts addressable. Further, addresses should allow comparing underlying contexts to prevent the neccessity of exact coinciding contexts for message exchange and to prevent specification of which context information to use. In this thesis a novel addressing scheme will be presented that fulfills these requirements and that is based on Bloom filters.

This matching of contexts is evaluated in depth by simulating a context-centric OSN from a publicly available dataset of eMail-communications. The underlying social graphs of eMail-communications and context-centric networking are compared with respect to well-known metrics from social network analysis.

Extensive implications for the applicability of context-centric OSNs emerge from a privacy-perspective towards the proposed addressing scheme. If context similarity can be used without revealing any context information the scheme allows for mutual proof of shared context as well as for statistic inquiry of data. Therefore, algorithms for privacy-preserving data mining are proposed that lack a trusted third party to prevent a central weak spot for the network to develop.

# Inhaltsverzeichnis

# 1 Einführung

„Im Internet weiß niemand, dass Du ein Hund bist“ – so untertitelte der Kartoonist Peter Steiner eine im Jahr 1993 im Magazin The New Yorker veröffentlichte Karikatur. Die Zeichnung zeigt einen chattenden Hund, der mit diesen Worten einem anderen Hund das Internet zu erklären scheint.


*"On the Internet, nobody knows you're a dog."*

Abbildung 1.1: „On the Internet, nobody knows you are a dog“, The New Yorker, 5. Juli 1993

Diese Karikatur wurde die meistzitierte Zeichnung des Magazins und markiert einen wichtigen Moment in der Geschichte des Internets. Das verließ damals allmählich die wissenschaftlichen Sphären und begann, eine breite Öffentlichkeit zu erobern. Das neue Publikum war sofort fasziniert von dem Gedanken, im Internet *anonym* zu kommunizieren. Die ersten rudimentären *Chatrooms*, die beim Betreten lediglich nach einem frei wählbaren Pseudonym fragten, sollten Kategorien wie Alter, Geschlecht, Rasse und Aussehen endlich überwinden helfen. Schließlich wurde die versprochene Anonymität auf keiner

Ebene gebrochen – nicht einmal das *Internet Protokoll* erforderte eine eindeutige Identifizierung. In der Vorstellung von damals hätte wirklich niemand gewusst, dass er mit einem Hund chattet.

In den frühen Zeiten des Internets, in denen weniger ökonomische Interessen und juristische Fallstricke lauerten, schien das Versprechen von vollkommener Anonymität – zumindest in der wissenschaftlichen Wahrnehmung – einen Hang zum Ausprobieren und Experimentieren mit der eigenen Identität zu provozieren. Personen schienen sich online zu inszenieren und die Grenzen des eigenen Selbst auszuloten. Für die Online-Identitäten etablierte sich schnell der Begriff der *persona*, ein römisch-antiker Terminus, der eine Charaktermaske bezeichnet. Eine Persona ist stets ein „symbolisches Konstrukt, das dem Selbstverständnis und dem Imaginären einer Person über sich selbst entspringt“ ([1], S.183). Sie ermöglicht erst ein Spiel mit virtuellen Identitäten, das, so die wissenschaftliche Argumentation, unvermeidlich zu einem Verlust an Authentizität der Kommunikatoren führe.

Diesem Gedankengang liegt ein weniger technisches Verständnis von Authentizität zugrunde. In der Informatik und insbesondere in der Informationssicherheit bezeichnet Authentizität die nachweisbare Urheberschaft einer Nachricht, die häufig durch digitale Signaturen für jeden Empfänger einer Nachricht überprüfbar wird. Eine digitale Signatur kann nur von einem bestimmten Schlüssel erzeugt worden sein. Indem die Signatur mit dem öffentlich zugänglichen Teil des Schlüssels des in einer Nachricht angegebenen Absenders überprüft wird, kann die Authentizität dieser Nachricht bestätigt werden, zumindest wenn der angegebene Absender auch der tatsächliche Absender ist. Hier wird Identität also mit dem Besitz eines eindeutigen Schlüsselpaares gleichgesetzt. Die philosophisch-ethische Perspektive auf Authentizität ist von ihrer technischen Umsetzung nicht weit entfernt.

In der Philosophie ist Authentizität keine Eigenschaft eines Gegenstandes, sondern die Wahrheit einer dem Gegenstand zugeschriebenen Eigenschaft. Demnach ist etwa eine Urkunde nicht an sich authentisch, sondern sie wird authentisch, wenn – und nur wenn – die Auskunft des Ausstellers über sich selbst der Wahrheit entspricht. Die Analogie zum Authentizitätsverständnis der Informationssicherheit liegt auf der Hand.

In der Ethik setzt sich der Begriff aus den beiden Bausteinen „Eigentlichkeit“ und „Wahrhaftigkeit“ zusammen. Während die Eigentlichkeit sich auf eine „angemessene Repräsentation in einem Medium und die angemessene Form der Rezeption dieses Mediums“ ([2], S.20) bezieht, ist mit Wahrhaftigkeit eine Intention oder Neigung gemeint, Wahrheit aussagen zu wollen. Authentizität entsteht nur, wenn beide Aspekte zusammenfinden. In diesem Zusammenhang bezieht sich auch in der ethischen Sicht die Wahrhaftigkeit nicht auf den Inhalt einer Nachricht, sondern auf die „Behauptung einer bestimmten Identität“ (ebd.). Ein Kommunikator ist also authentisch, wenn er in Bezug auf seine Identität die Wahrheit spricht. Der inhaltliche Wahrheitsgehalt seiner Aussagen ist davon unabhängig.

In Bezug auf seine Identität die Wahrheit zu sprechen – hinter dieser Formulierung verbirgt sich, dass Besitz und Behauptung einer Identität nicht zwingend zusammenfallen. Doch was heißt es überhaupt, in Bezug auf seine Identität die Wahrheit zu sprechen? Damit kann kaum gemeint sein, stets mit korrektem Namen auftreten zu müssen. In der *Face-to-Face*-Kommunikation stellen wir uns einander schließlich nicht mit unseren Ausweisen vor. Wir entwickeln vielmehr ein Gefühl für die Authentizität unseres Gegenübers. Es verhält sich wie mit dem chattenden Hund – im Internet könnte auch dieser als ein authentischer Gesprächspartner wahrgenommen werden. In diesem Zusammenhang ist mit Identität also etwas anderes als die rechtliche, objektiv überprüfbare Identität gemeint.

## 1.1 Das Spiel mit der eigenen Identität

Die Identität eines Menschen ist nicht einfach „da“, sie ist keine feste Größe, der man mit seinem Denken und Handeln jederzeit entsprechen könnte. Die Identität eines Menschen formt und entwickelt sich stetig weiter, indem situative Erfahrungen „übersituativ verarbeitet und generalisiert werden“ ([3], S. 21). Sich seiner Identität klar zu werden, bedarf einer stetigen Auseinandersetzung mit dem eigenen Selbst und der eigenen Umwelt. Damit ist ein Prozess beschrieben, der eine „aktive Identitätsarbeit“ ([4], S.339) erfordert. Dieser Prozess beinhaltet neben einer Selbstvergewisserung auch die Verarbeitung „biographischer Umbrüche“ oder die „Interpretation und Akzeptanz der Widersprüche im Selbst“ (ebd.).

Soziales Handeln und Kommunikation sind für diese Identitätsarbeit unabdingbar. Nur dadurch wird die Identität als wichtiger Teil der Persönlichkeit in allen Facetten ausdifferenziert. Dazu gehört, dass man im sozialen Handeln und in Kommunikationssituationen seine Identität zu Erkennen gibt, sie gewissermaßen dem Gegenüber darstellt. Allerdings haben nicht alle Identitätsaspekte in jeder Situation die gleiche Bedeutung: In Unterhaltungen über klassische Musik wird vor allem die eigene Einstellung und Haltung zu diesem Thema herausgestellt und verfeinert, während andere Aspekte wie beispielsweise motorische Fähigkeiten beim Kuchen backen oder demografische Informationen wie die familiäre Situation in den Hintergrund rücken. Welche Aspekte betont werden und welche nicht, ist häufig strategisch motiviert, um etwa als zu einer Gruppe dazugehörig wahrgenommen zu werden oder nicht.

Während Identität also ein selbstreflexiver Prozess des Aushandelns zwischen Innen- und wahrgenommener Außenperspektive ist, ist der vom strategischen Moment geprägte Prozess, mit dem die eigene Identität nach außen dargestellt wird, die Selbstdarstellung [3], [5]. Sie ist ein auf die eigene Identität bezogenes, „taktisches oder strategisches Verhalten“ ([4], S.340). Sie mag negative Assoziationen hervorrufen, weil damit häufig Eitelkeit oder Manipulation assoziiert werden. Im eigentlichen Wortsinn und nach der Definition von Schlenker bezeichnet die Selbstdarstellung aber alle bewussten und unbewuss-

ten Aspekte, mit denen das eigene Selbst einem Gegenüber vermittelt und mit dem einem Gegenüber ein authentisches Bild der eigenen Person zu präsentieren versucht wird [6]. Die Strategien der Selbstdarstellung reichen dabei von kleineren Änderungen im Vokabular bis hin zu signifikanten Korrekturen, etwa politischen Einstellungen [7].

An dieser Stelle offenbart sich der Zusammenhang zwischen Identität und Authentizität besonders deutlich. Eine Person und ihre Selbstdarstellung wird als authentisch wahrgenommen, „wenn sie im Kern mit dem wahrgenommenen Selbst übereinstimmt“ ([4], S.342). Damit ist die Außenperspektive eines Gegenübers gemeint und die Authentizität etwas, das in der Wahrnehmung des Gegenübers entsteht. Doch auch auf der Seite des Sprechers muss eine größtmöglich Kongruenz im inneren Prozess der Selbstdarstellung herrschen, um überhaupt authentisch wirken zu können: „Vom Individuum selbst [wird] keine Kluft zwischen der Selbstdarstellung und den im Augenblick wichtigen Aspekten der Identität empfunden“ (ebd.). Es geht also auch um „eine größtmögliche Übereinstimmung zwischen Selbstdarstellung und der eigenen, wahrgenommenen Identität“ (ebd.).

Doch wovon hängt nun ab, wie das eigene Selbst dargestellt wird? Wie wird der Prozess der Selbstdarstellung gesteuert? Dazu konnte Erving Goffmann schon 1959 zeigen, dass Individuen eine Reihe bewusster Entscheidungen treffen, die maßgeblich davon abhängen, mit wem sie jeweils kommunizieren [8]. Je nach Publikum selektiert ein bewusst ablaufender Prozess unterschiedliche Identitätsaspekte, die herausgestellt oder verborgen werden. Schlenker ermittelte in darauf aufbauenden Untersuchungen den Kontext und die Umwelt der Kommunikationssituation als zusätzliche Faktoren [9]. Präzisiert wurden diese Untersuchungen von Mark Leary, der die These aufstellt, dass die Selbstdarstellung von den Ansichten und Werten des Publikums beeinflusst wird oder sie bestimmte Reaktionen beim Gegenüber provozieren soll [10].

Die Darstellung der eigenen Identität ist also stets Umwelteinflüssen und bewussten Entscheidungsprozessen unterworfen. Die eigene Identität kann so ausprobiert und ihre Grenzen können ausgelotet werden. Partielle Maskierungen und sogar kleine Täuschungen führen dabei nicht zur „Konstruktion einer fremden oder falschen Identität“ ([4], S.342); vielmehr geht es um ein spielerisches Moment der Persönlichkeitsbildung.

Für die Kommunikation im Internet wurde der spielerische und strategische Aspekt lange Zeit besonders hervorgehoben. Der chattende Hund etwa sollte hinter der Anonymität der Online-Kommunikation seine wahre Identität verstecken, sich neue Identitäten konstruieren und trotzdem als authentischer Kommunikator wahrgenommen werden können. Der wissenschaftliche Diskurs der 1990er Jahre hat sich lange und intensiv mit dieser Idee auseinandergesetzt (vgl. [1], [2], [4] für eine Übersicht über die spezifischen Untersuchungen). Die neuen Plattformen der Online-Kommunikation und -Selbstdarstellung erfordern allerdings eine Neuausrichtung der Betrachtungen. Während die frühen Online-Medien, wie *Instant Messenger*, auf eine Eins-zu-eins-Kommunikation

ausgerichtet waren, und damit die Rahmen der Kommunikationssituation relativ klar gesteckt waren, bieten heutige Plattformen wie Twitter, Facebook und LinkedIn [226]–[228] vor allem eine Kommunikation mit einem unsichtbaren Publikum. Dies hat weitreichende Konsequenzen für die authentische Selbstdarstellung von Individuen im Internet.

## 1.2 Das Problem sich überschneidender Öffentlichkeiten

Was die Darstellung ihrer Persönlichkeit im Internet angeht, agieren Menschen sehr absichtsvoll. Diese Erkenntnis konnte in einer Studie von Nina Haferkamp gewonnen werden, nachdem die Forschung diesbezüglich eher widersprüchliche Ergebnisse geliefert hatte (vgl. [11]). So zitiert Haferkamp einserseits Untersuchungen, die darauf hindeuten, dass Personen Profilinformationen in Sozialen Netzen bewusst verfälschen, um ihre Außenwirkung auf eine intendierte Art zu beeinflussen. Andererseits führt sie Studien an, die vermuten lassen, dass die „Selbstdarstellung in Sozialen Netzwerken im Internet relativ eng an die Selbstdarstellung in der Alltagskommunikation angelehnt“ ([11], S.182) ist.

Die Absichten der Selbstdarstellung im Internet und dort insbesondere in Sozialen Netzen wie Facebook sind stets besonderen Dynamiken unterworfen, die Soziale Medien von traditionellen Interaktionsformen fundamental unterscheiden und die im Folgenden genauer beschrieben werden: ein unsichtbares Publikum, die verschwimmende Grenze zwischen Öffentlichem und Privatem und der sogenannte *context collapse* [12]. Die aktuelle Forschung fokussiert sich dabei auf die sichtbarste Form der Selbstdarstellung: Nachrichten, die über öffentliche Kommunikationskanäle veröffentlicht werden – sogenannte Status-Updates. Solche Nachrichten haben die größtmögliche Wahrscheinlichkeit, vom gesamten Netzwerk einer Person gesehen zu werden. Und wegen ihres selbstoffenbarenden Inhalts sind sie eine „essentielle Komponente der Aufrechterhaltung sozialer Beziehungen“ ([7], S.453).

Das Publikum solcher Statusnachrichten ist allerdings selten genau bestimmbar. Zwar werden diese Nachrichten meist an die Personen adressiert, zu denen eine Beziehung besteht, die also Teil der eigenen „Freundesliste“ sind. Dennoch ist es möglich, dass auch Freundesfreunde Nachrichten zu Gesicht bekommen. Und gleichzeitig kann es sein, dass Auswahlmechanismen bestimmte Nachrichten von Freunden gezielt verbergen – diese Nachrichten tauchen dann nicht mehr im *News-Feed* von Facebook auf und erreichen also nicht zangsläufig alle Freunde der eigenen Freundesliste.

Daraus resultiert unter anderem, dass die Grenze zwischen öffentlicher und privater Kommunikation zunehmend verschwimmt. Für die Verfasser von Statusnachrichten ist kaum zu kontrollieren, wen diese Nachrichten erreichen – gleichwohl werden solche Nachrichten häufig mit einer ganz bestimmten Vorstellung des intendierten Adressatenkreises veröffentlicht. Eine klare Grenze

zwischen öffentlicher und privater Kommunikation zu ziehen, kostet den Verfasser einer Nachricht allerdings Mühe und Aufwand. Zum einen muss der Verfasser sich über die Grenze bewusst werden, und zum anderen muss er diese Grenze dann in Form von Sichtbarkeitseinstellungen für jede Nachricht neu ziehen. Daher erscheint es in Bezug auf die benötigte Zeit, benötigtes Wissen und Fähigkeiten einfacher, Nachrichten für das gesamte Netzwerk zu veröffentlichen, obwohl Individuen sich bewusst sind, dass das einige Beziehungen womöglich negativ beeinflusst ([7], S.455).

Das wiederum ist ein Effekt des *context collapse*, also dem Zusammenfall kontextueller Unterschiede zwischen sozialen Gruppen des persönlichen Netzwerks. Heutige Soziale Netzwerke bieten keine praktikable Möglichkeit, die eigenen Kontaktlisten zu gruppieren. Wie bereits ausgeführt wurde, hängt die Art der Selbstdarstellung stark von dem adressierten Publikum ab. In der *Face-to-Face*- oder Eins-zu-eins-Kommunikation sind unterschiedliche soziale Gruppen stets klar voneinander getrennt. Die technischen Gegebenheiten heutiger Sozialer Netze sorgen allerdings dafür, dass das Publikum der Statusnachrichten zu einer homogenen Gruppe zusammenfällt – die zeitliche, räumliche und soziale Trennung unterschiedlicher Gruppen also verschwindet [7], [12].

Es ist dieser Zusammenfall der offline voneinander getrennten Sozialen Gruppen, der für Spannungen innerhalb dieser Gruppen und für latenten Stress unter den Benutzern sorgt. In [13] haben Binder et al. gezeigt, dass die öffentlich sichtbaren Kommunikationen, zu denen Nutzer in Sozialen Netzen angeregt werden, soziale Bindungen einer Person zu ihren unterschiedlichen Freundeskreisen destabilisieren können, je diverser das persönliche Netzwerk dieser Person ist. Die Autoren nennen dies das *problem of conflicting social spheres*, also das Problem sich überlappender Öffentlichkeiten. Es geht ihnen weniger um Auswirkungen auf die Privatsphäre einzelner Personen, etwa wenn kompromittierende Fotos für einen Fremden sichtbar werden. Mit dem Problem sich überschneidender Öffentlichkeiten beziehen sie sich auf Nachrichteninhalte, die nur innerhalb einer bestimmten sozialen Gruppe sichtbar sein sollten, die aber in anderen Sozialen Gruppen verfügbar werden.

Noch bevor es zu unbeabsichtigten Veröffentlichungen in „falschen“ Sozialen Gruppen kommt, sind sich die Nutzer alleine der Möglichkeit, den falschen Adressatenkreis zu erreichen, ständig bewusst. Diese Kopräsenz verschiedener Gruppen oder Freundeskreise zu „verwalten“, verursacht unterschwellig und beständig mentale Arbeit. Um mit dieser Arbeit und dem damit verbundenen latenten Stress umzugehen, entwickeln die Nutzer deshalb eine ganz bestimmte Strategie. Sie veröffentlichen weniger Statusnachrichten oder passen den Inhalt Ihrer Statusnachrichten dem „kleinsten gemeinsamen Nenner“ der kopräsenten Gruppen an. Kurzum: Sie zensieren sich selbst. Als Grund dieser Selbstzensur geben Nutzer explizit an, Konflikten vorbeugen und Situationen vermeiden zu wollen, die die eigene Identität gefährden. Sie halten bestimmte Inhalte zurück, weil sie davon ausgehen, dass diese Inhalte das eigene Selbst nicht in der gewünschten Weise repräsentieren [14], [15].

Zahlreiche Untersuchungen belegen, dass Nutzer – insbesondere von Facebook – einen Weg suchen, ihre Sozialen Gruppen zu trennen. Facebook bietet dazu zwar die Funktion, die Freundesliste in Gruppen einzuteilen. Diese wird aber nur von wenigen Nutzern ernsthaft genutzt. Es scheint ebenso viele Nutzer zu geben, die die Adressaten ihrer Freundeskreise explizit benennen, wie solche, die sich eventueller Probleme bewusst zu sein scheinen, die ihre Statusnachrichten dennoch an ihre gesamte Freundesliste „ausposaunen“ [7], [16]–[21]. Die Trennung Sozialer Gruppen wird wegen der technischen Gegebenheiten – zumindest am Beispiel von Facebook – als schwer machbar wahrgenommen [19]. Hinzu kommt, dass Nutzer sich offenbar ganz allgemein nicht mehrmals mit solchen Funktionen beschäftigen wollen – einmal gewählt, sollen Einstellungen auch in Zukunft sinnvoll sein und gültig bleiben und vor allem keine Aufmerksamkeit mehr erfordern [22].

Interessanterweise ist die gewünschte Kategorisierung der Freundesliste häufig naheliegend. Eine qualitative Befragung von Facebook-Nutzern ergab etwa, dass vor allem Arbeitskollegen oder Mitschüler separat adressiert werden sollten. Hinzu kamen Gruppen wie „Hockey-Freunde“ oder Leute, die einen ähnlichen Geschmack für spezielle Comic-Bücher haben [15]. Die Freundesliste soll also vor allem über bestimmte Eigenschaften gruppiert werden, die die befragten Nutzer mit ihren Freunden jeweils teilen. Solche beschreibenden Eigenschaften sind in der Informatik wohl bekannt und werden dort gerne auch als „Kontextinformationen“ bezeichnet.

## 1.3 Ziel und Aufbau dieser Arbeit

Mit der vorliegenden Arbeit wird ein Konzept vorgestellt, mit dem ein umfassender Kontextbezug für Soziale Netze hergestellt werden kann: Kontextzentrische Soziale Netze. In diesem Netzwerk werden Nachrichten an Kontexte adressiert. Empfangen werden Nachrichten, indem Kontexte abonniert werden; das wiederum ist nur möglich, wenn Empfänger und Sender einer Nachricht einen Kontext geteilt haben. Denn nur in einem geteilten Kontext können beide Kommunikationspartner Kontextbeschreibungen konstruieren, die zueinander ähnlich sind. Neu ist an diesem Konzept also unter anderem, dass Kommunikationspartner nur durch den Vergleich ihrer Kontextbeschreibungen vernetzt werden.

Im Kern basiert die kontextzentrische Vernetzung auf einer informationszentrischen Netzwerkarchitektur und einer generischen Beschreibung von Kontexten. Der Kontext wird in diesem Zusammenhang nicht wie bisher auf definierte Modalitäten begrenzt – neben Aufenthaltsorte können jetzt ebenso Arbeitgeber, Hobbies oder Vorlieben für bestimmte Comic-Bücher in eine Kontextbeschreibung aufgenommen werden. Die Sichtbarkeit von Nachrichten kann dadurch zeitlich, räumlich und sozial beschränkt werden. Die verwendeten Modalitäten können zudem erstmals zu späteren Zeitpunkten erweitert oder verändert werden, ohne dass die Vergleichsoperationen angepasst werden müssen.

Bei der Konstruktion der Kontextbeschreibungen wird die Privatsphäre der Nutzer geschützt, da aus einer Beschreibung nicht auf einzelne Kontextinformationen zurückgeschlossen werden kann. Der Vergleich zweier Kontextbeschreibungen setzt zudem keine vertrauenswürdige dritte Entität voraus, sondern kann prinzipiell von jedem Knoten im Netz durchgeführt werden. Damit eignet sich diese Architektur genauso für eine spontane *ad hoc*-Kommunikation wie für den großflächigen, pervasiven Einsatz für Soziale Netze.

Die Vorteile werden insbesondere im Hinblick auf Soziale Netze deutlich. Zum einen kann das Problem des unbekannten Publikums eliminiert werden, zum anderen ist dieser Vernetzung eine Trennung sozialer Gruppen inhärent und der *context collapse* wird vermieden. Indem Kommunikationspartner nur über gemeinsame Kontexte zusammenfinden, entstehen automatisch eigene Kommunikationskanäle für Soziale Gruppen. Durch diese Form der Vernetzung ist die gesamte Freundesliste quasi automatisch entsprechend der gemeinsam geteilten Kontexte gruppiert. Dadurch erlauben diese Kanäle eine authentische Kommunikation ohne Selbstzensur, die erstmals vergleichbar ist zur Offline-Kommunikation.

Das hier vorgestellte Konzept bietet auch die Möglichkeit zur Datenerhebung in Sozialen Netzen. Dazu wird zum einen gezeigt, wie allgemein anerkannte Methoden zur Analyse Sozialer Netze auch in kontextzentrischen Netzen eingesetzt werden können. Zudem werden Methoden zur Datenerhebung und Konsensbildung vorgestellt, die dem Gedanken der verteilten Netzarchitektur und der verteilten Datenhaltung Rechnung tragen und die Garantien an die Privatheit der Daten einzelner Nutzer geben können.

Die Anwendbarkeit der kontextzentrischen Architektur ist indes nicht auf Soziale Netze beschränkt. Die generische Kontextbeschreibung, der simple Kontextvergleich und nicht zuletzt die Methoden zur verteilten privatsphäreschonenden Datenerhebung erlauben einen Einsatz etwa im Internet der Dinge (*Internet of Things*) oder für dynamische Service-Kollaboration.

Die Arbeit gliedert sich wie folgt. In Kapitel 2 wird zunächst beschrieben, was Kontext und Kontextinformationen sind und wie diese bisher in Sozialen Netzen eingesetzt werden. Bevor das Konzept der kontextzentrischen Architektur detailliert beschrieben wird, werden in Kapitel 3 Bloom-Filter vorgestellt. Diese Datenstruktur wird zur generischen Beschreibung von Kontexten eingesetzt und bildet damit den Kern der kontextzentrischen Architektur. In Kapitel 4 wird die Architektur und ihr Potenzial für die authentische Online-Kommunikation beschrieben. Das Kapitel 5 beschäftigt sich schließlich mit der Anwendbarkeit der kontextzentrischen Architektur für ein echtes Soziales Netz. Es wird einerseits untersucht, ob mit der beschriebenen Architektur echte Soziale Strukturen abgebildet werden können. Andererseits beschäftigt sich das Kapitel mit der verteilten Erhebung von Profilinformationen mit Garantien an die Privatsphäre der Nutzer. Mit Kapitel 6 endet diese Arbeit.

## 1.4 Vorveröffentlichungen

Die vorliegende Dissertation basiert auf Ideen und Ergebnissen, die zum Teil bereits publiziert worden sind. So ist das Konzept der kontextzentrischen Sozialen Netze bereits in [241] veröffentlicht worden – einer gemeinschaftlichen Arbeit mit meinen Kollegen Dr. Martin Werner und Florian Dorfmeister. Darin ist das Konzept der generischen Kontextbeschreibung durch Bloom-Filter und die informationszentrische Vernetzung auf der Basis von Ähnlichkeitsvergleichen zwischen Bloom-Filtern beschrieben worden. Mein Beitrag zu dieser Veröffentlichung bestand insbesondere in der prototypischen Umsetzung des Konzepts (ca. 20%). Die Idee zur generischen Beschreibung von Kontexten mit Bloom-Filtern, die Dr. Werner zum kontextzentrischen Adressschema ausformuliert hat, ist während der gemeinsamen Arbeit an den Erweiterungen des Bloom-Filter-Protokolls entstanden.

Die zwei Varianten von Bloom-Filtern, die in den Abschnitten 3.1.2.1 und 3.1.3.2 vorgestellt werden, wurden in [242] und [245] veröffentlicht – beide in Zusammenarbeit mit Dr. Werner. In [245] wurde vorgeschlagen, OVSF-Codes mit Bloom-Filtern zu kombinieren, um baumlokale Falschpositive zu erhalten. Die Idee zu dieser Kombination stammt von Dr. Werner, mein Beitrag bestand in wesentlichen Teilen der Simulation und Evaluation (ca. 70%). Ähnlich verhält es sich mit den Gaußschen Bloom-Filtern, die in [242] erstmals beschrieben wurden. Hier bestand mein Beitrag insbesondere in der Integration und Auswertung des Verfahrens in der Big Data Datenbank Apache Cassandra (ca. 20%).

In Abschnitt 4.1.5 wird ein Ausblick zum kontextzentrischen Adressierungsschema formuliert. Dieser basiert auf den Vorarbeiten zu einem hybriden Radionetzwerk und der virtuellen Währung Bitcoin [240], [237], [243]. Das Konzept des Radionetzwerks, das in [240] veröffentlicht wurde, ist in Zusammenarbeit mit Dr. Martin Werner, Prof. Dr. Claudia Linnhoff-Popien und Alexander Erk vom Institut für Rundfunktechnik entstanden. Herr Erk hat hier insbesondere seine Expertise zu Rundfunktechniken und dem Radiostandard DAB beigesteuert, Prof. Dr. Linnhoff-Popien hat bei der Diskussion unterstützt. Dr. Werner hat einige offene Herausforderungen des vorgestellten Konzepts formuliert und Beiträge zum angedachten Data Mining Schema geleistet. Mein Anteil bestand in der Ausarbeitung des Konzepts und den Funktionalitäten bezüglich verteiltem Data Mining, Personalisierung, Schutz der Privatsphäre, Skalierbarkeit und allgemeiner Offenheit und Zugänglichkeit (ca. 70%).

Die Vorarbeiten [237], [243], die sich mit der Netzwerkstruktur und der Ausfallsicherheit von Bitcoin beschäftigen, sind in Zusammenarbeit mit Sebastian Feld und Dr. Werner entstanden. Die Idee, die Verteilung des Bitcoin-Netzes über Autonome Systeme im Internet zu untersuchen, stammt von Sebastian Feld. Dr. Werner hat bei der Diskussion von Methodik und Ergebnissen unterstützt. Mein Beitrag ist hier insbesondere der Evaluation und Diskussion der Ergebnisse zuzurechnen (jeweils ca. 20%).

In Abschnitt 5.2.3 wird die Erhebung von schützenswerten personenbezogenen Daten in verteilten Netzen beschrieben. Diese Ausführungen basieren auf der in [244] publizierten Arbeit, in der ein verteiltes Protokoll beschrieben wird, das den Schutz der Privatheit von Daten durch statistische Überlagerungen garantiert. Mein Beitrag zu dieser Arbeit bezog sich auf Konzept, Umsetzung und Evaluation (ca. 70%). Dr. Werner hat vor allem die Idee zur statistischen Überlagerung eingebracht.

# 2 Soziale Netze mit Kontextbezug

„Kontext“ ist ein Begriff, der seit mehr als 50 Jahren in der Informatik bekannt ist. Erwähnung findet dieser Ausdruck zwar in unterschiedlichsten Fachbereichen, formuliert wird damit aber stets der Gedanke, dass ein Computer oder ein Algorithmus Umwelteinflüssen ausgesetzt ist. Bestimmte Einflüsse kann der Computer adaptieren, wenn er sie über Sensoren erfassen und aus den Messwerten erkennen kann. Seit Computer zunehmend zu mobilen, handlichen, unaufdringlichen und ständigen Begleitern werden, wandelt sich dieses Verständnis von Kontext. Neben den Messungen von Umwelteinflüssen etabliert sich ein sozialer Aspekt von Kontext langsam als Schwerpunkt aktueller Forschung.

Ziel dieses Kapitels ist es, einen Einblick in aktuelle Forschungen zu Kontext im Zusammenhang mit Sozialen Medien zu geben, der für den weiteren Verlauf der Arbeit hilfreich ist. Dazu wird ein Überblick über die Vielfalt der Definitionen und Anwendungen von Kontext und Kontext in Sozialen Netzen gegeben. Es wird gezeigt, wie sich diese Vielfalt sinnvoll systematisieren lässt.

## 2.1 Online Soziale Netze

Ein Online Soziales Netz ist ein über das Internet verfügbarer Dienst, der Individuen Kommunikation innerhalb ihrer persönlichen Netzwerke ermöglicht. Ein Online Soziales Netz bietet zunächst also die Möglichkeit, persönliche Netzwerke an einer Stelle zu versammeln. Motiviert ist die Nutzung dadurch, dass Nutzer beispielsweise am Alltag ihrer Freunde interessiert sind oder sie ihre beruflichen Netzwerke sichtbar machen wollen.

Was mit Online Sozialen Netzen genau bezeichnet wird, wurde in zahlreichen Arbeiten versucht zu definieren. Die Definitionen unterscheiden sich hauptsächlich darin, wie der Begriff mit dem soziologischen Verständnis von Sozialen Netzen als dem persönlichen Beziehungsgeflecht in Einklang zu bringen ist. Eine weit verbreitete und akzeptierte Definition haben Boyd und Ellison in [23] vorgestellt. Sie verwenden den restriktiven Terminus *social network site*, das man in ihrem Sinne wohl als Schauplatz persönlicher Netzwerke übersetzen könnte, also dem Ort im Internet, an dem Individuen ihre persönlichen Beziehungsnetzwerke sichtbar machen. Damit grenzen sie sich hauptsächlich zu anderen Definitionen ab, die Online Soziale Netze als *social networking site* verstanden wissen wollen, also einem Ort im Internet, an dem aktiv Kontakte

geknüpft werden – „Networking“ ist inzwischen auch im Deutschen geläufig und sogar im Duden zu finden. Das impliziere aber eine Aktivität, die meist gar nicht erkennbar sei, da die Menschen diese Dienste vor allem nutzen, um mit ihrem bestehenden Netzwerk zu kommunizieren.

Da der deutsche Begriff der Online Sozialen Netze etwas umständlich ist, werden die Begriffe Online Soziales Netz und Soziales Netz im Folgenden synonym verwendet. Bezeichnet werden damit stets die im Internet verfügbaren *social network sites* im Sinne Boyds und Ellisons.

In dieser Arbeit bezeichnen Soziale Netze also internetbasierte Dienste, die Individuen erlauben, öffentliche oder halb-öffentliche Profile anzulegen, die innerhalb dieses Dienstes verfügbar sind. Des weiteren erlauben Soziale Netze ihren Nutzern, Listen anderer Profile zu pflegen, mit denen sie in Verbindung stehen. Diese Listen können innerhalb des Dienstes von allen Nutzern eingesehen und durchlaufen werden. Was die Verbindung zwischen zwei Profilen genau bedeutet oder wie sie bezeichnet wird, ist von Dienst zu Dienst unterschiedlich – häufig sind es Beziehungen zwischen Freunden, Kontakten oder zu Fans.

Diese Beziehungen können in einer zentralen Datenstruktur abgebildet werden, die allen Sozialen Netzen gemein ist: dem Sozialen Graphen. Der Soziale Graph ist ein Graph mit einem Knoten pro Profil und einer Kante für jede Verbindung zwischen zwei Profilen. Typischerweise bestehen in Sozialen Graphen allerdings sehr viele Verbindungen zwischen Profilen, die nicht miteinander interagieren. Dieses Problem wird als *pollution*, Verschmutzung, des Sozialen Graphen bezeichnet [24].

In Sozialen Netzen, deren Sozialer Graph von solcher Verschmutzung betroffen ist, kann das Geflecht der Beziehungen besser durch die Interaktionen zwischen Profilen beschrieben werden. An der Interaktion können beide Profile aktiv beteiligt sein, etwa indem sie Nachrichten austauschen oder Fotos kommentieren. Es gilt aber ebenso als Interaktion, wenn die Profilinformationen oder Beiträge eines Profils gelesen werden, ein Profil also nur passiv beteiligt ist. Neben dem Sozialen Graphen werden Soziale Netze deshalb häufig über eine zweite Graphstruktur beschrieben und analysiert, die Interaktionsgraph genannt wird. Der Interaktionsgraph ist ein gerichteter Multigraph, in dem die Knoten auch Profile repräsentieren, Kanten aber nur eingefügt werden, wenn zwischen zwei Profile tatsächlich eine Interaktion stattgefunden hat.

Die Konzepte von Sozialen und Interaktionsgraphen werden im weiteren Verlauf der Arbeit nochmals aufgegriffen. An dieser Stelle soll aber der Zusammenhang zwischen Sozialen Netzen und Kontext erörtert werden. Daher widmet sich der nächste Abschnitt zunächst dem Kontextbegriff.

## 2.2 Zum Verständnis von Kontext

Im natürlichen Sprachgebrauch bezeichnet „Kontext“ eine Verbindung oder einen (Sinn-)Zusammenhang zu einer Umgebung. Analog dazu wird der Begriff auch in der Informatik als Verbindung zwischen Umwelt und Fallunterschei-

dungen in Computer-Prozessen aufgefasst. 1994 wurde der Begriff *context-aware computing*, zu deutsch etwa Kontextsensitivität und -adaptivität, geprägt, mit dem die Fähigkeit eines Computers oder eines Prozesses verstanden wird, sich an den Ort seiner Benutzung und die Objekte in seiner unmittelbaren Umgebung anzupassen [25]. Da Kontext meist als Ausgabewert von Sensoren vorlag, beschreibt die gängigste Definition von Dey aus dieser Zeit Kontext vor allem als Information:

*„Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.“ [26]*

Demnach ist Kontext jede Art von Information, die die Situation einer Entität beschreiben kann. Unter Entität wird eine Person, ein Ort oder ein Objekt verstanden, das für die Interaktion zwischen Nutzer und Applikation von Bedeutung ist. Der Nutzer und die Applikation können selbst auch Entitäten sein.

Dey bringt mit dieser Definition zahlreiche Vorarbeiten zu Kontext und Kontextsensitivität auf einen Nenner. Bis zu dieser Formulierung gab es nur diffuse Beschreibungen der Begrifflichkeiten, die Kontext nur selten von „Situation“ oder „Umwelt“ trennen konnten [27]–[29]. Zudem trug Dey der Sorge Rechnung, dass mit der Modellierung der Umgebung des Nutzers der Nutzer selbst aus dem Fokus geraten könnte [30].

Neuere Arbeiten heben hervor, dass die oben genannte Definition äußerst unspezifisch ist. Zudem wird das Verständnis von Kontext als eine Momentaufnahme kritisiert. Die zugrundeliegende Annahme, dass Kontext durch Sensormessungen – also einer Menge numerischer Werte – definiert wird, die nur während einer Interaktion zwischen Nutzer und Anwendung kurzzeitig gültig sind, sei nicht mehr zeitgemäß. Kontext sollte vielmehr als gelerntes Wissen aufgefasst werden, das aus einem kontinuierlichen Strom von Informationen deduziert wird [31]. In diesem Zusammenhang sollten Kontext und Kontextinformation auch begrifflich voneinander getrennt werden – der Kontext bezeichnet demnach das gelernte oder abstrahierte Wissen, das auf Kontextinformationen aufbaut. Außerdem kann mit Kontext nicht nur die Situation einer Entität beschrieben werden – Kontext kann auch aus der Aktivität der Entität hervorgehen oder selbstständig von der Anwendung erzeugt werden [32]. Sogar eine Interaktion zwischen Nutzer und Anwendung ist nicht zwingend erforderlich: Ein kontextadaptives System kann sich den Umwelteinflüssen auch anpassen, ohne dass ein Nutzer eingreifen oder diese Anpassung bemerken müsste [33]. Das ist insbesondere bei mobilen Anwendungen der Fall, die zum Beispiel aus fehlender Interaktion eigene Schlüsse über den Kontext des Benutzers ziehen können.

Aus diesen Gedanken zu Deys Definition wird bereits deutlich, dass Kontext kein scharf umrissener Begriff ist, der eine definierte Informationsmenge oder einen einheitlichen Vorgang beschreibt. Vielmehr entwickelt sich eine mögliche Definition stetig weiter, wird an mancher Stelle präzisiert und an anderer Stelle bewusst erweitert. Was unter relevantem Kontext verstanden wird, ist immer abhängig von der jeweiligen Anwendung. Zudem wird der jeweils relevante Kontext von jeder Anwendung individuell erkannt, repräsentiert und verwaltet – in der Literatur finden sich deshalb unzählige Taxonomien zur Systematisierung von Kontextinformationen und den dazugehörigen Erkennungsalgorithmen [34]–[36]. An dieser Stelle sollen aber weniger die unterschiedlichen Varianten von Kontexterkennung, Ontologien oder Vermittlungsschichten (*middlewares*) im Vordergrund stehen. Für diese Arbeit ist vielmehr von Interesse, warum die Themen Kontext und Soziale Netze sich überhaupt überschneiden.

Die Nähe der Themen liegt in der Durchdringung des Alltags mit computergestützter Intelligenz begründet – in der Fachliteratur als *pervasive computing* bezeichnet. Seit Computer zu unaufdringlichen und ständigen Begleitern werden, erfassen deren Sensoren zunehmend auch unseren Alltag. Die so gewonnenen Kontextinformationen enthalten damit zunehmend facettenreicheres Wissen zum Beispiel über die zwischenmenschlichen Interaktionen von Nutzern. Dieses Wissen kann als Teilmenge der allgemeinen „Situationsbeschreibung“ im Sinne Deys angesehen werden. In der Literatur etablierte sich dafür zunächst der Begriff des *social context*. Damit wurden diejenigen Kontextinformationen bezeichnet, die die wichtigsten Orte, Regelmäßigkeiten im Alltag und die sozialen Bindungen eines Nutzers beschreiben [37]. Verfeinert wurde dieser Begriff zum *situated social context*, mit dem eine zeitlich und räumlich unmittelbarere Gültigkeit des *social context* ausgedrückt werden sollte:

> *Situated Social Context of an individual is the set of people that share some common spatio-temporal relationship with the individual, which turn them into potential peers for information sharing or interaction in a specific situation. [38]*

Als *situated social context* wird hier also die Menge von anderen Personen bezeichnet, die zeitlich und räumlich mit einem Nutzer in Beziehung stehen. Das macht diese Personen zu potenziellen Empfängern von Informationen oder Kollaborateuren in bestimmten Situationen. Hervorzuheben ist an dieser Definition vor allem die Begrenzung auf einen Personenkreis. Damit wird Kontext erstmals unterteilt in die Messwerte aus Sensoren der mobilen Endgeräte und das soziale Umfeld des Nutzers.

Die eigentliche Verschmelzung von Kontextinformationen und Sozialen Netzen wird schließlich in der folgenden Definition zum *pervasive social context* ausgedrückt:

*Pervasive social context of an individual is the set of information that arises out of direct or indirect interaction with people carrying sensor-equipped pervasive devices connected to the same social network service. [35]*

Mit dem *pervasive social context* des Nutzers eines mobilen Endgeräts werden demnach alle die Informationen bezeichnet, die aus direkter oder indirekter Interaktion mit anderen Nutzern solcher Endgeräte, die (darüber) mit demselben Sozialen Netz verbunden sind, entstehen. Kontexterkennungsalgorithmen werten demnach weiterhin Sensormessungen der Endgeräte aus. Soziale Netze liefern dann aber notwendige externe Informationen, um aus diesen Messungen auf das soziale Umfeld eines Nutzers schließen zu können. Der Fokus der Kontexterkennung erweitert sich damit von der Beschreibung der Situation einer Person hin zu deren Einbettung in eine soziale Gruppe.

Kontextinformationen können dazu verwendet werden, die Qualität solcher Einbettungen in soziale Gruppen zu bewerten. Diese Qualität wird in der Soziologie als *closeness* bezeichnet. In [37] dienten zum Beispiel Ortsinformationen dazu, die *closeness* von Personen zu bewerten. Die Autoren trafen dazu die Annahme, dass räumliches Aufeinandertreffen und die Zeitpunkte und Häufigkeiten solcher Begegnungen Rückschlüsse auf die Art und Intensität der Beziehung zulassen: Arbeitskollegen trifft man tendenziell tagsüber, Familienmitglieder und Freunde eher abends und am Wochenende, engere Freunde häufiger als lose Bekanntschaften. Diese Annahmen wurde später eingehend analysiert und weitgehend bestätigt – unter anderem von Scellato et al. [39].

In dem in [37] vorgeschlagenen Experiment haben die Autoren eine Versuchsgruppe mit mobilen Endgeräten ausgestattet, die die geographische Position des Endgeräts über einen GPS-Empfänger ermitteln konnten. Zudem waren die Endgeräte mit Bluetooth-Modulen ausgestattet, über die sie in unmittelbarer räumlicher Nähe direkt miteinander kommunizieren konnten. Indem diese beiden Sensoren ein sehr genaues Abbild der räumlichen Nähe der Versuchspersonen lieferten, konnten die Autoren unter anderem die soziale Struktur der Versuchgsgruppe nachbilden und eine Schätzung der *closeness* der Personen liefern: Die Autoren konnten Familienmitglieder, Freunde und Arbeitskollegen in der Versuchsgruppe allein auf Basis der Informationen zu räumlicher Nähe unterscheiden.

In [40] wurden ortsbezogene Kontextinformationen verwendet, um die Struktur einer vergleichsweise großen sozialen Gruppe abschätzen zu können. So haben die Autoren Bewegungsprofile von knapp 500 Versuchsteilnehmern analysiert, um daraus den Sozialen Graphen nachzubilden. Die Versuchsteilnehmer wurden über Facebook rekrutiert und deren Bewegungsprofile über eine eigens entwickelte Erweiterung gesammelt. Aus den Bewegungsprofilen wurden dann zunächst die Orte extrahiert, die einzelne Profile regelmäßig besucht haben. Aus den besuchten Orten aller Profile wurde dann ein *co-location*-Netzwerk gebildet – eine ungerichtete Graphstruktur, in der Profile über einen Ort verknüpft werden, den sie beide innerhalb einer gleichen Zeitspanne besucht ha-

ben. Obschon dieses *co-location*-Netzwerk ungefähr dreimal so viele Kanten enthielt wie der ursprüngliche Soziale Graph, konnten doch Erkenntnisse über den Zusammenhang zwischen räumlicher Nähe und Verknüpfungen in dem zugrundeliegenden Sozialen Netz gewonnen werden.

Kontextinformationen dienen aber nicht nur dazu, die Qualität der Verknüpfungen in einem Sozialen Graphen einzuschätzen. Der folgende Abschnitt zeigt, wie Kontextinformationen helfen können, Verknüpfungen in einem sozialen Graphen zu etablieren oder Kollaboration in bestehenden Sozialen Graphen zu ermöglichen.

## 2.3 Kontext in Sozialen Netzen

Mit der zunehmenden Verbreitung mobiler Endgeräte werden auch Soziale Netze zunehmend mobil genutzt. Und weil mobile Endgeräte es über entsprechende Schnittstellen der mobilen Betriebssysteme einfach ermöglichen, zusätzlich zu Nachrichten und Bildern auch Kontextinformationen aufzuzeichnen, werden Wege vorgeschlagen, wie diese Kontextinformationen in Soziale Netze integriert werden können. Inzwischen bieten existierende Soziale Netze natürlich Anwendungen für Betriebssyteme mobiler Endgeräte und Schnittstellen, mit denen mobil erzeugter Inhalt mit Kontextinformationen annotiert werden kann. Das ist aber nur eine Art, Kontextinformationen zu integrieren. In der Literatur existieren noch zahlreiche weitere Varianten.

Eine Variante, Kontextinformationen und Soziale Netze zu verbinden, wurde in [41] vorgestellt. Diese Systematik eröffnet mehrere Perspektiven auf mögliche Verbindungen und soll daher an dieser Stelle aufgegriffen werden. Zunächst wird grob zwischen zwei konträren Vorgehensweisen unterschieden: Zum einen kann ein Soziales Netz als reine Informationsquelle für eine Infrastruktur dienen, die erst auf der Basis dieser Informationen allgemein verbreitete, den Alltag durchdringende (*pervasive*) Netzwerke oder Dienste ermöglichen kann. Das andere Extrem betrachtet ein Soziales Netz als eine Vermittlerschicht (*middleware*) zwischen Sensoren und allgemein verbreiteten Diensten. Damit können Dienste vorgeschlagen werden, die dem Sozialen Netz Kontextinformationen bereitstellen oder dieses selbst als Infrastruktur für neuartige Dienste ansehen. Grafik 2.1 visualisiert diese beiden Extrema.

Zwischen beiden Extrema gibt es wiederum zwei feinere Unterteilungen, die sich entweder mehr auf das Netz als Informationsquelle für spezielle Dienste oder die Sensoren zur Etablierung neuer Netze stützen, wie Grafik 2.1 zeigt. Die entsprechenden Verfahren beziehen aber stets sowohl das Soziale Netz als Infrastruktur als auch die gängigsten Sensoren als Informationsquelle mit ein: Die erwähnten Dienste basieren meist auf neuartigen Routingstrategien, die Inhalte kontextabhängig verteilen und dabei die Struktur Sozialer Netze in Routingentscheidungen miteinbeziehen; neue Soziale Netze werden demgegenüber in räumlicher Nähe, anwendungsspezifisch und ohne weitere Infrastruktur (*ad hoc*) etabliert.

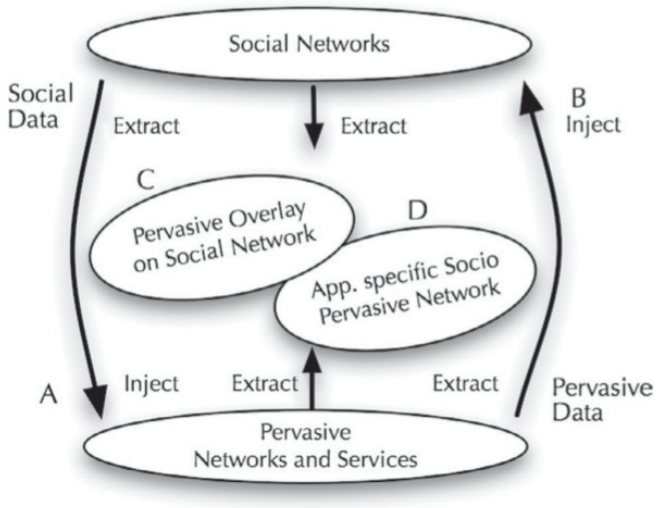



Abbildung 2.1: Integration von Kontextinformationen in Soziale Netze [41]

Im folgenden werden Anwendungen vorgestellt, die jeweils einem dieser vier Aspekte genügen – die also Soziale Netze als Quelle für bestimmte Kontextinformationen ansehen (Abschnitt 2.3.1), die Sozialen Netzen Kontextinformationen bereitstellen (Abschnitt 2.3.2), die kontextspezifische Overlay-Netze für Soziale Netze aufbauen (Abschnitt 2.3.3) oder die eigene Soziale Netze kontextbasiert etablieren (Abschnitt 2.3.4).

### 2.3.1 Soziale Netze als Quelle für Kontextinformationen

Die folgenden Arbeiten verwenden Soziale Netze als Quelle für Informationen über den sozialen Kontext eines Nutzers. Diese Informationen werden dann genauso integriert wie Sensormesswerte, die eher den situativen Kontext des Nutzers und der Nutzung beschreiben.

Eine solche Arbeit wird in [42] vorgestellt. Darin wird die räumliche Nähe zu Freunden aus einem Sozialen Netz festgestellt. Für diese Freunde können dann gezielt Nachrichten an gemeinsamen Orten hinterlassen werden. Die Autoren haben zu Demonstrationszwecken eine digitale ortsbasierte Schnitzeljagd realisieren können.

In [43] wird ein Empfehlungssystem vorgestellt, das Soziale Netze als Informationsquelle für Interessen von Personen nutzt. Die Endgeräte der Personen, die sich in räumlicher Nähe zueinander aufhalten, tauschen dazu über drahtlose Kommunikationskanäle ihre Identifikatoren in einem Sozialen Netz aus. Von dort werden Profilinformationen abgerufen und über alle anwesenden Personen aggregiert. Daraus können Empfehlungsalgorithmen Inhalte vorschlagen, die dem Geschmack der Gruppe entsprechen.

Ein ähnliches Verfahren wird in [44] vorgeschlagen. Damit werden kontextsensitive Anwendungen für einen Kreis von Personen ermöglicht, die sich in

räumlicher Nähe zueinander aufhalten und in einem Sozialen Netz miteinander verbunden sind. Als Beispiel wird das unmittelbare Verteilen von Fotos auf einer Party an Freunde genannt.

Andere Verfahren, die in [45] und [46] vorgestellt werden, unterstützen die Annotation von Multimedia-Inhalten. Kontextinformationen mit räumlichem und zeitlichem Bezug werden im Gegensatz zu den vorherigen Verfahren erst zu einem späteren Zeitpunkt ausgewertet. Soziale Netze dienen aber auch hier als Informationsquelle – in diesen Fällen für Empfehlungsalgorithmen: Diese präsentieren eine Auswahl von Schlagworten, die befreundete Nutzer zu anderen Fotos aus zeitlich und räumlich ähnlichem Kontext abgegeben haben.

In [47] wird *SocialTelescope* vorgestellt. Dabei handelt es sich um ein Empfehlungssystem für beliebte Orte wie Restaurants oder Parks. Die Datenbasis der Empfehlungen wird vollkommen automatisch aus dem Sozialen Netz Twitter [228] extrahiert, indem Nachrichten ausgewertet werden, die mit einem räumlichen Bezug annotiert sind. Die Evaluationen des Verfahrens zeigen, dass diese Form der Informationssammlung eine Datenbasis erzeugen kann, die qualitativ vergleichbar ist zu nutzergenerierten Datenbanken wie der von Yelp [229] – einem großen Bewertungsportal für Restaurants und Bars.

Ein umgekehrter Ansatz wird in [48] verfolgt. Darin wird ein Framework für Empfehlungssysteme vorgestellt, das Anfragen zu Bewertungen an bestimmte Nutzer verteilt, die aufgrund ihrer Historie und aktuellen Kontextinformationen eine Bewertung abgeben können. So darf eine Bar etwa nur von einer Person bewertet werden, die die Bar in näherer Vergangenheit selbst besucht hat.

In [49] wird eine Architektur vorgestellt, die Daten aus Sozialen Netzen kontextbasiert annotiert und entsprechend aggregieren kann. Dazu werden Nachrichten an Soziale Netze mit Informationen zu ihrem Entstehungsort versehen. So können gezielt Nachrichten abonniert werden, die an einem bestimmten Ort von bekannten Profilen verfasst worden sind.

### 2.3.2 Existierende Soziale Netze als Vermittlungsschicht

Arbeiten aus diesem Bereich sehen existierende Soziale Netze als eine Art Vermittlungsschicht zu pervasiven Sensoren an. Mit Hilfe dieser Vermittlungsschicht können Sensormessungen bestimmter Personen(-kreise) gezielt gesammelt und aggregiert und zur weiteren Verwendung ausgewertet werden – die Anwendungen aus diesem Bereich sind häufig sogenannte *crowd-sensing*-Anwendungen.

In [50] wird eine klassische *crowd-sensing*-Anwendung vorgestellt, die einen aktuellen Wetterbericht zusammenstellt, der anhand von entsprechenden Sensormessungen von Nutzern des Sozialen Netzes Twitter [228] zusammengetragen wird. Twitter wird also als Vermittlungsschicht genutzt, um die Sensormessungen der Nutzer an den Wetterdienst und um umgekehrt den Wetterbericht an die Nutzer zu vermitteln. In dieser Arbeit werden zudem nicht nur Wet-

terdaten gesammelt, sondern auch Daten zu ortsbasierter Lärmbelastung. Ein ähnliches Verfahren wird in [51] vorgestellt, das nicht die Verteilung von nutzergemessenen Daten ermöglicht, sondern echte Sensornetze an Soziale Netze anbindet.

Das in [52] vorgestellte Verfahren verwendet Twitter als eine Vermittlungsschicht zwischen Computerprozessen und Menschen. Das Soziale Netz dient in diesem Fall als Steuerungs- und Überwachungseinheit. Über Twitter können Ressourcen reserviert und beansprucht werden. Die verteilt ablaufenden Prozesse kommunizieren zudem untereinander über den Kurznachrichtendienst. Zur Überwachung der verteilten Berechnungen können einfach deren Steuerungsnachrichten abonniert werden.

Ein weiteres Beispiel für eine Vermittlung von Sensordaten wird mit *CenceMe* vorgestellt [53], [54]. Das System erfasst Kontextinformationen und leitet daraus aktuelle Aktivitäten der Nutzer ab. Diese Aktivitäten können archiviert, eingesehen und über Soziale Netze kontrolliert geteilt werden.

### 2.3.3 Overlay-Netze für Soziale Netze

Overlay-Netze sind im Allgemeinen eine anwendungsspezifische Abstraktionsschicht, mit der die darunterliegende Vernetzung einer hardwarenäheren Schicht oder Infrastruktur verschattet wird. Das ermöglicht Anwendungen, simplere oder semantisch bedeutsame Routingentscheidungen zu treffen. Das Routing, das auf dem Overlay-Netz aufbaut, wird von der Abstraktionsschicht dann auf die darunterliegende Schicht oder Infrastruktur abgebildet.

Arbeiten, die solche Overlay-Netze für Soziale Netze etablieren, sind keinem der beiden Extrema aus Grafik 2.1 ausschließlich zuzuordnen, weil diese Arbeiten sowohl das Soziale Netz als Quelle für Kontextinformationen verwenden, als auch als eine Art Vermittlerschicht zu den Knoten im Sozialen Netz fungieren. Daher ist die Gruppe dieser Anwendungen in der genannten Grafik auch zwischen beiden Extrema angesiedelt.

Das erste Beispiel für diese Kategorie von Arbeiten ist *SCAMPI* – eine Service-Plattform, die die Ressourcen von mobilen Endgeräten in opportunistischen Netzwerken zur Verfügung stellt. Das sind Netzwerke, die ohne Infrastruktur (*ad hoc*) entstehen, meist nur für kurze Zeiträume existieren und lokal begrenzt sind. Die Architektur sieht vor, dass ein Endgerät keine Ressourcen für fremde Nutzer freigibt, sondern nur für diejenigen Nutzer, zu denen eine Verbindung in einem Sozialen Netz besteht. So werden in einem *ad hoc*-Netzwerk spontan zusätzlicher Speicher, Rechenleistung oder Internetverbindungen anderer Endgeräte zur Verfügung gestellt, um darauf beliebige Anwendungen aufsetzen zu können [55].

In [56] wird ein ähnliches Ziel verfolgt – nämlich die Ressourcen mobiler Endgeräte einer sozialen Gruppe zur Verfügung zu stellen. Anders als SCAMPI wird hier nicht von einem opportunistischen Service-Netzwerk ausgegangen. Mit diesem Verfahren soll hauptsächlich die Datenmenge reduziert werden,

die über das Internet übertragen werden muss, indem Inhalte über direkte und lokale Kommunikation zwischen Endgeräten verteilt werden. Der Kern der Arbeit besteht darin, Endgeräte als Verteilerknoten zu bestimmen, denen alle Geräte in räumlicher Nähe „vertrauen“. Das Vertrauen wird dabei aus den Beziehungen der Besitzer der Endgeräte abgeleitet, die in einem Sozialen Graphen einsehbar sind, und ist nicht auf direkte Nachbarn im Sozialen Graph beschränkt. Aus der Kombination von Geräten in räumlicher Nähe und den sozialen Beziehungen ihrer Besitzer wird dann eine Overlay-Struktur konstruiert, mit der die Endgeräte nur noch über den gewählten Verteilerkonten mit dem Internet verbunden sind.

Ein ähnliches Verfahren wird in [57] vorgestellt. Auch hier soll der Datenverkehr über das Internet reduziert werden. Anders als in [56] erfolgt die Auswahl der Verteilerknoten nicht aufgrund transitiven Vertrauens, das über mehrere Hops hinweg entstehen kann, sondern nur über direkte Verknüpfungen in einem Sozialen Graph.

Mit [58] wird ein Ansatz vorgestellt, der die räumliche und soziale Nähe von Personen für Routingentscheidungen in opportunistischen Netzwerken ausnutzt. Das vorgestellte Verfahren analysiert daher zunächst den sozialen und räumlichen Kontext der Nutzer, um vorhersagen zu können, wo sich welche Nutzer wann sehr wahrscheinlich treffen werden. Nur an diesen sporadischen Treffpunkten können überhaupt Daten ausgetauscht werden. Das Wissen über die zukünftigen Bewegungen und sozialen Kontakte der Nutzer wird genutzt, um Inhalte so intelligent weiterzuleiten, dass möglichst alle Nutzer alle Inhalte möglichst schnell empfangen.

In [59] wird ein ähnliches Verfahren vorgestellt, das zusätzlich auch das Interesse an bestimmten Inhalten mit einbezieht. Nutzer sollen hier also spezifischer und gezielter mit Inhalten versorgt werden.

### 2.3.4 Anwendungsspezifische Soziale Netze

Eine weitere Kategorie von Arbeiten, die sich mit dem Zusammenspiel von Kontextinformationen und Sozialen Netzen beschäftigen, etabliert eigene anwendungsspezifische Soziale Netze auf der Basis von Kontextinformationen. Daher sind die Arbeiten dieser Kategorie ebenfalls keinem der äußeren Bereiche in Grafik 2.1 zuzuordnen.

Eine frühe Arbeit aus dieser Kategorie heißt *SMILE*. In diesem System kommt eine Verknüpfung in dem Sozialen Netz der Anwendung zustande, wenn Endgeräte eine räumliche Nähe über ein drahtloses Nahbereichskommunikationsverfahren (wie zum Beispiel Bluetooth) festgestellt haben. In räumlicher Nähe werden Schlüsselpaare ausgetauscht, mit denen das räumliche Aufeinandertreffen zu einem späteren Zeitpunkt bewiesen werden kann. Der Austausch der Schlüsselpaare kommt ohne Interaktion der Nutzer aus – die Nutzer müssen sich also nicht kennen oder das Aufeinandertreffen bewusst wahrnehmen und können auch erst zu einem späteren Zeitpunkt merken, dass sie sich in

der Vergangenheit begegnet sind. Die SMILE Architektur wurde in weiteren Arbeiten einer Sicherheitsanalyse unterzogen und zum Beispiel gegen Identitätsdiebstahl gesichert [60], [61].

In [62] und [63] werden Systeme vorgestellt, die Menschen motivieren sollen, Kontakte in unmittelbarer räumlicher Nähe zu bisher unbekannten Personen zu knüpfen. Dazu wird wie in SMILE per Bluetooth nach anderen Geräten in räumlicher Nähe gesucht. Werden Geräte in der Nähe entdeckt, werden die Nutzer benachrichtigt und ein Chat etabliert. In [62] vergleichen die Geräte zudem Profilinformationen miteinander, um den Nutzern mögliche Gesprächsthemen für ein Kennenlerngespräch vorzuschlagen.

Mit *VEGAS* wird ein eigenes verteiltes Soziales Netz vorgeschlagen, in dem Verknüpfungen nur in räumlicher Nähe und durch aktive Interaktion der Nutzer zustande kommen. Der Fokus dieser Architektur liegt auf der Sicherheit der Kommunikation und der informationellen Selbstbestimmung der Nutzer. Beim räumlichen Aufeinandertreffen werden Schlüsselpaare ausgetauscht, die für die Kommunikation notwendig sind. Ein Weg, die Schlüssel auszutauschen, ist ein optischer Barcode, den die Nutzer gegenseitig abfotografieren müssen [24], [64].

Neben diesen ausgewählten Beispielen sind in dieser Kategorie zahlreiche weitere Arbeiten anzusiedeln, die in der Literatur als *Mobile Social Networking Applications* firmieren [65]. Der Begriff umfasst zwar auch die Smartphone-Applikationen existierender Sozialer Netze. Neuere Forschungsarbeiten zu diesem Thema fusionieren aber zunehmend Mobilität und Sensorik der Endgeräte, wie es in den genannten Beispiele bereits erkennbar ist und wie es mehrere Übersichtsarbeiten prognostiziert haben [66]–[68]. Für Details zu Architekturen und Varianten sei hier auf umfangreiche Übersichtsarbeiten verwiesen [69], [70]; auch die Sicherheitsrisiken solcher Netze und daraus erwachsende Gefahren für die Privatsphäre der Nutzer wurden bereits in zahlreichen Arbeiten detailliert untersucht [71]–[73].

Aktuelle Arbeiten zu Mobilen Sozialen Netzen können in drei Klassen unterschieden werden, die als *where*, *who* und *what* also Wo, Wer und Was bezeichnet werden [69]. In der Klasse „Wo“ werden Applikationen aufgeführt, die auf den Ort ihrer Ausführung Bezug nehmen – in dieser Kategorie sind also die klassischen ortsbasierten (*location-based-*) Anwendungen zu finden. In der Klasse „Wer“ wird vor allem die Nähe zu oder zwischen Objekten berücksichtigt. Anwendungen dieser Kategorie bauen zum Beispiel Kommunikationskanäle zu Nutzern in räumlicher Nähe auf. Die letze Kategorie „Was“ umfasst alle Anwendungen, die ein bestimmtes Thema fokussieren – also beispielsweise das Teilen von Multimedia-Inhalten.

### 2.3.5 Zur Einordnung

Die vorherigen Abschnitte haben eine Übersicht über die Integration von Kontextinformationen und Sozialen Netzen gegeben. Es wurde ein Schema vor-

gestellt, mit dem sich die zahlreichen Vorarbeiten und Anwendungen sinnvoll unterscheiden lassen. Danach können Soziale Netze entweder selbst Quelle für Kontextinformationen sein oder ein Netzwerk von Konsumenten derartiger Informationen repräsentieren. In Abschnitt 2.3.1 wurden Arbeiten der ersten Kategorie aufgeführt; Abschnitt 2.3.2 hat Arbeiten vorgestellt, die Kontextinformationen an Interessenten in Sozialen Netzen verteilen. Die darauf folgenden Abschnitte 2.3.3 und 2.3.4 haben Arbeiten vorgestellt, die keiner der ersten beiden Kategorien eindeutig hätten zugeordnet werden können – entweder, weil die Arbeiten Abstraktionsschichten für eine kontextbasierte Vernetzung der Knoten Sozialer Netze definieren, oder weil eigene anwendungsspezifische Soziale Netze vorgestellt werden, die auf der Basis von Kontextinformationen etabliert werden.

Durch die Vielzahl der aufgeführten Arbeiten, Schemata oder Klassifikationen sollte das vielfältige Verständnis von Kontext deutlich werden. Den Anwendungen liegen stets unterschiedliche Auffassungen oder Repräsentationen von Kontext zugrunde. Zudem sind die Modalitäten der Kontextinformationen von Anwendung zu Anwendung höchst unterschiedlich.

In keiner der vorgestellten Arbeiten werden Kontextinformationen dazu verwendet, soziale Kontakte zu gruppieren oder zu organisieren. Kontextinformationen wurden nur dazu eingesetzt, neue Verbindungen zu etablieren – etwa indem Kontextähnlichkeit auf der Basis einzelner Stichwörter festgestellt wird, wie es in Abbildung 2.2 schematisch dargestellt ist, oder indem Personen in räumlicher Nähe einander bekannt gemacht werden. „Kontextabhängige“ Kommunikation in räumlicher Nähe wurde vor allem durch die eingesetzte Funktechnologie realisiert, deren Reichweite entsprechend begrenzt ist.

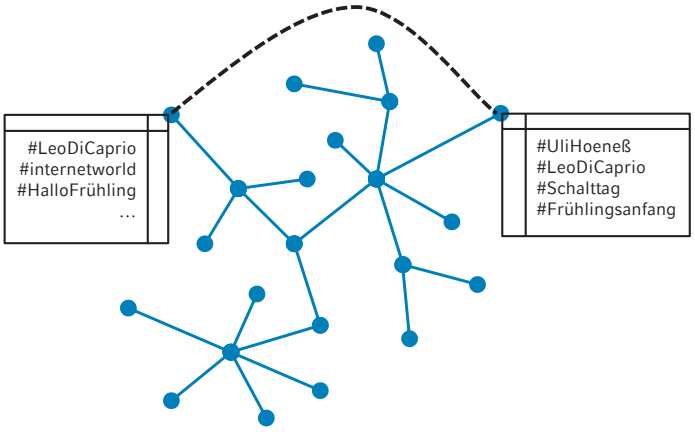



Abbildung 2.2: Gemeinsame Stichwörter in Kontextbeschreibungen führen zu neuen Verknüpfungen im Sozialen Graphen

In Bezug auf die eingangs vorgestellten besonderen Dynamiken der computervermittelten Kommunikation in Sozialen Netzen haben diese Arbeiten entweder keine Auswirkungen oder sie bieten keine großflächigen Einsatzmöglichkeiten. Dass Kommunikationskanäle beispielsweise nur zu Personen in unmittelbarer Umgebung aufgebaut werden, verhindert zwar, mit Mitgliedern unterschiedlicher Sozialen Gruppen gleichzeitig zu kommunizieren. Die technischen Beschränkungen verhindern aber ebenso, diese Anwendung als allgemein einsetzbares, pervasives Soziales Netz in Betracht zu ziehen.

Bisher ist auch noch kein Konzept bekannt, das Endgeräte kontextbasiert und trotzdem unabhängig von einzelnen Modalitäten vernetzt und diese Vernetzung dann für allgemeine kontextbasierte Dienste zugänglich macht. Vielmehr sind die vorgestellten Anwendungen stark von den jeweiligen Kontextmodalitäten abhängig und es würde eine erhebliche Modifikation der zugrundeliegenden Architektur bedeuten, zusätzliche Modalitäten einzubauen. Und kaum eine Arbeit ist als abstrakte Basis für kontextbasierte Anwendungen und Dienste geeignet. Die vorgestellten Arbeiten stellen also vor allem Insellösungen dar, die noch keine allgemeine Anwendbarkeit zulassen.

## 2.4 Zusammenfassung

Dieses Kapitel hat den Zusammenhang zwischen Kontext und Sozialen Netzen dargestellt.

Dazu wurde zunächst beschrieben, was mit Sozialen Netzen in dieser Arbeit gemeint ist. In Abschnitt 2.2 folgten einige Definitionen von Kontext und Kontextinformationen. Es wurde deutlich, dass es keinen Konsens gibt, wie Kontext anwendungsübergreifend repräsentiert werden kann. Der darauf folgende Abschnitt 2.3 hat das verdeutlicht. Darin wurden Vorarbeiten vorgestellt, die Kontextinformationen und Soziale Netze auf unterschiedlichste Arten verbinden. Jeder der Vorarbeiten liegt eine eigene Auffassung von Kontext oder den notwendigen Modalitäten zugrunde. Diese anwendungsspezifische Festlegung erschwert es erheblich, zusätzliche Modalitäten einzubinden oder allgemeine kontextbasierte Dienste darauf aufzusetzen.

Einige Arbeiten wurden vorgestellt, die ein eigenes anwendungsspezifisches Soziales Netz auf der Basis von Kontextinformationen etablieren. Diese Sozialen Netze wurden entworfen, um Sicherheitsbedenken oder Gefahren für die Privatsphäre der Nutzer zu begegnen, oder um kontextabhängige Kommunikation in unmittelbarer räumlicher Umgebung zu ermöglichen. Damit hatte keine der Anwendungen erkennbare Einflüsse auf die allgemeine Kommunikationssituation mit einem diversifizierten persönlichen Netzwerk.

Das folgende Kapitel stellt nun eine Datenstruktur vor, mit der Kontexte modalitätsunabhängig beschrieben werden können: Bloom-Filter. Der klassische Einsatz dieser Filter liegt im Bereich von großen Datenbanken, in denen Bloom-Filter eine zuverlässige Abschätzung von Zugehörigkeitsanfragen liefern können. Suchanfragen können damit deutlich optimiert werden. In dieser

Arbeit dienen Bloom-Filter ebenfalls zur Beschreibung von Mengen. Anstelle für Zugehörigkeitsanfragen werden sie hier zum generischen Vergleich von Mengen eingesetzt. Insbesondere für den Vergleich von Kontexten bietet das weitreichende Möglichkeiten und Vorteile.

# 3 Bloom-Filter zur Beschreibung von Kontext

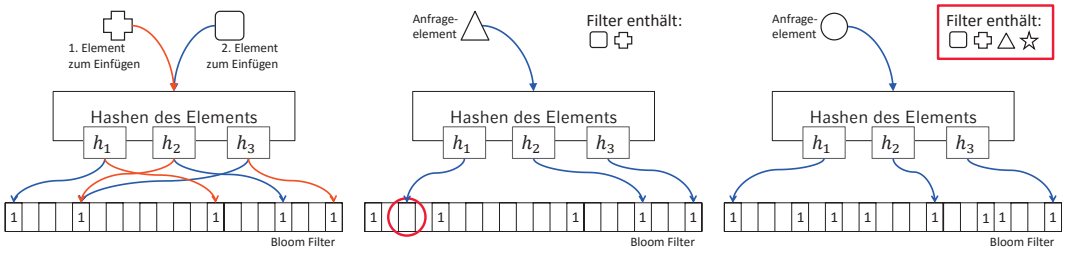
Bloom-Filter nehmen im Rahmen dieser Arbeit eine zentrale Rolle ein. Ursprünglich zur Optimierung von Suchanfragen an große Datenbanken entwickelt, werden sie hier vor allem zum generischen Vergleich von Mengen eingesetzt. Da Mengen in dieser Arbeit vor allem aus heterogenen Kontextinformationen bestehen, ermöglicht die Vergleichsoperation auf Bloom-Filtern erstmals den typlosen und modalitätenunabhängigen Vergleich von Kontexten. Daher bilden Bloom-Filter den Kern des kontextzentrischen Adressierungsschemas.

Bevor das Konzept kontextzentrischer Sozialer Netze im Detail beschrieben wird, widmet sich dieses Kapitel zunächst dem klassischen Einsatz von Bloom-Filtern als Indexstruktur in Datenbanksystemen. Es werden einige Erweiterungen der ursprünglichen Definition vorgestellt, die zeigen, wie vielfältig Bloom-Filter einsetzbar sind. Zudem wird beschrieben, wie bekannte Maße zum Vergleich zweier Mengen mit Hilfe von Bloom-Filtern abgeschätzt werden können. Mit dem Schluss dieses Kapitels wird schließlich erörtert, warum Kontexte mit Hilfe von Bloom-Filtern beschrieben werden können.

**Veröffentlicht** wurden Teile dieses Kapitels in [242], [245] und [241]. Die Abschätzung der Ähnlichkeit zweier durch Bloom-Filter beschriebener Mengen über die Jaccard-Distanz wurde in [241] beschrieben. Die beiden Vorarbeiten [242] und [245] stellen jeweils Varianten von Bloom-Filtern vor, die die Wahrscheinlichkeit falschpositiver Filterantworten bei der Stichwortsuche verringern.

## 3.1 Bloom Filter und ihr klassischer Einsatz

Ein Bloom-Filter ist eine probabilistische Datenstruktur zur speichereffizienten Beschreibung von Mengen. Vorgeschlagen wurde diese Datenstruktur von Burton H. Bloom im Jahr 1970 in [74] und seitdem beständig weiterentwickelt: Mittlerweile existieren zahllose Varianten mit denen Bloom-Filter für spezielle Anwendungsfälle optimiert werden. Im Kern geht es aber stets darum, über eine Anfrageoperation an einen Bloom-Filter zu entscheiden, ob ein Element in der zugrundeliegenden Menge enthalten ist oder nicht. Die Stärke von Bloom-Filtern liegt darin, dass sie diese Anfrageoperationen niemals falschnegativ beantworten und die Wahrscheinlichkeit falschpositiver Antworten konfigurierbar ist.




(a) Einfügen (b) Anfragen (c) Falschpositive Antwort

Abbildung 3.1: Bloom-Filter: Protokoll und Falschpositive Filterantworten

Ein Bloom-Filter ist zunächst ein Array $F$ von $m$ Bits, die initial alle den Wert Null annehmen. Dazu gehört eine Menge von $k$ Hash-Funktionen. Zur Beschreibung einer Menge von Elementen werden auf jedes Element alle $k$ Hash-Funktionen angewendet, die die Elemente auf einen Hash-Wert kleiner $m$ abbilden. Diese Hash-Werte werden als Adressen im Bloom-Filter interpretiert und die entsprechenden Stellen im Bit-Array $F$ auf Eins gesetzt. Das ist in Abbildung 3.1a dargestellt. Wird ein Element in einem Bloom-Filter angefragt, werden wiederum $k$ Hash-Werte berechnet und die entsprechenden Stellen in $F$ überprüft. Zeigt nur eine Hash-Funktion auf eine Null – wie in Abbildung 3.1b illustriert – ist sicher, dass das Element nicht in den Bloom-Filter eingefügt wurde. Aus diesem Grund liefern Bloom-Filter keine falschnegativen Antworten. Es kann aber vorkommen, dass alle Hash-Werte auf „gesetzte“ Bits zeigen, obwohl das Element nie in den Filter eingefügt wurde. In diesem Fall liefert der Bloom-Filter eine falschpositive Antwort, wie in Abbildung 3.1c erkennbar. Falschpositive Filterantworten entstehen meist deshalb, weil es zu Hash-Kollisionen beim Einfügen unterschiedlicher Elemente kommt; in Abbildung 3.1a ist erkennbar, wie zwei Hash-Funktionen beim Einfügen unterschiedlicher Elemente auf dasselbe Bit im Filter zeigen.

Falschpositive Antworten hängen von der Größe $m$ eines Bloom-Filters, der Anzahl $k$ der Hash-Funktionen und der Anzahl der eingefügten Elemente ab. Die Wahrscheinlichkeit, mit der falschpositive Antworten auftreten, kann berechnet werden. Dazu wird die Wahrscheinlichkeit $p$ benötigt, mit der ein Bit im Bloom-Filter durch Anwenden einer Hash-Funktion nicht gesetzt wird, also Null bleibt:

$$1 - \frac{1}{m}.$$

Nach der Berechnung von jeweils $k$ Hash-Werten für $n$ Elemente sieht diese Wahrscheinlichkeit $p$ folgendermaßen aus:

$$p = \left(1 - \frac{1}{m}\right)^{kn} \approx \exp\left(-\frac{kn}{m}\right). \tag{3.1}$$

Das Gegenereignis dazu – also die Wahrscheinlichkeit mit der ein einzelnes Bit nach dem Einfügen von $n$ Elementen gesetzt ist – ist gegeben durch

$$1 - \left(1 - \frac{1}{m}\right)^{kn}.$$

Damit angenommen wird, ein Element sei in der Menge enthalten, müssen alle $k$ referenzierten Bits im Filter gesetzt sein. Die Wahrscheinlichkeit dafür ergibt sich durch

$$p_{fp} = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^{k}.$$

Dieser Term kann als Wahrscheinlichkeit $p_{fp}$, mit der ein Bloom-Filter eine falschpositive Antwort liefert, durch die folgende Approximation abgeschätzt werden:

$$p_{fp} \approx (1 - p)^{k}. \tag{3.2}$$

Um nun die minimale Wahrscheinlichkeit einer falschpositiven Filterantwort zu bestimmen, wird $p_{fp}$ nach $k$ abgeleitet und aufgelöst. Daraus ergbit sich ein optimales $k_{opt}$:

$$k_{opt} = \frac{m}{n} \log 2, \tag{3.3}$$

woraus der optimale Anteil Nullen im Bit-Array bestimmt werden kann:

$$p_{opt} = \frac{1}{2}. \tag{3.4}$$

Dieser Anteil fließt in die minimale Wahrscheinlichkeit $p_{opt}(fp)$ für falschpositive Filterantworten ein. Dazu wird aus 3.1, 3.2 und 3.3 $p_{opt}(fp)$ bestimmt als

$$p_{opt}(fp) = \left(\frac{1}{2}\right)^{\frac{m}{n} \log 2} \approx 0.6185^{\frac{m}{n}}.$$

Damit kann die Wahrscheinlichkeit für falschpositive Antworten in einigen Fällen zwar unterschätzen werden [75]; für die weiteren Betrachtungen ist diese Abschätzung aber hinreichend genau. Diese Wahrscheinlichkeit wird auch als Fehlerwahrscheinlichkeit, und der Anteil der zu erwartenden falschpositiven Filterantworten als Falschpositivrate bezeichnet. Umgekehrt kann zu einer gewünschten Fehlerwahrscheinlichkeit und der erwarteten Anzahl an Elementen genauso die Größe $m$ des Bloom-Filters bestimmt werden:

$$m = \frac{n \ln p}{(\ln 2)^2} \tag{3.5}$$

Da in die Bestimmung der minimalen Wahrscheinlichkeit $p_{opt}(fp)$ der optimale Anteil Nullen im Bit-Array unmittelbar eingeflossen ist, kann daraus auch auf den optimalen „Füllstand“ eines Bloom-Filtes geschlossen werden:

Die Fehlerwahrscheinlichkeit ist dann minimal, wenn die Hälfte der Bits Null ist (siehe Gleichung 3.4); das Bit-Array eines Filters, dessen Parameter nach diesen Formeln bestimmt wurden, sollte möglichst nur zu etwa 50% gesetzte Bits enthalten.

Diese Konfigurierbarkeit und die Tatsache, dass falschnegative Filterantworten ausgeschlossen sind, haben Bloom-Filter vor allem zu einer beliebten Datenstruktur in Datenbanksystemen gemacht. So setzen zahlreiche aktuelle Datenbank- oder Dateisysteme wie Google BigTable [76], Apache HBase (das ähnlich aufgebaut ist wie Google BigTable und auf dem Hadoop Dateisystem aufsetzt [77]) oder Apache Cassandra [78] an zentraler Stelle auf diese Datenstruktur, um damit überflüssige lineare Suchen in Speicherbereichen zu verhindern.

Da die überflüssigen Speicherzugriffe stets auf eine falschpositive Filterantwort zurückzuführen sind, gibt es zahlreiche Forschungsarbeiten, die Varianten der ursprünglichen Bloom-Filter vorschlagen, um die Wahrscheinlichkeit solcher Fehlzugriffe weiter zu reduzieren. Solche Varianten lassen sich danach unterteilen, ob über dem Suchraum, also der von einem Bloom-Filter beschriebenen Menge, eine bestimmte Ordnung der Elemente existiert, die im Bloom-Filter abgebildet wird, oder nicht. In Abschnitt 3.1.2 werden solche Varianten beschrieben, die keine Ordnung unter den Elementen voraussetzen oder ausnutzen. Abschnitt 3.1.3 stellt dann einige Arbeiten vor, die eine solche Ordnung ausnutzen. Im folgenden Abschnitt werden zunächst Arbeiten vorgestellt, die sich bestimmten Erweiterungen der Operationen auf Bloom-Filtern widmen.

### 3.1.1 Protokollerweiterungen

Die ursprüngliche Definition von Bloom-Filtern umfasst nur Operationen zum Einfügen und Anfragen von Elementen. Werden Elemente aus der zugrundeliegenden Menge entfernt, muss der Bloom-Filter neu aufgebaut werden, um Inkonsistenzen und falschnegative Filterantworten zu verhindern. In Szenarien, in denen die beschriebene Menge häufigen Änderungen unterworfen ist, kann dieser Nachteil den Nutzen von Bloom-Filtern schnell übersteigen.

Ein Beispiel für ein solches Szenario ist der Austausch über lokale Cache-Inhalte in Peer-to-Peer-Netzwerken. In solchen Netzwerken werden Inhalte über mehrere Knoten hinweg von einem Anbieter zum Konsumenten vermittelt. Um häufiger angefragte Daten nicht bei jeder Anfrage durch das gesamte Netzwerk schicken zu müssen, können sie entlang frequentierter Übertragungswege zwischengespeichert werden. Die Übertragungswege können aber nur dann optimiert werden, wenn die Knoten sich gegenseitig über die Inhalte ihres lokalen Zwischenspeichers informieren. Damit dabei möglichst wenig Daten übertragen werden müssen, werden Bloom-Filter eingesetzt. Sie beschreiben den Cache speichereffizient und erlauben den Nachbarknoten dennoch, effiziente Routingentscheidungen zu treffen.

Je nach Caching-Strategie sind aber gerade diese lokalen Zwischenspeicher häufigen Änderungen unterworfen. Ein erneutes Aufbauen des Bloom-Filters kann je nach Komplexität der Netzwerkknoten viel Kapazitäten beanspruchen. Um diesem Problem zu begegenen, wurden mit [79] sogenannte *Counting Bloom-Filter* vorgeschlagen. Sie unterstützen erstmals auch das Löschen von Elementen aus dem Filter.

Dazu wird zusätzlich zum Bloom-Filter eine Übersicht verwaltet, die zählt, welche Bits wie oft von einer Hash-Funktion adressiert wurden. Dieser namensgebende *counter* wird inkrementiert, wenn das korrespondierende Bit beim Einfügen eines Elements getroffen wird. Dementsprechend wird der Zähler dekrementiert, wenn ein Bit beim Löschen eines Elements getroffen wird. Mit dieser Hilfsstruktur kann nun entschieden werden, wann ein Element aus dem Filter gelöscht, also wann ein Bit im Filter auf Null gesetzt werden kann: Das ist erst erlaubt, wenn der Zähler für dieses Bit Null erreicht - wenn also kein anderes Element mehr im Filter enthalten ist, das beim Einfügen dieses Bit gesetzt hat. Nur so hilft diese Zählerstruktur, falschnegative Filterantworten zu verhindern.

Diese Hilfsstruktur wurde in einigen weiteren Arbeiten genauer untersucht. Während in [79] beschrieben wird, dass es ausreicht, den Zähler jedes Bits mit 4 Bits zu repräsentieren, wird in [80] ein Verfahren vorgestellt, mit dem die Hilfsstruktur noch kompakter dargestellt werden kann. Demgegenüber werden in [81] sogenannte spektrale Bloom-Filter vorgestellt, in denen das Bit-Array des Filters durch einen Vektor von Integer-Werten ersetzt wird. Das erhöht zwar den Speicherbedarf des Filters, bringt aber die Unterstützung für Multimengen mit sich, in denen dieselben Elemente mehrmals enthalten sein können. Der spektrale Bloom-Filter unterstützt durch diese Konstruktion neben bekannten Zugehörigkeitsanfragen auch die Anfrage, wie oft ein Element in der Menge enthalten ist. Analog zu den Bloom-Filtern mit Zählern ist auch in spektralen Bloom-Filtern ein Löschen der Elemente möglich. Wegen der Information über die Anzahl der Instanzen eines Elements sind mit spektralen Bloom-Filtern darüber hinaus verteilte *JOIN*-Operationen möglich, wodurch ein Histogramm über zwei auf unterschiedlichen Datenbank-Servern gespeicherten Mengen erstellt werden kann.

In [82] wird ein Verfahren vorgestellt, das auch auf Zähler als Zusatzinformationen zum Bit-Array setzt. Darin wird der Zähler allerdings eingesetzt, um das Altern von Elementen im Filter zu simulieren. In einem sogenannten *Time-Decaying Bloom Filter* werden die Zähler aller Bits periodisch dekrementiert, um Elemente so nach gewisser Zeit automatisch aus dem Filter zu löschen. Ein solcher Filter enthält damit eine Zusammenfassung über die in den letzten Perioden am häufigsten auftretenden Elemente – also die, die noch im Filter enthalten sind oder deren Zähler besonders hoch sind. Diese spezielle Art von Bloom-Filtern wird deshalb oft eingesetzt, um in dem oben beschriebenen Peer-to-Peer-Szenario die beliebtesten Inhalte zu identifizieren, die ein Knoten häufig weiterleiten muss. Im Allgemeinen bieten sich Time-Decaying

Bloom-Filter für die Analyse von Daten-Streams an, also einem kontinuierlichen, (zeitlich) geordneten und in Echtzeit eintreffenden Strom von Daten.

Eine andere Protokollerweiterung wird mit skalierbaren Bloom-Filtern in [83] vorgestellt. Diese Variante heißt deshalb skalierbar, weil sie an die Anzahl der beschriebenen Elemente dynamisch angepasst werden kann. Im Gegensatz zum klassischen Bloom-Filter muss die erwartete Anzahl der einzufügenden Elemente damit nicht mehr im Voraus bekannt sein. Im klassischen Protokoll fließt diese Zahl in die Konfiguration des Bloom-Filters ein und bestimmt dort vor allem über die Größe des Filters bei optimaler Anzahl an Hash-Funktionen (siehe Formel 3.5). Im Gegensatz dazu wird in skalierbaren Bloom-Filtern der Füllstand des Filters überprüft. Überschreitet der einen gewählten Grenzwert, wird ein zusätzlicher Filter eingeführt, der mit einer um einen Faktor $r$ geringeren Wahrscheinlichkeit falschpositive Antworten liefert: Weil $0 < r < 1$ gilt, ist die Falschpositivrate $P_1$ des neuen Filters gegeben durch $P_1 = P_0 r$. Indem die hinzugefügten Filter nun immer strikter konfiguriert werden, kann für einen skalierbaren Bloom-Filter eine Gesamtwahrscheinlichkeit falschpositiver Antworten angegeben werden, die dem Produkt der Einzelwahrscheinlichkeiten aller $l$ Filter entspricht:

$$P = 1 - \prod_{i=0}^{l-1}(1 - P_0 r^i).$$

Damit wurden in diesem Abschnitt Techniken zum Löschen und automatischen Altern von Elementen vorgestellt und solche, die verteilte *JOIN*-Operationen und Histogrammanfragen ermöglichen. Neben diesen Protokollerweiterungen gibt es zahlreiche weitere Arbeiten, die sich mit Bloom-Filtern beschäftigen und vor allem deren Falschpositivrate senken sollen. Die folgenden Abschnitte stellen davon einige vor.

### 3.1.2 Erweiterungen ohne Ordnung auf dem Suchraum

Die Falschpositivrate von Bloom-Filtern kann gesenkt werden, ohne dabei spezielles Wissen über die Daten oder deren Struktur zu verwenden. Vielmehr müssen die Bestandteile der ursprünglichen Definition der Filter variiert werden.

In [84] wird zum Beispiel ein Verfahren beschrieben, wonach die Elemente vor dem Einfügen in einen Bloom-Filter gruppiert und mit unterschiedlichen Hash-Funktionen pro Gruppe in den Filter eingefügt werden. Zur Gruppierung der Elemente wird eine eigene Hash-Funktion $h_0$ verwendet. Die Hash-Funktionen, die dann auf eine Gruppe angewendet werden, werden so gewählt, dass damit der Anteil der gesetzten Bits – der Füllfaktor des Filters – minimiert wird. Dazu werden in $t$ Iterationen jeweils $k$ Hash-Funktionen aus einer Menge von $H$ möglichen Hash-Funktionen gezogen, mit denen die Elemente der Gruppe $j$ zeitweise in den Filter eingefügt werden, um den Füllfaktor für diese Kon-

stellation an Hash-Funktionen zu berechnen. Nach den $t$ Iterationen werden die $k$ Hash-Funktionen ausgewählt, die einen minimalen Füllfaktor verursacht haben. Dieses Verfahren wird für alle $g$ Gruppen wiederholt.

Damit für das Testen von Elementen die gruppenspezifischen Hash-Funktionen verwendet werden, müssen von den $m$ Bits des Bloom-Filters $g * k * \log_2 H$ Bits reserviert werden, um für jede der $g$ Gruppen die $k$ der $H$ zur Verfügung stehenden Hash-Funktionen bestimmen zu können.

Die Experimente haben gezeigt, dass die Falschpositivrate deutlich gesenkt werden kann, je mehr Hash-Funktionen in $H$ potenziell zur Verfügung stehen, je größer die Anzahl der Gruppen $g$ und je mehr Iterationen $t$ aufgewendet werden, um die $k$ besten Hash-Funktionen pro Gruppe auszuwählen. Das gilt zumindest so lange, wie der Bloom-Filter mit $m - g * k * \log_2 H$ freien Bits noch groß genug ist, um alle Elemente aufzunehmen.

Dieses Verfahren eignet sich wegen der aufwändigen Wahl der Hash-Funktionen nur für Einsatzzwecke, in denen ein Bloom-Filter nur selten aufgebaut werden muss. In [84] wird beschrieben, dass sich das Verfahren zum Beispiel für Firewalls eignet, die IP-Pakete entsprechend ihrem Inhalt filtern. Die Filterregeln solcher Firewalls werden in der Regel selten – also einmal täglich oder einmal wöchentlich – aktualisiert, während sie möglichst schnell auf die ankommenden IP-Pakete angewendet werden müssen.

Dieses Verfahren baut auf einer Arbeit von Lumetta und Mitzenmacher auf. In [85] schlagen die Autoren vor, $c$ feste Mengen von jeweils $k$ Hash-Funktionen zu definieren. Zum Einfügen eines Elementes wird die Menge an Hash-Funktionen verwendet, die die wenigsten Bits im Filter setzt. Zum Testen werden alle $c*k$ Hash-Funktionen ausgewertet. Das Element wird im Filter vermutet, sobald es eine Menge von Hash-Funktionen gibt, von der keine auf eine Null zeigt. Mit diesem Verfahren kann ein Bloom-Filter auch in einer Offline-Phase konstruiert werden. Dazu müssen alle Elemente im Voraus bekannt sein. Es wird dann iterativ der Filter mit dem geringsten Füllfaktor ausgewählt.

Gegenüber dem weiter oben beschriebenen Verfahren aus [84], nach dem auf Gruppen von Elementen unterschiedliche Hash-Funktionen angewendet werden, müssen bei dieser Variante zwar ingesamt mehr Hash-Funktionen ausgewertet werden. Gleichzeitig entfällt hier die Reservierung von zusätzlichen Bits, um die verwendeten Hash-Funktionen zu kodieren.

Ein weiteres Verfahren, das ähnlich zu dem von Lumetta und Mitzenmacher auch verschiedene Gruppen von Hash-Funktionen verwendet, wird in [86] vorgestellt. Zum Einfügen eines Elements wird aus $l$ Gruppen von jeweils $k$ Hash-Funktionen eine zufällige Gruppe gezogen und das Element damit wie üblich, ohne eine Optimierung hinsichtlich des Füllfaktors oder ähnlichem, in den Filter eingefügt. Dadurch beschreiben die Bloom-Filter Multimengen, in denen das gleiche Element mehrmals enthalten sein kann. Die Gruppen der Hash-Funktionen dienen hier der Abschätzung der Häufigkeit eines Elements. Bei der Anfrage nach einem Element werden deshalb alle $l$ Gruppen überprüft und gezählt, wie viele der Gruppen das Element im Filter vermuten.

Mit dem Gaußschen Bloom-Filter wird in [242] ein weiteres Verfahren beschrieben, das ohne Zusatzwissen über die Daten die Falschpositivrate von Bloom-Filtern reduzieren kann. Das soll im Folgenden detailliert betrachtet werden.

#### 3.1.2.1 Gaußsche Bloom-Filter

Bei näherer Betrachtung der Bloom-Filter fällt auf, dass nur nicht gesetzte Bits dazu beitragen, falschpositive Filterantworten zu verhindern. Eine falschpositive Antwort wird schließlich nur verhindert, wenn mindestens eines der untersuchten Hash-Felder nicht gesetzt ist, also eine Null enthält. Insbesondere diese Beobachtung motiviert die Erweiterung zum Gaußschen Bloom-Filter. Ziel ist es, mehr Informationen in die Felder hineinzukodieren, die eine Null enthalten: In benachbarten „Null-Feldern“ soll gespeichert werden, welche Hash-Funktion das umschlossene Bit gesetzt hat. Anhand der Nachbarfelder kann bei einer Anfrage an den Filter entschieden werden, ob das geprüfte Bit beim Speichern des Anfrage- oder eines anderen Elements gesetzt wurde. Wenn erkannt werden kann, dass das untersuchte Bit von einer anderen Hash-Funktion gesetzt worden ist, kann eine falschpositive Filterantwort verhindert werden.

Zur Konstruktion Gaußscher Bloom-Filter wird das Bit-Array $F$ traditioneller Bloom-Filter durch ein Array kleiner Fließkommazahlen ersetzt, das die feste Länge $m$ behält. Zur Initialisierung enthalten alle Zellen weiterhin eine Null, und das leere Hash-Feld wird mit $F = 0$ bezeichnet. Gegeben ist zudem eine Menge $h_i$ von $k$ paarweise unabhängigen Hash-Funktionen, die ein Universum $U$ auf die Menge $\underline{m} = \{1, 2, \ldots, m\}$ abbilden. Neben der Anpassung des Arrays werden nur die Einfüge- und Testoperation angepasst, die auf diesem Hash-Feld $F$ wie folgt definiert werden:

1. Insert($F$, $e$) zum Einfügen eines Elements $e$ in das Hash-Feld $F$:

   Erstelle für jede der $k$ Hash-Funktionen $h_i$ eine Gaußsche Wahrscheinlichkeitsdichtefunktion

   $$\mathcal{N}_i = \mathcal{N}(h_i(e), i)$$

   für die der Mittelwert $\mu = h_i(e)$ durch den Wert der Hash-Funktion und deren Standardabweichung $\sigma = i$ durch den Index der Hash-Funktion bestimmt wird. Im Hash-Feld $F$ werden alle Einträge auf das Maximum des bereits im Hash-Feld gespeicherten Werts und des Werts der *normalisierten Signaturfunktion*

   $$\tilde{\mathcal{N}_i} = \frac{\mathcal{N}_i}{\max(\mathcal{N}_i)}$$

   gesetzt, die ihr Maximum von 1 bei $h_i(e)$ und sonst Werte zwischen 0 und 1 annimmt:

   $$F[t] := \max\left(F[t], \tilde{\mathcal{N}_1}(t), \ldots, \tilde{\mathcal{N}_k}(t)\right)$$

2. Test($F$, $e$) zum Anfragen des Hash-Felds $F$ mit einem Element $e$:

   Erstelle dieselbe normalisierte Signaturfunktion $\tilde{\mathcal{N}}_i$ und erzeuge daraus die Elementsignatur $S$ durch Auswählen des Maximums aus jedem Slot:

   $$S[t] = \max\left(\tilde{\mathcal{N}}_1(t), \ldots, \tilde{\mathcal{N}}_k(t)\right)$$

   Beantworte die Anfrage genau dann positiv, wenn $F[i] \geq S[i]$ für alle $i \in \underline{m}$.

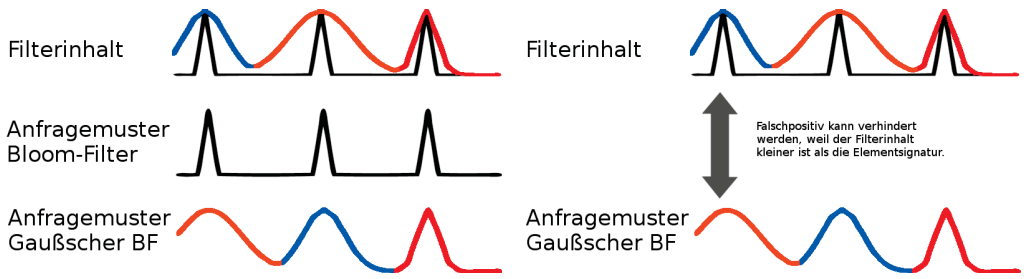
Durch diese Konstruktion wird in den Feldern um ein Maximum herum der Index der Hash-Funktion gespeichert, die das umschlossene Maximum (also das umschlossene gesetzte Bit) verursacht hat. In Filtern mit wenigen Elementen stehen für diese Informationen viele Felder zur Verfügung, da nur wenige Felder ein Maximum enthalten. Erreicht der Füllfaktor des Filters den bereits genannten optimalen Wert von ungefähr 50%, stehen weniger Felder für diese Zusatzinformationen zur Verfügung.

Zum Speichern dieser Informationen bietet sich der Gaußsche Kern besonders an. Es könnten natürlich auch andere Funktionen verwendet werden, die mit entsprechender Parameterisierung den Index der Hash-Funktion kodieren. Indem aber eine Funktion eingesetzt wird, die nur ein Maximum annimmt und deren Maximum in dem Feld liegt, auf das der Wert der Hash-Funktion verweist, kann auf einfache Weise der traditionelle Bloom-Filter extrahiert werden: Aus einem gegebenen Gaußschen Bloom-Filter $F_G$ kann der zugrundeliegende Bloom-Filter $F_B$ extrahiert werden, indem alle Slots mit dem Maximalwert $M$ der Signaturfunktion verglichen werden:

$$F_B[i] = (F_G[i] == M)$$

Auf diese Weise bleibt der klassische Bloom-Filter mit identischer Konfiguration (Größe $m$ und $k$ Hash-Funktionen) erhalten. Der so extrahierte Filter $F_B$ hat zudem dieselbe Ausprägung wie ein Filter, der mit dem traditionellen Verfahren befüllt worden wäre. Damit ist die Gaußsche Erweiterung kompatibel zu allen Anwendungen, in denen traditionelle Bloom-Filter zum Einsatz kommen. Insbesondere in verteilten Systemen kann der Gaußsche Bloom-Filter eingesetzt werden, ohne dass seine Verwendung für alle Komponenten bindend würde. Vielmehr kann er genau so lange eingesetzt werden, wie der Einsatz einen spürbaren Vorteil bringt und der zusätzliche Speicherplatz zur Verfügung steht.

Analog zum klassischen Bloom-Filter liefern auch Gaußsche Bloom-Filter keine falschnegativen Filterantworten. Denn: zu jedem beliebigen Zeitpunkt sind alle Werte der Filterstruktur $F$ größer als die der größten eingefügten Elementsignatur. Zudem basiert die Test-Operation auf einem Vergleich einer Elementsignatur mit dem Filter und liefert nur dann ein positives Ergebnis,




(a) Anfrage an einen Gaußschen Bloom-Filter

(b) Der Gaußsche Bloom-Filter verhindert eine falschpositive Antwort

Abbildung 3.2: Der Gaußsche Bloom-Filter kann eine falschpositive Filterantwort des klassischen Bloom-Filters verhindern

wenn alle Felder der Signatur größer oder gleich der Felder im Filter sind. Die Signatur des Anfragemusters ist aber nur dann in jedem Feld größer als der Filter, wenn das Anfrageelement nicht im Filter enthalten ist. In diesem Fall liefert die Test-Operation ein negatives Ergebnis.

Der Gaußsche kann genauso wie der traditionelle Bloom-Filter falschpositive Antworten liefern. Eine falschpositive Antwort des Gaußschen Filters impliziert gleichzeit eine falschpositive Antwort des darunterliegenden traditionellen Filters, oder anders ausgedrückt: Die Falschpositivrate eines Gaußschen Bloom-Filters ist kleiner oder gleich der Falschpositivrate eines traditionellen Bloom-Filters. Ein Gaußscher Bloom-Filter liefert dann eine falschpositive Antwort, wenn alle Felder des Filters größer sind als die der Elementsignatur des Anfrageelements. Das gilt insbesondere für die Felder, in denen die Signatur den Maximalwert 1 annimmt. Diese Felder wurden allerdings von Hash-Funktionen anderer Elemente direkt adressiert.

Diese beiden Überlegungen werden in der Abbildung 3.2 beispielhaft skizziert. Es wird eine Situation beschrieben, in der mit der Gaußschen Erweiterung eine falschpositive Filterantwort klassischer Bloom-Filter verhindert werden kann.

Angenommen, das Element $e_1$ wurde mit den Hash-Werten $h_1(e_1) = 50$, $h_2(e_1) = 10$ und $h_3(e_1) = 30$ in einen leeren Filter eingefügt. Dieser Filter wird nun mit einem Element $e_2$ angefragt, das die Hash-Werte $h_1(e_2) = 50$, $h_2(e_2) = 30$ und $h_3(e_2) = 10$ annimmt. Diese Situation wird in Abbildung 3.2 auf der linken Seite dargestellt.

Nachdem alle drei adressierten Hash-Felder eine Eins enthalten, würde die Test-Operation eines klassischen Bloom-Filters eine positive und damit falschpositive Antwort für Element $e_2$ liefern. Das wird mit der Gaußschen Erweiterung verhindert. Anhand des Gaußschen Bloom-Filters können die Hash-Funktionen unterschieden werden, die die gesetzten Bits im Hash-Feld verursacht haben. Die rechte Seite der Abbildung zeigt, wie die Signatur des Anfrageelements sich vom Filter unterscheidet. Insbesondere an den Hash-Adressen 10 und 30 unterscheiden sich die Muster in ihrer Standardabwei-

chung. Daraus kann geschlossen werden, dass die Bits von unterschiedlichen Hash-Funktionen gesetzt worden sein müssen und deshalb von unterschiedlichen Elementen stammen.

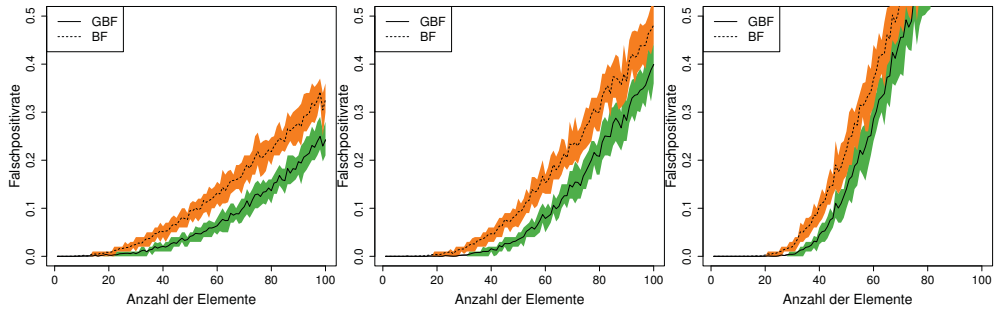
An dieser Stelle ist noch offen, wie ein Gaußscher Bloom-Filter konfiguriert werden soll. Eine zentrale Annahme bei der Analyse der Bloom-Filter ist die Unabhängigkeit der Informationen von anderen Bits. Diese Unabhängigkeit ist mit der Gaußschen Erweiterung gewissermaßen verletzt, da auch der Inhalt anderer Felder des Filters zu dem Ergebnis der Test-Operation beiträgt. Aufgrund der Erkenntnis, dass ein Gaußscher Bloom-Filter keine schlechteren Ergebnisse liefert als der zugrundeliegende Bloom-Filter, können zur Konfiguration der Größe des Filters und der Anzahl der Hash-Funktionen die bekannten Methoden klassischer Filter verwendet werden.

Damit die Kompatibilität zu traditionellen Bloom-Filtern insgesamt erhalten bleibt, ist ein Punkt allerdings besonders zu beachten: Sowohl die Signaturfunktion als auch die Auflösung und Kodierung der Fließkommazahlen sollte so gewählt werden, dass keine Zelle, die nicht den Maximalwert enthält, auf den Maximalwert aufgerundet wird. Insbesondere wenn für die Filteroperationen Recheneinheiten wie *Graphics Processing Units (GPUs)* eingesetzt werden, die für die Berechnung von Fließkommazahlen optimiert sind, können durch ungewolltes Runden zusätzliche falschpositive Filterantworten beim zugrundeliegenden Bloom-Filter provoziert werden.

#### 3.1.2.2 Evaluation Gaußscher Bloom-Filter

Für eine allgemeine Evaluation der vorgestellten Datenstruktur hinsichtlich ihrer Falschpositivrate wurden mehrere Experimente im Vergleich zum traditionellen Bloom-Filter durchgeführt. Dazu wurden eher hohe Falschpositivraten provoziert, um mit kleineren Filtern und einer geringeren Anzahl an Hash-Funktionen die Situation zu simulieren, in der größere Filter deutlich mehr Elemente enthalten. Für die folgenden Experimente wurden stets Mengen zufällig erzeugter Strings verwendet. Als Hash-Funktionen kamen entweder SHA-1 [87] oder der Murmur-Hash [88] mit unterschiedlichen Präfixen zum Einsatz, um darüber unterschiedliche Hash-Funktionen zu erzeugen.

Abbildung 3.3 vergleicht den Gaußschen Filter jeweils mit einem äquivalenten traditionellen Bloom-Filter. Für alle Grafiken wurden die Experimente mit zufälligen Strings und variierenden Parametern häufig wiederholt, um einen aussagekräftigen Median über die gemessenen Falschpositivraten bestimmen zu können. Zusätzlich zu diesem Median zeigen die Grafiken das erste und dritte Quantil farblich hinterlegt. In der Abbildung ist erkennbar, dass der Gaußsche Bloom-Filter eine deutlich geringere Falschpositivrate erzeugt als der traditionelle, solange ausreichend Nullen im Filter vorhanden sind. In der linken Grafik standen im Bloom-Filter 256 Zellen zur Verfügung und Elemente wurden mit $k = 3$ Hash-Funktionen eingefügt. Für die rechten beiden Grafiken wurden dann $k = 5$, beziehungsweise $k = 10$ Hash-Funktionen verwendet. Die Größe der Filter blieb in allen Experimenten unverändert. Mehr




(a) $k = 3$ (b) $k = 5$ (c) $k = 10$

Abbildung 3.3: Vergleich zwischen Bloom-Filter (BF) und Gaußschem Bloom-Filter (GBF) ($m = 256$, $k \in \{3, 5, 10\}$ und $n \in \{1, \ldots 100\}$)

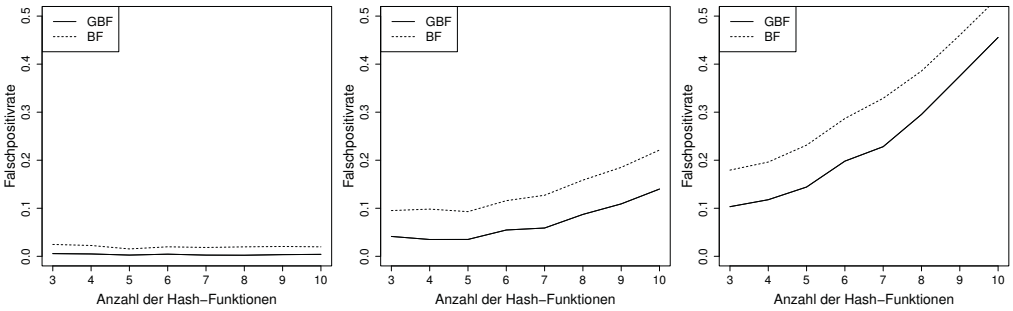
Hash-Funktionen erzeugen natürlich mehr Einsen im Filter und führen auch zu einer allgemein höheren Falschpositivrate in beiden Filtern. Mit mehr Einsen im Filter stehen aber auch weniger Felder mit nicht-maximalen Einträgen zur Verfügung, in denen Informationen über die Hash-Funktionen gespeichert werden könnten. Das führt dazu, dass der Vorteil des Gaußschen im Vergleich zum traditionellen Filter etwas abnimmt.

Der positive Effekt der Gaußschen Filter bleibt auch erhalten, wenn die Anzahl der Hash-Funktionen variiert wird. Abbildung 3.4 zeigt die Falschpositivraten bei unterschiedlicher Anzahl an Hash-Funktionen. Es ist erkennbar, dass Gaußsche Filter auch hier besser abschneiden als traditionelle Bloom-Filter. Der Vorteil liegt vor allem darin begründet, dass die Filter in den gezeigten Experimenten mit $n = 30$, $n = 50$ und $n = 70$ Elementen deutlich unter ihrer optimalen Füllrate blieben. Insgesamt unterstreichen diese Experimente aber die Empfehlung, Gaußsche Filter so zu konfigurieren wie klassische Bloom-Filter.

In [242] wurden weitere Optimierungen beschrieben und umfassend ausgewertet. So kann beispielsweise die CPU von den aufwändigeren Berechnungen des Gaußschen Filters entlastet werden, indem die Gaußschen Bloom-Filter in einem Grafikprozessor (*graphic processing unit, GPU*) konstruiert werden. Dadurch kann auf den zusätzlichen Speicherplatz der Gaußschen Filter echt parallel und ohne Synchronisationsaufwand zugegriffen werden. Das beschleunigt insbesondere die Konstruktion der Filter und erhöht die Performanz der Erweiterung deutlich.

Außerdem wurde beschrieben, wie die Berechnung des Gaußschen Kerns auf der CPU optimiert werden kann. Dazu wurde zum einen die „$3 - \sigma$-Regel“ vorgestellt. Da die Gauß-Funktion außerhalb des Intervals $[\mu - 3\sigma, \mu + 3\sigma]$ Werte annimmt, die kleiner sind als 0.0045 und die deshalb in diesem Szenario vernachlässigt werden können, wird die Funktion außerhalb des Intervals auch nicht ausgewertet.

Zum anderen wurde das Verfahren durch eine bestimmte Approximation der Gauß-Funktion optimiert. Dazu wird die Exponentialfunktion über eine




(a) $n = 30$ (b) $n = 50$ (c) $n = 70$

Abbildung 3.4: Der Einfluss der Anzahl der Hash-Funktionen für Gaußsche (GBF) und normale Bloom-Filter (BF) ($m = 256$, $n \in \{30, 50, 70\}$ und $k \in \{3, \ldots 10\}$)

Methode von Schraudolph berechnet, die nur Multiplikationen ganzer Zahlen, Additionen und Shift-Operationen verwendet [89].

Schließlich wurde auch eine spezielle Bit-Kodierung für Fließkommazahlen vorgestellt, mit der bereits mit 4 Bit pro Hash-Zelle eine Falschpositivrate erzeugt werden konnte, die zu einer Implementierung mit *double*-Genauigkeit vergleichbar ist. Das macht diese Erweiterung auch für Anwendungen attraktiv, in denen der zusätzliche Speicherbedarf für *double*- oder sogar *single*-Genauigkeit nicht zur Verfügung steht.

Auf diese Optimierungen soll an dieser Stelle nicht weiter eingegangen werden. Vielmehr soll mit dem folgenden Abschnitt die Auswertung des Verfahrens in der Datenbank-Software Cassandra vorgestellt werden.

#### 3.1.2.3 Gaußsche Bloom-Filter und Apache Cassandra

Um die Gaußsche Erweiterung in einem realitätsnahen Szenario zu evaluieren, wurde das Datenbanksystem Apache Cassandra [78] erweitert, dessen Quellcode frei verfügbar ist. Cassandra basiert auf dem verteilten Dateisystem Hadoop (HDFS) [77], das von der Firma Facebook entwickelt wurde. Inzwischen ist Cassandra auch bei Twitter, eBay, Netflix und vielen anderen im Einsatz [230].

Aufbauend auf dem Hadoop Dateisystem verwaltet Cassandra einen *key-value-store*, über den Werte in der Datenbank gespeichert und über einen Schlüssel wieder abgerufen werden können. Die Daten werden in sogenannten `SSTables` organisiert. Um den Zugriff auf diese Speicherbereiche zu reduzieren, werden Bloom-Filter eingesetzt. Sie beschreiben jeweils einen solchen Speicherbereich. Mit ihrer Hilfe wird entschieden, ob ein gesuchter Schlüssel in dem dazugehörigen Speicherbereich abgelegt wurde. Daher wird immer, wenn ein Schlüssel in einer `SSTable` gespeichert wird, der assoziierte Bloom-Filter aktualisiert. Cassandra verwaltet ein ausgeklügeltes System, mit dem Bloom-Filter dynamisch rekonfiguriert und an die benötigte Kapazität angepasst werden.

| Filter | Bloom-Filter | Gaußsche Filter | Verhältnis |
|---|---|---|---|
| 1 | 263,937 | 141,598 | 0.54 |
| 2 | 625,879 | 524,120 | 0.84 |
| 3 | 639,631 | 539,161 | 0.84 |
| 4 | 1,230,417 | 1,163,531 | 0.94 |
| 5 | 440,584 | 336,882 | 0.76 |
| 6 | 271,404 | 146,835 | 0.54 |
| 7 | 255,446 | 138,246 | 0.54 |
| 8 | 305,535 | 172,626 | 0.56 |
| 9 | 260,190 | 138,611 | 0.53 |
| Total | 4,293,023 | 3,301,610 | 0.67 |

Tabelle 3.1: Anzahl der Lesevorgänge von SSTables für jeden Bloom-Filter im Vergleich zum Gaußschen Filter während eines Stresstests

Für die folgenden Experimente wurde der klassische Bloom-Filter durch eine Gaußsche Erweiterung ersetzt. Der traditionelle Filter wurde als Instanzvariable der Erweiterung behalten, um das Verhalten des Systems nachvollziehen und so eine Grundwahrheit erzeugen zu können. Das hat allerdings zur Folge, dass aus dem integrierten Stresstest keine Rückschlüsse auf die Performanz des Systems gezogen werden können, da alle Operationen jeweils für beide Filter ausgeführt wurden.

Die Experimente wurden auf einem *single-node cluster*, also einem Netz von Speicherknoten, das aus einem einzelnen Rechner bestand, durchgeführt. Die Modifikationen wurden an der Cassandra Version 2.0.4 vorgenommen. Um möglichst viele Fälle zu provozieren, in denen das Verhalten zwischen Gaußscher Erweiterung und klassischem Bloom-Filter verglichen werden kann, wurde Cassandra angewiesen, eine Falschpositivrate von 10% zuzulassen – üblich und voreingestellt sind 1%. Weiterhin wurde Code eingefügt, der Informationen über den Zustand des Systems, die Anweisungen an die einzelnen Instanzen der Bloom-Filter und die Ergebnisse der jeweiligen Instanzen während der Laufzeit ausgibt. Ausgegeben werden sollten vor allem die Situationen, in denen die Bloom-Filter eine falschpositive Antwort vermeiden und so einen unnötigen Zugriff auf eine `SSTable` verhindern konnten. Auf dieser präparierten Version von Cassandra wurde der mitgelieferte Stresstest ausgeführt. Dieser schreibt zuerst eine Million Schlüssel in die Datenbank, rekonfiguriert die Bloom-Filter entsprechend mithilfe des ebenfalls mitgelieferten Tools *nodetool scrub* und liest schließlich eine Million zufällige Schlüssel aus der Datenbank aus.

Während dieser Evaluation auf einem single-node cluster wurden 23 Bloom-Filter erzeugt, von denen neun verwendet wurden, um die Performanz des Systems zu verwalten. Die übrigen Filter waren zur Schlüsselverwaltung eingesetzt und verzeichneten nur wenige Operationen. Tabelle 3.1 listet die Anzahl der Situationen auf, in denen beide Filter den Zugriff auf eine `SSTable` nicht verhindern konnten. Die Zahlen umfassen positive und falschpositive Zugriffe.

Insgesamt konnten mit der Gaußschen Erweiterung 991413 Zugriffe auf `SSTables` im Vergleich zu gleich konfigurierten klassischen Bloom-Filtern verhindert werden – das entspricht einem Anteil von durchschnittlich 33%. Erkennbar ist auch, dass während der Rekonfiguration der Bloom-Filter, bevor die Lesephase des Stresstests beginnt, beinahe 50% der Zugriffe auf eine `SSTable` verhindert werden konnten, die mit klassischen Filtern nicht verhindert werden konnten.

Die Performanz der Operationen wurde ebenfalls überwacht. Aus bereits beschriebenen Gründen war das nicht mit dem mitgelieferten Stresstest möglich, obwohl dieser entsprechende Funktionalitäten bereithält. Daher wurden zusätzliche Funktionen zur Zeitmessung eingefügt, mit der die Laufzeit der Einfüge- und Test-Operationen beider Filter gemessen werden konnte. Erwartungsgemäß benötigte der Gaußsche Filter mehr Zeit zur Berechnung. Insbesondere die Laufzeit für Einfüge-Operationen wich von der entsprechenden Operation des klassischen Filters ab. Hier benötigte die Operation der Gaußschen Erweiterung 42% mehr Zeit. Das liegt unter anderem daran, dass der klassische Bloom-Filter nur die adressierten Bits auf Eins setzen muss, während der Gaußsche Bloom-Filter insbesondere bei steigender Anzahl an Hash-Funktionen auf deutlich mehr Hash-Zellen zugreifen muss. Demgegenüber war der Anstieg der gemessenen Laufzeit der Test-Operation des Gaußschen Filters gegenüber der des klassischen Filters mit $8,82\%$ moderat. Das liegt daran, dass die Test-Operation des Gaußschen Filters abgebrochen werden kann, bevor alle potenziell betroffenen Bits untersucht worden sind. Diese Beobachtung macht die Gaußsche Erweiterung besonders für Datenbankapplikationen interessant, in denen deutlich mehr Lese- als Schreiboperationen zu erwarten sind.

Nach dieser detaillierten Beschreibung des Gaußschen Bloom-Filters folgen nun einige andere Varianten, die von einer Ordnung auf dem Suchraum ausgehen. Die folgenden Erweiterungen reduzieren die Falschpositivrate des Filters oder bilden spezielle Einsatzzwecke mit Bloom-Filtern ab, indem sie die Ordnung unter den Elementen auf bestimmte Art und Weise ausnutzen.

### 3.1.3 Erweiterungen mit Ordnung auf dem Suchraum

Die Arbeiten, die in diesem Abschnitt vorgestellt werden, verwenden alle eine Form von zusätzlichem Wissen über die Daten, um die Falschpositivrate von Bloom-Filtern zu senken. Die *gewichteten* Bloom-Filter beispielsweise nutzen die Tatsache aus, dass nach den Elementen in einem Bloom-Filter in der Regel unterschiedlich häufig gesucht wird [90]. Die Autoren stützen sich dabei auf Untersuchungen des Datenverkehrs in Netzwerken, wonach sich ein erheblicher Teil der übertragenen Daten auf wenige beliebte Dateien konzentriert.

Das Wissen über die Beliebtheit einzelner Elemente oder Kategorien von Elementen wird dann verwendet, um die Anzahl der Hash-Funktionen zu bestimmen, mit der Elemente in den Filter eingefügt werden: Je häufiger ein Element oder eine Kategorie von Elementen wahrscheinlich angefragt werden

wird, desto mehr Hash-Funktionen werden zum Einfügen des Elements oder eines Elements dieser Kategorie verwendet. Die genannte Arbeit zeigt, dass sich die Falschpositivrate des gewichteten im Vergleich zum traditionellen Bloom-Filter verbessert, also verringert, je unterschiedlicher die Anfragehäufigkeiten zwischen den Elementen ausfallen.

In *hierarchischen* Bloom-Filtern wird eine andere Art von Struktur unter den Daten vorausgesetzt [91]. Diese Variante eignet sich vor allem für das Auffinden von Teilzeichenketten unterschiedlicher Länge. Über die Daten wird hier also angenommen, dass es sich um Zeichenketten handelt, die in Blöcken unterschiedlicher Länge und in unterschiedlichen Konkatenationen auftreten können. Die Hierarchie wird dabei mit mehreren Bloom-Filter über die Konkatenationen der Teilketten aufgebaut.

Zunächst wird eine Zeichenkette, die mit einem hierarchischen Bloom-Filter beschrieben werden soll, in $s$ Blöcke der Länge $\lceil p/s \rceil$ unterteilt, wobei die Zeichenkette insgesamt $p$ Zeichen lang ist. In den Filter der untersten Ebene der Hierarchie werden die $s$ Blöcke jeweils einzeln eingefügt; in die Filter der darüber liegenden Ebenen werden die Blöcke als paarweise Konkatenationen der vorherigen Ebene eingefügt. Angenommen, bei der Teilung der Zeichenkette sind vier Blöcke $S_0S_1S_2S_3$ entstanden. In den Filter der untersten Ebene werden dann die vier Teilketten einzeln eingefügt. Die nächste Ebene wird mit den zwei Konkatenationen $S_0S_1$ und $S_2S_3$ befüllt. In den Filter der obersten Ebene wird schließlich die gesamte Kette $S_0S_1S_2S_3$ als erneute paarweise Konkatenation der vorherigen Ebene eingefügt.

Diese Hierarchie von Bloom-Filtern erlaubt die Suche nach unterschiedlichen Konkatenationen der Teilzeichenketten und damit einen simplen Mustervergleich. Beispielsweise kann nach einer Zeichenkette der Form $S_0S_1 * S_3$ gesucht werden. Dazu wird mithilfe der Filter der untersten Ebene zunächst die Existenz der drei einzelnen Blöcke überprüft. Auf den höheren Ebenen kann dann die Existenz von $S_0S_1$ geprüft werden. Und indem intelligente Annahmen über $*$ getroffen werden, kann schließlich sogar die Zeichenkette $S_0S_1S_xS_3$ auf der obersten Ebene der Hierarchie überprüft werden. Mit traditionellen Bloom-Filtern wäre hier nur eine Überprüfung der einzelnen Blöcke möglich gewesen.

Eine weitere Variante von Bloom-Filtern setzt auf distanzsensitive Hash-Funktionen [92]. Solche *distanzsensitiven* Bloom-Filter sollen zu einem Anfrageelement beantworten können, ob es zu einem Element aus der durch den Bloom-Filter beschriebenen Menge näher liegt als zu anderen: Gegeben einen metrischen Raum $(U, d)$, eine finite Menge $S \subset U$ und den Distanzparametern $0 \leq \epsilon < \delta$ kann ein distanzsensitiver Bloom-Filter zwischen Eingaben so unterscheiden, dass für $u \in U$ und einige $x \in S$ gilt $d(u, x) \leq \epsilon$ und für $u' \in U$ und alle $x \in S$ gilt $d(u', x) \geq \delta$.

Für diese Bloom-Filter Variante werden sogenannte *locality-sensitive hash functions* eingesetzt, die als ähnlichkeitserhaltende Hash-Funktionen bezeichnet werden. Sie unterscheiden sich von herkömmlichen oder kryptographischen Hash-Funktionen dadurch, dass geringe Änderungen an den Eingabewerten ge-

ringe oder keine Änderungen an den Ausgabewerten erzeugen. Diese ähnlichkeitserhaltenden Hash-Funktionen versuchen gewissermaßen die Wahrscheinlichkeit für Hash-Kollisionen zu maximieren. Eingesetzt werden sie zum Beispiel zur Dimensionsreduktion oder der Suche nach nächsten Nachbarn [93], [94].

Während in traditionellen Bloom-Filtern mit den durch die klassischen kryptographischen Hash-Funktionen adressierten Bits nur entschieden wird, ob ein Element im Filter enthalten ist oder nicht, ist in diesen Filtern mit ähnlichkeitserhaltenden Hash-Funktionen auch die Anzahl der gesetzten Bits von Bedeutung. Über die kann nämlich entschieden werden, ob der Filter ein Element enthält, zu dem das Anfrageelement besonders ähnlich ist. Je mehr der adressierten Bits bei einer Anfrage bereits gesetzt sind, desto größer ist die Ähnlichkeit des Anfrageelements zu einem bereits eingefügten Element.

Das bedeutet aber, dass die nicht gesetzten Bits ihre distinktive Aussagekraft verlieren: Durch die ähnlichkeitserhaltenden Hash-Funktionen können distanzsensitive Bloom-Filter falschnegative Antworten liefern. Falls die genannten Hash-Funktionen sehr ähnliche Objekte auf benachbarte – also nahe, aber unterschiedliche – Hash-Adressen abbilden, kann der Filter den Schluss zulassen, dass kein ähnliches Objekt enthalten ist, obwohl eines eingefügt wurde.

Dafür gewinnen distanzsensitive Filter aber eine neue interessante Eigenschaft. Wenn beispielsweise $U = \sum^l$ und die Distanzfunktion $d$ die relative Hamming-Metrik ist, repräsentiert die Anzahl der bei einer Anfrage adressierten gesetzten Bits die Hamming-Distanz zwischen dem Anfrageelement und einem Element im Filter. Diese Eigenschaft kann für zahlreiche Anwendungen interessant sein, in denen eine probabilistische Suche nach ähnlichen Elementen in einer Menge vonnöten ist – die Autoren nennen beispielsweise die Suche nach ähnlichen DNA-Sequenzen.

Neben diesen vorgestellten verwandten Arbeiten, ist auch im Rahmen dieser Arbeit für diese Kategorie eine Erweiterung entstanden: sogenannte OVSF-kodierte Bloom-Filter [245]. Dazu werden die in der Mobilkommunikation gebräuchlichen OVSF-Codes eingesetzt, um eine eindeutige Kombination von Hash-Funktionen zu erzeugen. Indem eine solche Kombination zum Einfügen von und Suchen nach Elementen eingesetzt wird, wird ein Einfügen und Suchen an einem Ort in der durch die OVSF-Codes repräsentierten Ordnung modelliert. Dieses Verfahren wird nun im Detail vorgestellt.

#### 3.1.3.1 Modellierung einer Ordnung mit OVSF-Codes

Der *Orthogonal Variable Spreading Factor* (OVSF) bezeichnet ein Verfahren zur Trennung von Signalen bei der Nachrichtenübertragung mit unterschiedlichen Datenübertragungsraten. Damit sind OVSF-Codes – oder auch orthogonale Spreizcodes – eine beliebte Implementierung des *Code Division Multiple Access (CDMA)*, einem Codemultiplexverfahren, mit dem in drahtlosen Kommunikationsverfahren die gleichzeitige Übertragung verschiedener Datenströme über die Luftschnittstelle ermöglicht und organisiert wird. So werden

OVSF-Codes beispielsweise im Mobilfunkstandard UMTS eingesetzt, um Teilnehmern einer Mobilfunkzelle eine bestimmte Datenübertratungsrate zuzuweisen.

OVSF-Codes sind binäre Codes, die einen vollständigen Binärbaum bilden. Ausgehend von einem Wurzelknoten $K_0 = 0$ oder $K_0 = 1$ werden die Elemente der Knotenmenge $\{K_i\}$ rekursiv erzeugt:

$$children\,(K_i) = \begin{cases} K_i K_i \\ K_i \neg K_i \end{cases} \tag{3.6}$$

Durch die so erzeugte hierarchische Struktur haben OVSF-Codes zwei wichtige Eigenschaften, die auch mit dem vorliegenden Verfahren ausgenutzt werden:

1. Orthogonalität und
2. Generalisierung.

Weil OVSF-Codes zunächst einmal einen vollständigen Binärbaum bilden, liefern sie damit eine generische hierarchische Struktur. Mit dieser Struktur kann ein beliebiges Universum partitioniert werden. Der OVSF-Baum kann zum Beispiels semantische Ontologien, einen Routing-Graphen oder die Organisationshierarchie eines Unternehmens repräsentieren. Die tatsächliche Bedeutung hängt vom jeweiligen Anwendungsfall ab. In dem hier vorgestellten Verfahren wird dieser Baum verwendet, um solche Informationen in ein Bit-Array fester Länge zu kodieren, die durch eine binäre Hierarchie strukturiert werden können.

Die OVSF-Codes einer Ebene sind alle wechselseitig orthogonal zueinander, ihr gemeinsames Skalarprodukt ist 0. Im Codemultiplexverfahren CDMA werden Codes einer Ebene deshalb einmalig an die Sender in einem Netzwerk vergeben. Jede Mobilstation multipliziert die eigenen Signale vor der Übertragung mit dem ihr zugewiesenen Code. Die Signale werden so über ein breites Frequenzspektrum gestreut und interferieren nicht mehr mit der Übertragung anderer Mobilfunkstationen. In dem vorliegenden Ansatz ist die Orthogonalität der Codes entscheidend, weil jeder Code dadurch eindeutig ist. Jeder Code kann so eine eindeutige Kombination von Hash-Funktionen erzeugen, mit der ein Einfügen eines Elements an dem entsprechenden Ort im Baum modelliert wird.

Durch die oben beschriebene rekursive Konstruktion sind die OVSF-Codes aller Ebenen zudem generalisierbar. Das bedeutet, dass jeder Knoten im Baum den gesamten Code aller seiner Elternelemente enthält. Im vorliegenden Verfahren werden Elemente, die an einem Ort in der Struktur über die für diesen Ort charakteristische Kombination von Hash-Funktionen eingefügt werden, deshalb auch an jedem übergeordneten Ort eingefügt, da stets auch alle Hash-Funktionen der Elternknoten verwendet werden. Das ist insbesondere für die Suche nach Elementen von Vorteil: Elemente können so auch an inneren Knoten des Baums gesucht werden, obwohl sie in einem Blattknoten eingefügt

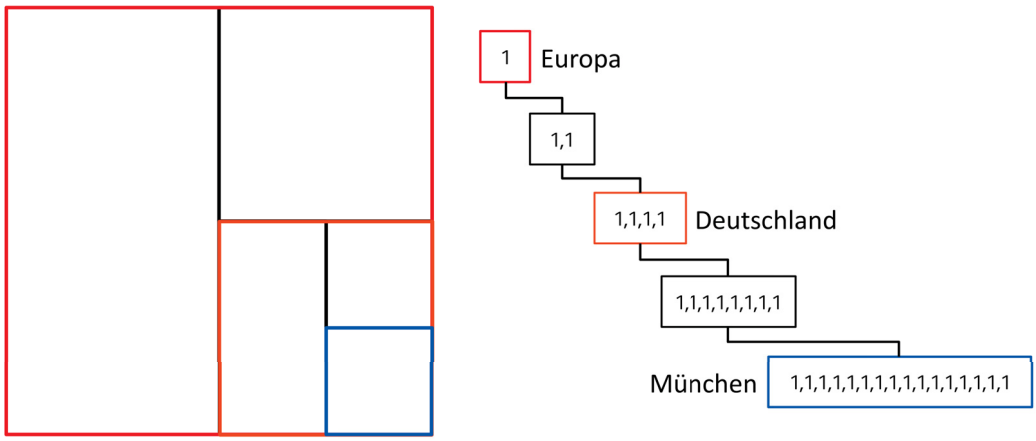



Abbildung 3.5: Ein OVSF-Code-Baum und die damit mögliche Segmentierung einer geographischen Region

wurden. Repräsentiert der Binärbaum beispielsweise eine geografische Region, kann mit dem Code gesucht werden, der eine ganze Stadt repräsentiert, während das gesuchte Element mit einem Kind dieses Knotens – etwa an einer Straßenadresse – eingefügt wurde.

Für die folgenden Betrachtungen repräsentiert der OVSF-Code-Baum weiter einen geografischen Bereich – das gilt insbesondere für das Anwendungsbeispiel, das in Abschnitt 3.1.3.4 beschrieben wird. Unabhängig von der zugrundeliegenden Semantik sind die kleinsten Einheiten der Segmentierung stets die Blattknoten, während die inneren Knoten mehrere Kindelemente und damit Segmente zusammenfassen. Jeder Knoten repräsentiert dabei einen Ort, an dem Elemente eingefügt werden können

### 3.1.3.2 OVSF-kodierte Bloom-Filter

In traditionellen Bloom-Filtern wird eine vorgegebene Anzahl an Hash-Funktionen zum Einfügen und Testen von Elementen verwendet. Wenn sowohl Elemente als auch Strukturinformationen in den Filter kodiert werden sollen, stellt das eine starke Einschränkung dar. Mit OVSF-kodierten Bloom-Filtern wird deshalb ein Ansatz vorgestellt, mit dem sowohl das Interesse an Elementen als auch zusätzliche Ortsinformationen gleichzeitig in einen Filter kodiert werden können. Dazu werden die Ortsinformationen von einer binären Baumstruktur von räumlichen Indizes repräsentiert und ein OVSF-Baum derselben Größe erzeugt.

Grundsätzlich bestimmt ein OVSF-Code die Menge von Hash-Funktionen, die zum Einfügen eines Elements an einem gegebenen Ort verwendet werden. Daher erfordert die Einfügeoperation eines OVSF-kodierten Bloom-Filters neben dem Element selbst auch einen OVSF-Code, der den Einfügeort repräsentiert.

Für die folgende Beschreibung des Algorithmus sei ein finiter OVSF-Baum angenommen. Ein *OVSF Steuerungs-Code* ist dann ein OVSF-Code aus diesem Baum, der von links mit Nullen aufgefüllt wird (*padding*), sodass der Steuerungs-Code ein Array $c_l$ fester Länge $k$ ist und jede Stelle dieses Arrays eine Ziffer von $\{0, 1, -1\}$ ist. Sodann werden $2k$ voneinander unabhängige Hash-Funktionen $H = \{h_1^+, ..., h_k^+\} \cup \{h_1^-, ..., h_k^-\}$ gewählt, die das Universum der Elemente auf das Ganzzahlenintervall $\{1, ..., m\}$ und damit auf das Bit-Array des Filters gleichverteilt abbilden.

Ein Element $e$ an einem Ort einzufügen, der durch einen *OVSF Steuerungs-Code* gegeben ist, bedeutet, über den Steuerungs-Code zu iterieren: Für eine 0 an Position $i$ des Codes passiert nichts. Eine 1 an Position $i$ sorgt dafür, dass die Hash-Funktion $h_i^+$ verwendet wird; eine $-1$ bewirkt dementsprechend, dass Hash-Funktion $h_i^-$ zum Einsatz kommt. Das Ergebnis der so gewählten Hash-Funktion wird dann wie gewohnt als Index in dem Bit-Array des Bloom-Filters angesehen: das korrespondierende Bit wird auf 1 gesetzt.

Nach diesem Algorithmus führt ein kurzer OVSF-Code dazu, dass weniger Hash-Funktionen aktiviert, also auf ein Element angewendet werden. Längere Codes aktivieren zuerst dieselben Hash-Funktionen wie die Codes ihrer Elternknoten im OVSF-Baum aber auch noch einige zusätzliche. Dadurch erlaubt dieser Algorithmus die Generalisierung in Bezug auf den Einfügeort der Elemente: Beim Einfügen eines Elements an einem Ort eingefügt, der von einem OVSF-Code repräsentiert wird, werden auch alle Hash-Funktionen übergeordneter Steuerungs-Codes verwendet. Ein Test dieses Elements fällt somit für alle übergeordneten Orte positiv aus. Mit anderen Worten führt das Einfügen eines Element in einer tiefen Ebene im OVSF-Baum dazu, dass das Element zunächst an allen übergeordneten Ebenen eingefügt wird, weil die Hash-Funktionen aller übergeordneten Ebenen zum Einsatz kommen. Erst die zusätzlichen Hash-Werte, die für die Einfügeebene und den verwendeten Code charakteristisch sind, erzeugen ein Muster im Filter, das sich von dem Code eines Geschwisterknotens unterscheidet. An dieser Stelle zeigt sich, dass OVSF-Codes für das hier vorgeschlagene Verfahren anderen Binärbäumen überlegen sind. Es sind zwar andere Binärbäume zur Repräsentation der Hierarchie einsetzbar, die oben vorgestellten Eigenschaften von OVSF-Codes – Orthogonalität und Generalisierung – lassen sie hier aber als besonders geeignet erscheinen.

Falschpositive Filterantworten sollten generell lokal im OVSF-Baum auftreten: Je näher sich zwei Codes im OVSF-Baum sind, desto länger ist das gemeinsame Präfix. Zwei Elemente, die mit Codes eingefügt werden, die einander im Baum nahe sind, werden also mit vielen gleichen und nur wenigen unterschiedlichen Hash-Funktionen eingefügt. Dementsprechend ist auch die Wahrscheinlichkeit höher, dass ein Element entlang dieses gemeinsamen Asts vermutet wird – zumindest im Vergleich zu weit entfernten Einfügeorten, die schließlich mehr unterschiedliche Hash-Funktionen aktivieren.

Doch auch zwischen Geschwisterknoten sollte die Falschpositivrate begrenzt sein. Während zwei Knoten derselben Länge zwar ein langes gemeinsames

Präfix haben, unterscheiden sich die Codes nach diesem Präfix komplett – eine zentrale Eigenschaft der rekursiven Konstruktion von OVSF-Codes, die an dieser Stelle ausgenutzt wird. Damit aktivieren auch Geschwisterknoten stets disjunkte Mengen von Hash-Funktionen und reduzieren so bestmöglich die Falschpositivrate zwischen Geschwisterknoten.

Der Nachteil des beschriebenen Verfahrens liegt in der Anzahl der Hash-Funktionen, die von der Höhe des OVSF-Baums abhängt. Mit zunehmender Höhe des Baums werden exponentiell mehr Hash-Funktionen notwendig.

Im Folgenden wird der beschriebene Ansatz hinsichtlich der Lokalität der Falschpositiven Filterantworten evaluiert. Dazu wird der Ansatz mit traditionellen Bloom-Filtern verglichen. Für den Vergleich wird als Verknüpfung von Element und Ort die Konkatenation des entsprechenden Codes mit dem Element und einer festen Anzahl an Hash-Funktionen eingefügt. Die Filterparameter werden so gewählt, dass dieselbe Anzahl an Hash-Funktionen verwendet wird.

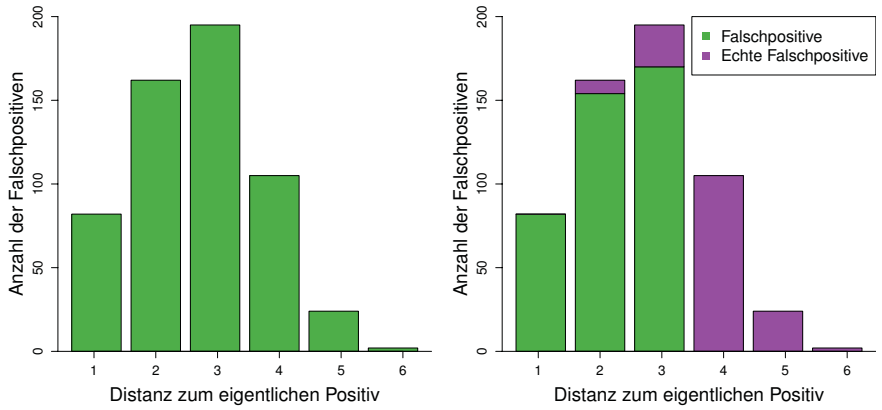
#### 3.1.3.3 Evaluation OVSF-kodierter Bloom-Filter

Zur Evaluation des beschriebenen Ansatzes wurde ein Element $u \in U$ mit einem OVSF-Code $c_l \in C$ in einen Bloom-Filter eingefügt und dieser Filter dann mit jedem Element $u' \in U$ an jedem möglichen Ort $c'_l \in C$ angefragt. Für jede falschpositive Filterantwort wurde der kürzeste Weg durch den OVSF-Baum identifiziert, der den Einfügeort $c_l$ mit dem Ort der falschpositiven Filterantwort $c'_l$ verbindet. Dieser kürzeste Pfad wird als die *Falschpositivdistanz* $d_{fp}$ bezeichnet.

Für jeden falschpositiven Test wurde zudem analysiert, ob der Ort $c'_l$ dem Einfügeort $c_l$ im OVSF-Baum übergeordnet ist. In diesem Fall ist die falschpositive Antwort erwartet und erwünscht, da das der Effekt der Generalisierung der OVSF-Codes ist. War $c_l$ kein direkter oder indirekter Kindknoten von $c'_l$, wurde die Distanz zwischen diesen beiden Knoten als $d_{fp_t}$ gemessen; diese Distanz wird im weiteren Verlauf als *echte Falschpositivdistanz* bezeichnet.

Dieses Vorgehen wurde für alle $u \in U$ durchgeführt, die nacheinander an allen Orten $c_l \in C$ eingefügt wurden. Getestet wurde stets mit allen Kombinationen von Elementen und Orten. Alle Experimente wurden in GNU Octave durchgeführt.

Zu Beginn wurde ein Universum $U$ definiert, das 7 unterschiedliche Elemente umfasst. Dazu wurde ein OVSF-Baum $C$ der Höhe 4 erzeugt, der dementsprechend über 15 Knoten und damit $c_l = 15$ mögliche Einfügeorte verfügt. Da ein OVSF-Code $c_l$ der Ebene $l = 4$ 8 Bit lang ist, enthält die Menge der Hash-Funktionen 8 voneinander unabhängige Funktionen $h_i$ mit $i = 1..8$. Jede $h_i$ verwendet den bekannten md5 Algorithmus mit unterschiedlichen *seeds*. Da das Einfügen an einem Blattknoten des oben beschriebenen Baums 8 Hash-Funktionen benötigt und dementsprechend maximal 8 Bit gesetzt werden, wurde ein Bloom-Filter der Länge $m = 16$ gewählt – während der Experimente enthält der Filter schließlich nur ein Element.




(a) Falschpositivdistanzen $d_{fp}$ (b) Distanzen echter Falschpositive $d_{fp_t}$

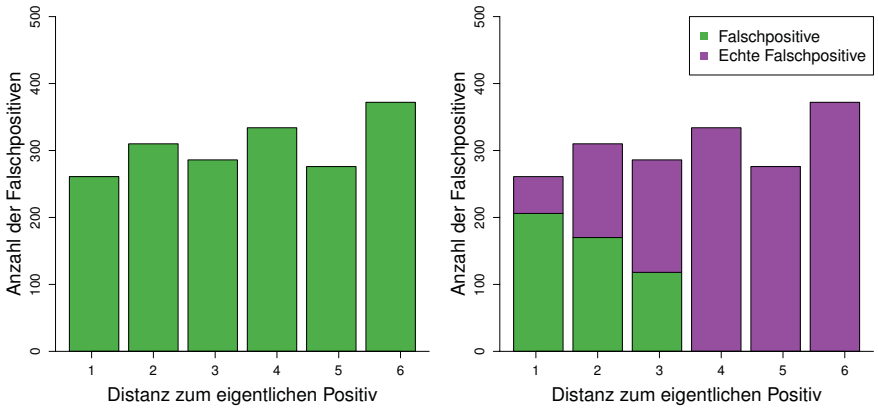
Abbildung 3.6: Histogramme der Distanzen von Falschpositiven und echten Falschpositiven mit der Größe eines Bloom-Filters von $m = 16$, $|U| = 7$ und $|C| = 15$

Die Abbildung 3.6(a) zeigt das resultierende Histogramm der Falschpositivdistanzen des oben beschriebenen Experiments. Die pyramidenartige Form ist bereits ein vielversprechendes Resultat, da die hierarchische Struktur des OVSF-Baum offenbar erfolgreich auf den eindimensionalen Bloom-Filter abgebildet werden konnte.

Bei der Distanz $d = 3$ hat das Histogramm ein deutlich erkennbares Maximum. Diese Distanz entspricht der Länge des kürzesten Pfades von einem Blatt- zum Wurzelknoten in einem OVSF-Baum der Höhe 4. Jede Distanz $d > 3$ bedeutet also, dass der Pfad zwischen Fundort und Einfügeort die Wurzel passiert hat und das Element fälschlicherweise in dem anderen Teilbaum gefunden wurde. Für Falschpositivdistanzen von $d = 2$ oder $d = 3$ muss das nicht unbedingt der Fall sein.

Abbildung 3.6(b) zeigt daher den Anteil der echten falschpositiven Filterantworten, wenn also der Filter eine positive Antwort liefert, obwohl das Element an einem Geschwisterknoten eingefügt wurde. Dieser Effekt wurde vor allem für die Distanzen $d > 3$ erwartet, kann aber in einigen Fällen auch für Distanzen von $d = 2$ und $d = 3$ beobachtet werden. Für $d = 2$ erklärt sich die geringe Zahl der echten Falschpositivmeldungen womöglich durch die Orthogonalität der OVSF-Codes derselben Ebene, da $d = 2$ Schritte stets zu einem direkten Geschwisterknoten führen.

In weiteren Experimenten wurde der beschriebene Ansatz mit traditionellen Einfügestrategien verglichen. Dabei bestand die Herausforderung vor allem darin, eine Vergleichbarkeit zu traditionellen Strategien herzustellen, da diese keine implizite Verknüpfung von dem Interesse an einer spezifischen Information und dem Ort dieser gewünschten Information erlauben. Außerdem werden in traditionellen Strategien stets $k$ Hash-Funktionen ausgewertet.




(a) Falschpositivdistanzen $d_{fp}$ (b) Distanzen echter Falschpositive $d_{fp_t}$

Abbildung 3.7: Histogramme der Distanzen von Falschpositiven und echten Falschpositiven und traditioneller Einfügestrategie mit der Größe eines Bloom-Filters von $m = 16$, $|U| = 7$ und $|C| = 15$

Um dennoch eine Verknüpfung zwischen Interesse und Ort zu simulieren, wurde die Ortsbeschreibung und das Einfügeelement konkateniert. Für die Anzahl $k$ der notwendigen Hash-Funktionen wurde die durchschnittliche Anzahl der Hash-Operationen bei OVSF-kodierten Einfügeoperationen verwendet. In dem beschriebenen Experiment mit einem OVSF-Baum der Höhe 4, in dem Blattknoten 8 und der Wurzelknoten 1 Bit lang sind, waren durchschnittlich 5.66667 Hash-Operationen notwendig. Für die folgenden Untersuchungen wurde deshalb $k = 5$ für traditionelle Einfügestrategien gesetzt.

Abbildung 3.7(a) zeigt das Histogramm der Falschpositivdistanzen $d_{fp}$ für eine konventionelle Einfügestrategie. Im Vergleich zu Abbildung 3.6(a) fällt zunächst die hohe Anzahl falschpositiver Filterantworten auf. Außerdem zeigt das Histogramm keine erkennbaren eindeutigen Tendenzen hinsichtlich der Baumdistanzen; vielmehr ist eine Schwankung der Distanzen erkennbar, die nahezu zufälligen Charakter hat. Dieser Eindruck wird etwas revidiert, wenn Abbildung 3.7(b) hinzugezogen wird, die die echten Falschpositivdistanzen $d_{fp_t}$ enthält. Hier zeigen wiederum alle Distanzen für $d > 3$ nur echte Falschpositive, wie erwartet.

Bemerkenswert ist bei dieser Einfügestrategie, dass die Falschpositiven, die bei Distanzen $d \leq 3$ auftraten, offenbar in Richtung $d = 3$ hin abnehmen. Das ist insofern eine wünschenswerte Eigenschaft, als dass sich Falschpositive nahe dem tatsächlichen Einfügeort häufen sollen. Insbesondere im Vergleich zu Abbildung 3.6(b), wo die Falschpositiven in Richtung Wurzelknoten gehäuft auftreten, kann das als Vorteil gegenüber dem vorgeschlagenen Verfahren angesehen werden. Allerdings verblasst dieser Vorteil deutlich, wenn die tatsächliche Falschpositivrate betrachtet wird.

| | OVSF-Kodierte | traditionelle Filter |
|---|---|---|
| Anfragen | 11025 | 11025 |
| Falschpositive | 576 | 1959 |
| Falschpositivrate | 5.224490% | 17.768707% |
| Echte Falschpositive | 170 | 1465 |
| Rate echter Falschpositive | 1.541950% | 13.287982% |
| Echte Falschpositivdistanzen | 3.782353 | 3.9501 |
| Anzahl berechneter Hash-Werte | 595 | 540 |
| Hash-Operationen pro Knoten | 5.666667 | 5.142857143 |

Tabelle 3.2: Zusammenfassung der Distanzhistogramme für Bloom-Filter mit $m = 16$, $|U| = 7$ und $|C| = 15$

Tabelle 3.2 enthält eine Zusammenfassung der vorgestellten Ergebnisse. Die Anfragen an die Filter entsprechen der Gesamtzahl der Anfragen an den Bloom-Filter während des Experiments. Da jedes der $|U| = 7$ Elemente an $|C| = 15$ verschiedenen Orten eingefügt wurde und so einen eigenen Bloom-Filter erzeugt hat, der wiederum mit jeder Kombination von Elementen und Orten angefragt wurde, umfasst ein Experiment $(7 * 15) * (7 * 15) = 11025$ Anfragen an den Bloom-Filter.

Der vorgestellte Ansatz des OVSF-kodierten Einfügens und Testens produzierte 576 falschpositive Filterantworten, wovon 170 echte Falschpositive waren, in denen das Element nirgends entlang des Anfrageasts eingefügt worden war. Das entspricht einer Falschpositivrate von $5,2245\%$ im Vergleich zu $17,7687\%$ der traditionellen Strategie. Werden nur die echten Falschpositiven berücksichtigt, hat der OVSF-kodierte Ansatz eine Falschpositivrate von $1,542\%$ im Vergleich zu $13,288\%$. Im vorgestellten Ansatz waren $29,51\%$ der falschpositiven Filterantworten echte Falschpositive im Vergleich zu $74,78\%$. Schließlich konnte die durchschnittliche Distanz zu einem echten falschpositiven Ergebnis auf $3,78$ Schritte durch den OVSF-Baum reduziert werden.

Eine Erklärung für die hohe Falschpositivrate des traditionellen Verfahrens mag in der Konkatenation der Elemente und Orte liegen. Diese entspricht dem karthesischen Produkt der Menge der Elemente und der Menge der Orte, wodurch die Größe des Universums drastisch zunimmt. Um die Falschpositivraten des OVSF-kodierten Ansatzes zu erreichen, müssen also die Parameter der Bloom-Filter entsprechend angepasst werden – zum Beispiel, indem der Filter deutlich vergrößert wird.

Das zeigt, wie der vorgestellte Ansatz die benötigte Filtergröße reduzieren und gleichzeitig die Falschpositivrate verringern kann. Schließlich häufen sich die falschpositiven Filterantworten näher an den tatsächlichen Einfügeorten und seltener im falschen Zweig des Baums.

Im folgenden wird ein Anwendungsbeispiel beschrieben, in dem OVSF-kodierte Bloom-Filter in einem drahtlosen Sensornetzwerk zur Verwaltung von Sensorknoten eingesetzt werden.

#### 3.1.3.4 Anwendungsbeispiel: Verwaltung eines Sensornetzwerks

Der im vorherigen Abschnitt beschriebene Ansatz zu OVSF-kodierten Bloom-Filtern wird nun in einem prototypischen drahtlosen Sensornetzwerk, einem *wireless sensor network (WSN)*, evaluiert. Ein solches WSN wird im allgemeinen definiert als

> (...) *ein Rechnernetz von Sensorknoten, winzigen („Staubkorn“) bis relativ großen („Schuhkarton“) per Funk kommunizierenden Computern, die entweder in einem infrastruktur-basierten oder in einem sich selbst organisierenden Ad-hoc-Netz zusammenarbeiten, um ihre Umgebung mittels Sensoren abzufragen und die Information weiterzuleiten.*[1]

Die im folgenden eingesetzte Architektur stellt keine besonderen Anforderungen an die Art oder Kapazität der eingesetzten Computer. Vielmehr geht es um die Modellierung der Architektur eines Sensornetzwerks, die dadurch charakterisiert ist, dass mehrere Sensoren an einem Ort verfügbar aber doch voneinander getrennt sind. Jeder Sensor soll einzeln aktiviert werden können, was für bestimmte Szenarien interessant ist. In einem sogenannten *crowd-sourcing*-System beispielsweise, in dem die Sensorknoten moderne sensorbestückte Smartphones sind und Nutzer sich ihre Endgeräte für Messungen von bestimmten Umweltinformationen gegenseitig bereitstellen, könnte das Endgerät eines Nutzers nach einer Messung der Umgebungslautstärke gefragt werden, während ein anderer Nutzer von ebendiesem Endgerät eine Messung des Luftdrucks anfordert. Beide Anfragen können über das beschriebene Verfahren in einen einzelnen Bloom-Filter kodiert werden, der dann an einen lokalen Verwaltungsknoten, einen sogenannten *cluster head*, weitergeleitet würde. Dieser Verwaltungsknoten extrahiert dann aus dem Bloom-Filter, welche Sensoren bei welchen seiner nachfolgenden Knoten aktiviert werden sollen.

Ein weiteres denkbares Szenario ist ein Verkehrsüberwachungssystem, das an Verkehrsinfrastrukturknoten, sogenannten *roadside units (RSUs)*, ausgebracht wird. Das könnten zum Beispiel Ampeln sein. Diese RSUs könnten jedenfalls bestimmte Informationen von vorbeifahrenden Fahrzeugen, wie deren Durchschnittsgeschwindigkeit, erfragen. Die RSU würde die entsprechende Anfrage ebenfalls in einen einzelnen Bloom-Filter kodieren und diesen an die vorbeifahrenden Fahrzeuge verteilen. Ein Fahrzeug könnte die Messanfrage aus dem Bloom-Filter dekodieren und mit den entsprechenden Messdaten antworten.

Unabhängig von den spezifischen Details wird in beiden Szenarien an wichtigen Stellen eine hierarchische Struktur angenommen. In dem partizipativen Sensornetzwerk besteht eine Hierarchie in der Netzwerktopologie, da mehrere Knoten zu Gruppen zusammengefasst werden und gruppenweise von den genannten *cluster heads* verwaltet werden. Das Verkehrsüberwachungssystem würde eine hierarchische Struktur unter den Sensoren eines Fahrzeugs anneh-

[1] Wikipedia-Eintrag zum Stichwort „Sensornetz“ vom 12.04.2016

men. In beiden Fällen müssen alle Knoten außer den Blattknoten wissen, welche Informationen von welchen nachfolgenden Knoten geliefert werden können. Für Sensornetzwerke ist dies eine typische Annahme. So ist jeder Knoten selbst für die geeignete Weiterleitung von Anfragen zuständig, und die Komplexität wird nicht unnötig auf alle Knoten verteilt.

Im Folgenden soll ein generisches Szenario angenommen werden, in dem Sensoren in einer Baumstruktur organisiert sind. Damit ein Sensor Daten misst, muss dieser Knoten explizit aktiviert werden. In diesem Szenario sollen Bloom-Filter dazu eingesetzt werden, bestimmte Sensoren an bestimmten Orten in der Hierarchie aufzuwecken – damit würde die hier beschriebene generische Architektur beide genannten Szenarien unterstützen. Eine hohe Falschpositivrate des Bloom-Filters führt jedenfalls dazu, dass falsche Knoten aktiviert und Energie verschwendet wird. Damit ein spezifischer Sensor an einem bestimmten Ort aufgeweckt wird, wird der angefragte Messwert zusammen mit dem OVSF-Code des Orts in einen Bloom-Filter eingefügt und dieser Filter an den Wurzelknoten des Netzwerks übergeben. Der Wurzelknoten überprüft zunächst, ob an dem Wurzelknoten selbst Sensoren aktiviert werden sollen, um den Filter anschließend unverändert an diejenigen Kindknoten weiterzugeben, die selbst Sensoren aktivieren oder den Filter entsprechend weiterleiten sollen. Dieser Ablauf wiederholt sich für jeden nachfolgenden Kindknoten, der den Bloom-Filter erhält.

Evaluiert wird in diesem Szenario die Effizienz des Netzwerks in Form der verbrauchten Energie der aktivierten Sensoren. Im Vergleich zu traditionellen Einfügestrategien sollte der Energieverbrauch der oben beschriebenen OVSF-basierten Strategie wegen der geringeren erwarteten Falschpositivrate auch geringer sein. Falls Sensoren doch fälschlicherweise aufgeweckt werden, sollte er theoretisch näher am Anfrageort liegen, weil die echte Falschpositivdistanz $d_{fp_t}$ geringer sein sollte.

Zur Evaluation wurde ein solches generisches Sensornetzwerk implementiert, das zudem in der Lage ist, den erwarteten und den tatsächlichen Energieverbrauch jedes Sensors im Netzwerk auszugeben. Die folgenden Experimente laufen stets über mehrere Runden, in denen jeweils eine zufällige Anzahl Sensoren an zufällig ausgewählten Orten aktiviert werden. Diese Anfragen werden wie beschrieben in Bloom-Filter eingefügt – einmal auf traditionelle Weise und einmal entsprechend der OVSF-basierten Strategie. Um den Bloom-Filter nicht zu überfüllen, ist die Anzahl der gleichzeitig aufzuweckenden Sensoren pro Runde begrenzt. Die tatsächliche Anzahl liegt stets zwischen 0 und diesem oberen Grenzwert. Für jede Runde werden die ursprünglich angefragten und die tatsächlich aktivierten Sensoren gespeichert, um den erwarteten und tatsächlichen Energieverbrauch angeben zu können.

#### 3.1.3.5 Evaluation des Anwendungsbeispiels

Die im vorherigen Abschnitt beschriebene Evaluation des Konzepts stützte sich auf ein simuliertes Sensornetzwerk mit 15 unterschiedlichen Orten, an

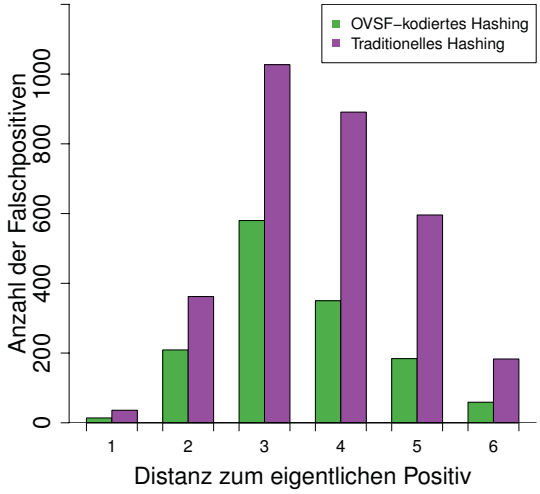



Abbildung 3.8: Histogramm der durchschnittlichen Distanzen von Falschpositiven

denen jeweils bis zu 4 Sensoren verfügbar waren. Insgesamt wurden 32 Sensoren zufällig über das Netzwerk verteilt – an jedem Ort waren somit durchschnittlich 2, 13 Sensoren ausgebracht. Pro aktiver Runde verbrauchte ein Sensor 1 Einheit Energie, während jeder Sensor für 3 Runden aktiviert blieb. Pro Runde durften maximal 5 Sensoren gleichzeitig aktiviert werden. Zum Einsatz kamen stets Bloom-Filter mit $m = 16$ Bits. Die Simulationen dauerten 150 Runden, in denen insgesamt 6750 Anfragen an Bloom-Filter gestellt wurden.

Der OVSF-basierte Ansatz ist der traditionellen Einfügestrategie hinsichtlich der Falschpositivrate und damit dem überflüssigen Energieverbrauch deutlich überlegen. In beiden Ansätzen sind zunächst hohe absolute Falschpositivraten zu verzeichnen – 22.34% für OVSF-kodierte und 53.72% für traditionelle Filter. Diese haben auch in beiden Fällen zu einem hohen überflüssigen Energieverbrauch geführt: Faktor 7.512428 mehr als das Optimum für OVSF-kodierte und Faktor 13.7726 für traditionelle Filter. Dennoch wurde mit der OVSF-basierten Strategie im Vergleich 65% weniger Energie verschwendet.

Abbildung 3.8 zeigt ein Histogramm der Anzahl der Falschpositiven Aktivierungen in Verbindung mit der durchschnittlichen Falschpositivdistanz für beide Hashing-Strategien. Während das traditionelle Verfahren insgesamt deutlich mehr Falschpositive erzeugt, zeigen beide Verfahren ein deutliches Maximum für $d = 3$. Das scheint die Erkenntnisse aus Abbildung 3.7(b) insbesondere für das OVSF-basierte Verfahren zu falsifizieren. Da aber in diesem Fall stets mehrere Elemente gleichzeitig in den Bloom-Filter eingefügt wurden, konnten die Falschpositivdistanzen nicht für einzelne Elemente gemessen werden. Insbesondere konnten echte und herkömmliche Falschpositive nicht voneinander unterschieden werden. Im Gegensatz zu den Abbildungen 3.6(b) und 3.7(b) zeigt die Grafik 3.8 deshalb auch nur die durchschnittlichen Distanzen. Es kann an diesem Histogramm deshalb vor allem abgelesen werden, dass die OVSF-basierte Strategie insgesamt weniger Falschpositive erzeugt.

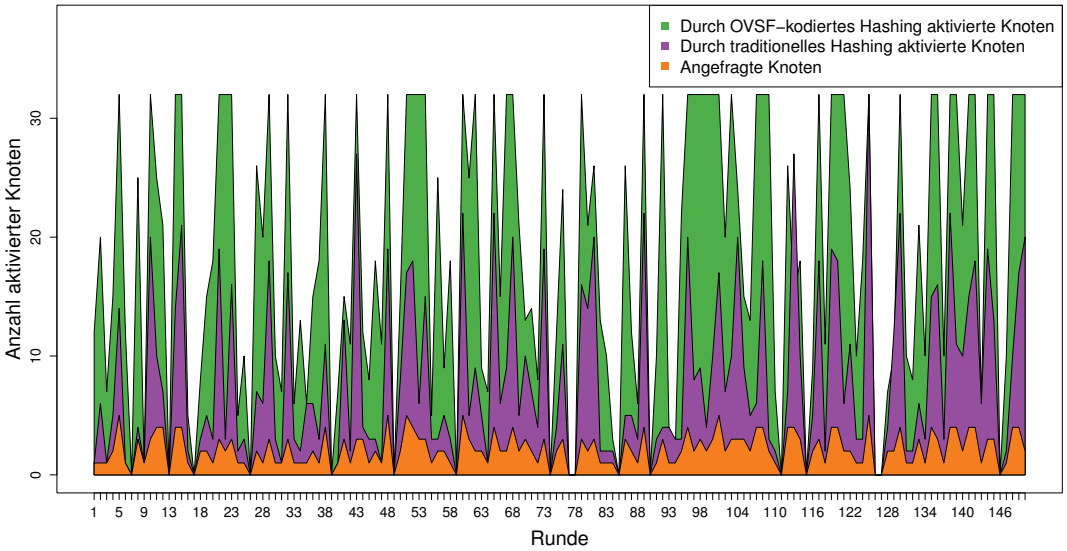


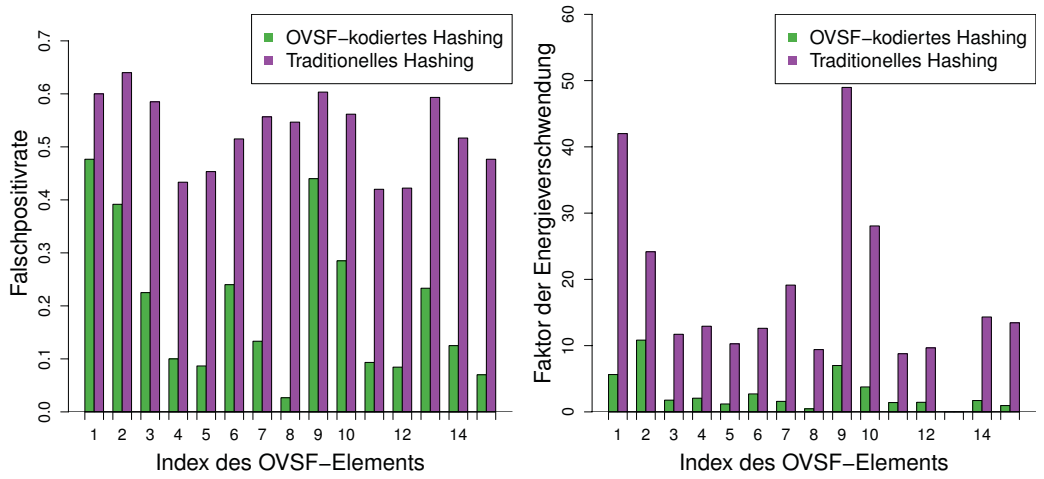

Abbildung 3.9: Anzahl der angefragten Sensoren im Vergleich zur Anzahl tatsächlich aktivierter Sensoren

Abbildung 3.9, in der die Anzahl der angefragten und tatsächlich aktivierten Sensorknoten pro Runde abgetragen ist, bestätigt den Vorteil der OVSF-basierten Strategie. Beide Strategien aktivieren zwar stets mehr Sensoren als nötig, das hierarchische Verfahren scheint aber insgesamt besser an die tatsächlichen Anfragen zu adaptieren. Das traditionelle Verfahren sorgt sogar dafür, dass mehrmals alle der 32 Sensoren aktiviert werden, wodurch der Graph aussieht, als sei er bei dem Wert abgeschnitten.

Die unnötigen Aktivierungen sind das Resultat einer hohen Falschpositivrate. Deshalb wurde die Falschpositivrate für jeden Knoten im OVSF-Baum separat untersucht. Das Ergebnis ist in Abbildung 3.10a erkennbar. Das Histogramm stellt die Falschpositivrate der einzelnen Orte dar. In der Abbildung ist der Wurzelknoten bei $x = 1$ und seine direkten Kinder bei $x = 2$ und $x = 9$ abgetragen. Die Kinder von $x = 2$ liegen wiederum bei $x = 3$ und $x = 6$, die Kinder von $x = 9$ bei $x = 10$ und $x = 13$. Mit dieser Zuordnung zur Baumstruktur ist erkennbar, dass Falschpositive beim OVSF-kodierten Hashing-Verfahren in Richtung der Blätter des Baumes zunehmend seltener auftreten. Dieses Verhalten entspricht den Beobachtungen zu Abbildung 3.6(a) und 3.6(b). Gleichzeitig ist für das traditionelle Hashing-Verfahren kein Zusammenhang zwischen dem Auftreten von Falschpositiven und dem Ort des Auftretens im OVSF-Baum erkennbar.

Abbildung 3.10(b) zeigt die relative Energiebilanz jedes OVSF-Ortes. Ein Faktor $> 1$ bedeutet, dass Energie verschwendet wurde. Die Lücke bei $x = 13$ im Histogramm ist korrekt, da an diesem Ort kein Sensor ausgebracht war.

Dass das OVSF-kodierte Verfahren weniger Energie verschwendet als das tradtionelle, ist im Anbetracht der Falschpositivrate und dem Faktor der Abweichung des Energieverbrauchs wenig überraschend. Bemerkenswert ist an




(a) Falschpositivrate

(b) Energieverbrauch im Verhältnis zum jeweiligen Optimum

Abbildung 3.10: Falschpositivrate und Energievebrauch aller Orte

dieser Stelle der Abweichungsfaktor des Wurzelknotens $x = 1$. Aus Abbildung 3.10(a) hätte man schließen können, dass der Energieverbrauch an diesem Ort im Baum besonders hoch ist, da die Abbildung für $x = 1$ die höchste Falschpositivrate verzeichnet. Zudem verarbeitet der Wurzelknoten jeden eingehenden Bloom-Filter und zur Aktivierung eines Sensors an dieser Stelle ist nur ein Bit notwendig – schließlich ist der Wurzelknoten des OVSF-Baums genau ein Bit lang. Erkennbar ist hier, dass vor allem die direkten Kinder des Wurzelknotens $x = 2$ und $x = 9$ von hohem unnötigen Energieverbrauch betroffen sind. Das führt zu dem Schluss, dass besonders die Sensoren, die auf der zweiten Ebene des OVSF-Baums ausgebracht werden, unnötig oft aktiviert werden, während diese Sensitivität für tiefere Ebenen abnimmt.

Mit dem hier beschriebenen Verfahren wurde ein neuartiges Hashing-Verfahren vorgestellt, mit dem hierarchisch strukturierte Daten in ein eindimensionales Bit-Array kodiert werden können. Zur umfassenden Evaluation des Verfahrens wurde ein OVSF-Code-Baum zur Strukturierung der Daten und ein Bloom-Filter als Bit-Array eingesetzt. In Bezug auf die Baumstruktur konnte die Falschpositivrate des Bloom-Filters durch das OVSF-kodierte Hashing-Verfahren erkennbar reduziert werden. Eine Beispielanwendung in einem drahtlosen Sensornetzwerk hat zudem gezeigt, wie das Verfahren eingesetzt werden kann, um spezifische Sensoren an spezifischen Orten aufzuwecken.

Der folgende Abschnitt beleuchtet nun, warum Bloom-Filter auch für die Beschreibung von Kontexten besonders geeignet sind.

## 3.2 Bloom-Filter zur Beschreibung von Kontexten

Ein Kontext im Sinne einer Zustandsbeschreibung mit Kontextinformationen besteht in der Regel aus einer Menge heterogener Elemente unterschiedlicher Datentypen, Größen und Semantiken. Alleine der Ort einer Person kann durch zahlreiche unterschiedliche Kontextinformationen beschrieben werden: neben einer Straßenadresse bieten sich auch GPS-Koordinaten, Identifikatoren von Mobilfunkzellen und die Hardware-Adressen verfügbarer WLAN-Zugangspunkte der Umgebung an. Alleine diese Ortsbeschreibungen haben schon alle unterschiedliche Datentypen. Und jeder dieser Datentypen ist spezifischen Semantiken und Modalitäten unterworfen: WGS84-Koordinaten aus dem GPS-System müssen anders miteinander verglichen werden als etwa Straßenadressen und Hausnummern. Darüber hinaus genügt der Ort selten, um die Situation einer Person wirklich umfassend zu beschreiben. Ebenso hilfreich wären schließlich aktuelle Kalendereinträge, Personen in der Umgebung, die körperliche Aktivität der Person – eben viele weitere Kontextinformationen.

Kapitel 2 und insbesondere Abschnitt 2.2 haben sich diesen Schwierigkeiten im Umgang mit Kontextinformationen bereits ausführlich gewidmet. Deutlich wurde, dass sich bestehende Anwendungen meist auf eine einzelne Modalität konzentrieren, weil Kontextinformationen modalitätsspezifisch verarbeitet werden müssen. Daher können aktuelle Anwendungen nur um weitere Modalitäten erweitert werden, wenn Verarbeitungs- und Vergleichsoperationen ebenfalls angepasst werden.

Eine kontextzentrische Vernetzung, die ein Thema dieser Arbeit darstellt und Routing auf der Basis ähnlicher Kontexte realisiert, ist unter diesen Gesichtspunkten nur eingeschränkt zu realisieren. Nicht zuletzt, weil die Kontextinformationen, die für eine Vernetzung verwendet werden sollen, im Vorhinein bekannt sein müssten; schließlich müssten die Algorithmen zum modalitätsspezifischen Vergleich von Kontextinformationen für jede Modalität einzeln implementiert werden. Zudem müssten alle verwendeten Verfahren in jedem Knoten, der kontextabhängige Routingentscheidungen trifft, bekannt sein. Ansonsten könnte die Vernetzung kaum sinnvoll erreicht werden. Für ein ernstzunehmendes pervasives System sind das kaum überwindbare Hürden.

Die größte Schwierigkeit in diesem Aufbau liegt in der Heterogenität von Kontextinformationen. Genau diese Schwierigkeit kann mit Hilfe von Bloom-Filtern beseitig werden: Bloom-Filter sind Bit-Arrays und beschreiben Mengen unabhängig von Datentypen, Größe der Elemente oder sonstigen sensor- oder quellenabhängigen Eigenschaften. Eine Kernidee auf dem Weg zu echtem kontextabhängigen Routing besteht deshalb darin, einen Kontext mit einem Bloom-Filter zu beschreiben. Bloom-Filter bilden also gewissermaßen einen Fingerabdruck eines Kontexts.

Diese Idee bringt viele Vorteile mit sich: Zum einen werden damit die spezifischen Details einzelner Bestandteile der Zustandsbeschreibung implizit trans-

parent. Der Kontext wird nun vollkommen typlos beschrieben. Des weiteren schützt die Kontextbeschreibung durch einen Bloom-Filter die Privatsphäre der Nutzer. Kryptographische Hash-Funktionen verhindern wirkungsvoll, dass aus Hash-Werten Rückschlüsse auf zugrundeliegende Kontextinformationen gezogen werden können. Und schließlich können Kontextbeschreibungen in Form von Bloom-Filtern auch erstmals umfassend und modalitätenunabhängig miteinander verglichen werden, wie der folgende Abschnitt 3.3 zeigen wird. Es muss also weder eingeschränkt werden, welche Kontextinformationen zum Routing verwendet werden sollen, noch muss eine solche Menge über die gesamte Lebensdauer eines Netzwerks feststehen. Es ist nicht einmal notwendig, dass alle Knoten im Netzwerk dieselben Kontextinformationen verwenden.

Insgesamt ist damit erstmals eine echte kontextzentrische Vernetzung sinnvoll möglich, weil Kontexte bei jedem Knoten, der Routingentscheidungen treffen muss, generisch, modalitätenunabhängig und privatsphäreschonend miteinander verglichen werden können.

Ein Nachteil der Kontextbeschreibung durch Bloom-Filter besteht allerdings darin, dass manche modalitätsspezifische Beschreibungen von Ähnlichkeit verloren gehen: Anhand ihrer Hash-Werte kann beispielsweise keine räumliche Nähe zwischen zwei Paaren von WGS84-Koordinaten mehr festgestellt werden. Um diesem Problem zu begegnen, können solche Messwerte zum einen diskretisiert werden. Für GPS-Koordinaten bietet sich eine Z-kurvenbasierte Kodierung an, wie es etwa im weit verbreiteten *Geohash* Anwendung findet [231]. Damit können Bereiche auf der Erde als Strings fester Länge repräsentiert werden. Die Größe der Bereiche ist definierbar und könnte je nach Anwendungsfall variiert werden. Denkbar ist auch, unterschiedliche Granularitäten gleichzeitig in eine Kontextbeschreibung aufzunehmen.

Zum anderen wird dieser Nachteil auch dadurch ausgeglichen, dass Kontexte jetzt wesentlich umfassender beschrieben und miteinander verglichen werden können als nur auf der Basis einzelner Modalitäten. Geo-Koordinaten einer Ortsbeschreibung bilden beispielsweise nur ein spezifisches Detail eines Kontextes. Ähnliche Kontexte können auch an zwei unterschiedlichen Orten auftreten. Und wird der Kontext beispielsweise über aktuell laufende Musik, die U-Bahn als aktuelles Transportmittel und das Ziel der aktuellen Fahrt beschrieben, ist der genaue Aufenthaltsort sogar eine eher unwichtige Information. Obwohl also möglicherweise modalitätsspezifische Ähnlichkeitsinformationen durch die Binärdarstellung verloren gehen, könnten Kontexte trotzdem eine gewisse Ähnlichkeit behalten, weil andere Details in den Beschreibungen übereinstimmen.

Über die Tatsache hinaus, dass Bloom-Filter allgemein dazu geeignet sind, Kontexte zu beschreiben, bieten einige der weiter oben beschriebenen Varianten von Bloom-Filtern spezielle, nützliche Eigenschaften. So wurden zum Beispiel *counting* Bloom-Filter als eine Protokollerweiterung vorgestellt. Diese verwalten einen Zähler zu jedem gesetzten Bit und erlauben so das Löschen von Elementen aus dem Filter. Damit bietet sich diese Erweiterung an, wenn Kontexte mit sehr vielen oder stark schwankenden Informationen beschrie-

ben werden. Ortsbeschreibungen in Gebäuden etwa, die verfügbare WLAN-Zugangspunkte der Umgebung einbeziehen, sind solchen Schwankungen unterworfen. Bewegt sich eine Person flink durch ein Gebäude und sieht unterwegs viele verschiedene Zugangspunkte, kann es sich lohnen, Elemente aus dem Filter zu löschen, anstelle den Filter bei jeder Änderung komplett neu aufzubauen.

Vorgestellt wurden auch spektrale Filter für Multimengen, in denen gleiche Elemente mehrmals enthalten sein können. Auch diese Erweiterung kann für Kontextbeschreibungen ausgenutzt werden, wenn etwa modelliert werden soll, mit welchen Kollegen in einer gewissen Zeitspanne am häufigsten kommuniziert wurde.

Ebenso interessant sind *time-decaying* Bloom-Filter, aus denen Elemente nach einer definierten Zeitspanne automatisch verschwinden. Solche Filter bieten sich an, wenn das Vergessen von Informationen modelliert werden soll – etwa das Verblassen von Freundschaften, zu denen sehr lange kein Kontakt bestand. Diese Erweiterung bietet sich aber auch an, um gewisse Informationen über ihre Gültigkeit hinaus zu behalten. Das kann interessant sein, wenn der Kontext einen zurückgelegten Weg enthalten soll. So können vergleichbare Kontextbeschreibungen für Personen konstruiert werden, die innerhalb der letzten 100 Kilometer dieselben Autobahnabschnitte passiert haben – vorausfahrende Personen behalten so ähnliche Ortsbeschreibungen bei, wie diejenigen, die eben auf die Autobahn aufgefahren sind. Für das Routing von Stauinformationen ist das sehr hilfreich.

Neben den Protokollerweiterungen wurden auch Varianten vorgestellt, die die Falschpositivrate bei Stichwortsuchen reduzieren – ein Beispiel war der im Rahmen dieser Arbeit entwickelte Gaußsche Bloom-Filter. Solche Varianten können natürlich eingesetzt werden, wenn gezielt nach bestimmten Kontextinformationen in fremden Filtern gesucht wird – etwa, weil Personen gezielt nach Kommilitonen aus demselben Kurs suchen.

Schließlich bieten auch die Erweiterungen vielversprechende Ansatzpunkte, die eine Ordnung auf den zugrundeliegenden Daten modellieren. Die OVSF-kodierten Filter etwa, die auch im Rahmen dieser Arbeit entwickelt wurden, gehen von einer hierarchischen Ordnung in den Daten aus. Eine solche Struktur ist auch unter Kontextinformationen zu finden, etwa bei Ortsbeschreibungen oder Kursplänen von Studiengängen. Der Ort einer Person etwa kann mit zunehmender Genauigkeit über das Land, die Stadt, den Stadteil und die Straßenadresse beschrieben werden. Oder anders ausgedrückt: mit dem Länder-Code der Mobilfunkzelle, dem Identifikator der Mobilfunkzelle, einer GPS-Region und den verfügbaren WLAN Zugangspunkten. Die ersten Ergebnisse zur Evaluation der OVSF-kodierten Filter zeigen, dass bestimmte Hierarchien im Bloom-Filter abgebildet werden können und sich falschpositive Filterantworten baumlokal häufen. Es sollte daher weiter untersucht werden, wie sich auch allgemeinere Strukturen unter Kontextinformationen in die Filter übertragen lassen.

Allgemein nehmen Bloom-Filter eine zentrale Rolle im Rahmen dieser Arbeit ein: Sie erlauben eine typlose, modalitätenunabhängige Beschreibung von Kontexten. Der Kontext eines Nutzers wird dabei als eine heterogene Menge von Kontextinformationen verstanden, in der Elemente aus unterschiedlichen Datenquellen, Messwerte aus verschiedensten Sensoren oder Erkenntnisse aus diversen Inferenzalgorithmen enthalten sein können. Alle diese Elemente, Messwerte oder Kategorien eignen sich entweder direkt als Eingabewerte für Hash-Funktionen oder lassen sich so zu Strings transformieren, dass sie in einen Bloom-Filter eingefügt werden können. Die Heterogenität der Kontextinformationen kann daher durch Bloom-Filter zu einer allgemeinen, generischen Kontextbeschreibung umformuliert werden, die dennoch die Vielfalt der Kontextausprägungen widerspiegelt. Gleichzeitig bleibt die Privatsphäre der Nutzer schützt, weil verwendete Kontextinformationen aus Bloom-Filtern nicht herausgelesen werden können.

Der folgende Abschnitt beschreibt nun einen wichtigen Aspekt kontextzentrischen Routings: den Vergleich von Kontexten über Ähnlichkeitsvergleiche zwischen Bloom-Filtern.

## 3.3 Mengenvergleiche mit Bloom-Filtern

Die vorherigen Abschnitte haben einige Varianten von Bloom-Filtern und ihre Einsatzmöglichkeiten insbesondere für die Beschreibung von Kontexten vorgestellt. Die kompakte, speichereffiziente und generische Repräsentation zugrundeliegender Mengen mit kontrollierbaren Falschpositivraten für Suchanfragen haben dazu geführt, dass diese Speicherstruktur bisher in zahlreichen Arbeiten zu verteilten Systemen und informationszentrischen Netzen Anwendung findet [95]–[97]. Eingesetzt werden sie häufig für die Beschreibung eines Zwischenspeichers, den Austausch über dessen Inhalt mit anderen Knoten und ein darauf aufbauendes inhaltsbasiertes Routing [98] (dies wird in Abschnitt 4.1.1 genauer beschrieben). Ihre geringe Größe verursacht wenig überflüssige Datenübertragung beim Austausch über den Zustand der verteilten Speicherknoten [99] und erlaubt dennoch verteilte Vergleiche der Speicherstände der Knoten. In diesen Arbeiten werden bisher hauptsächlich Suchanfragen mit Stichwörtern an Bloom-Filter gestellt, um gezielt nach bestimmten Elementen in fremden Zwischenspeichern zu suchen. Alternative Verfahren wie [100] setzen auf zusätzliche Hilfsstrukturen wie Merkle-Bäume oder Patricia-Tries.

Im Kern geht es jedoch stets um den Vergleich zweier Mengen. Anstelle auf paarweise Anfragen einzelner Elemente zu setzen, kann auch die Ähnlichkeit zwischen den Mengen quantifiziert werden. Zu diesem Zweck gibt es bereits weit verbreitete Maße, wie beispielsweise die Jaccard-Ähnlichkeit. Sie baut auf dem Jaccard-Index auf – dem Quotient aus Schnittmenge und Vereinigung zweier Mengen:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

Die Jaccard-Ähnlichkeit ist nicht zuletzt deshalb so beliebt, weil ihr Inverses eine Metrik beschreibt, die Jaccard-Distanz genannt wird:

$$d_J(A, B) = 1 - J(A, B).$$

Die Jaccard-Distanz $d_J$ nimmt Werte aus dem Interval $[0, 1]$ an, wobei 0 bedeutet, dass die Mengen identisch sind, und 1 zwei disjunkte Mengen anzeigt. Auf zwei Mengen von Strings kann diese Jaccard-Distanz beispielsweise leicht berechnet werden – besonders, wenn beide Mengen sortiert vorliegen, da der Vergleich der Mengen auf das Zählen gemeinsamer und unterschiedlicher Elemente hinausläuft.

Auch für diese Arbeit kann der Vergleich von Mengen interessant sein: Die Mengen, die in dieser Arbeit eine zentrale Rolle spielen, sind vor allem Kontextbeschreibungen, also Zusammenfassungen (heterogener) Kontextinformationen. Ein Vergleich zweier solcher Beschreibungen liefert also Informationen darüber, wie ähnlich sich zwei Kontexte sind.

In beiden Fällen ist das Berechnen der Jaccard-Distanz unter Umständen problematisch: Für Routingentscheidungen insbesondere in Netzwerkknoten mit geringen Rechenleistungen kann das Sortieren und Vergleichen von Mengen von Strings aufwändig und speicherintensiv sein. Der Vergleich von Kontextmengen wiederum kann stark typabhängig und deshalb auch nicht trivial zu berechnen sein.

Aufbauend auf Vorarbeiten von Swamidass und Baldi kann die Jaccard-Distanz über Bit-Arrays abgeschätzt werden [101]. Sie haben gezeigt, dass anhand eines Fingerabdrucks einer Menge, der als Bit-Array repräsentiert wird, die Anzahl der Elemente der zugrundeliegenden Menge approximiert werden kann. Solche Fingerabdrücke werden vor allem in großen Datenbanken chemischer Sequenzen eingesetzt. Die Bit-Arrays der Fingerabdrücke werden darin über Modulo-Operationen „gefaltet“ und komprimiert gespeichert. Gefaltete Sequenzen können sich allerdings überlappen und so einen systemischen Fehler einführen. Solange die Konfiguration der Operationen aber bekannt ist, kann dieser Fehler mathematisch korrigiert werden. Daraus wurde abgeleitet, dass auch die Anzahl der Elemente in einem Bloom-Filter approximiert werden kann, solange die Konfiguration der Filter hinsichtlich der Anzahl $k$ der verwendeten Hash-Funktionen bekannt ist [101].

Dazu wird aus der Größe $m$ des Bloom-Filters, dem Anteil $\oslash_{F_A}$ an Nullen in einem Bloom-Filter $F_A$ und der Anzahl $k$ der Hash-Funktionen die Anzahl der eingefügten Elemente $n_{F_A}$ folgendermaßen abgeschätzt:

$$n_{F_A} \approx -\frac{\log(\oslash_{F_A})m}{k} \tag{3.7}$$

Da für zwei Filter identischer Konfiguration die Vereinigung $F_A \cup F_B$ durch ein bitweises ODER berechnet werden kann, kann die Anzahl der Elemente dieser Vereinigung analog abgeschätzt werden:

$$n_{F_A \cup F_B} \approx -\frac{\log\left(\oslash_{F_A \cup F_B}\right) m}{k}. \tag{3.8}$$

Darin ist $\oslash_{F_A \cup F_B}$ der Anteil der Nullen der bitweisen ODER-Verknüpfung der Filter $F_A$ und $F_B$. Wegen

$$|A \cup B| = |A| + |B| - |A \cap B|$$

kann aus 3.7 und 3.8 auch die Größe der Schnittmenge der beiden Filter $F_A$ und $F_B$ approximiert werden:

$$n_{F_A \cap F_B} \approx n_{F_A} + n_{F_B} - n_{F_A \cup F_B}. \tag{3.9}$$

Damit kann mittels

$$d_J(F_A, F_B) = 1 - \frac{n_{F_A \cap F_B}}{n_{F_A \cup F_B}} \tag{3.10}$$

die Jaccard-Distanz zwischen zwei Bloom-Filtern berechnet werden. Insbesondere für das kontextzentrische Routing ist damit ein generischer, privatsphäreschonender Vergleich von Kontexten möglich. Der Vergleich ist deshalb generisch, weil er unabhängig von den Datentypen der Elemente der Mengen auf Bit-Arrays durchgeführt wird. Grundsätzlich ist über den Vergleich zweier Bloom-Filter ein Ähnlichkeitsvergleich von Mengen mit heterogenen Elementen möglich, ohne dass spezielle Kenntnisse über die Datentypen einzelner Elemente eingesetzt werden muss. Privatsphäreschonend ist der Vergleich deshalb, weil er prinzipiell auch von einer nicht-vertrauenswürdigen dritten Partei durchgeführt werden kann. Durch den Einsatz kryptgraphischer Hash-Funktionen ist sichergestellt, dass ein Bloom-Filter keine Rückschlüsse auf die eingefügten Elemente zulässt. Dieses Vorgehen wurde im Rahmen dieser Arbeit entwickelt und ist bereits in [241] beschrieben.

Weiterentwickelt wurde dieses Verfahren von Werner in [102]. Darin beschreibt der Autor, wie eine untere Schranke für die Jaccard-Ähnlichkeit bestimmt werden kann, die zwischen einer Menge von Anfrageelementen und einer Menge in einer Datenbank berechnet werden kann, obwohl zu der Menge aus der Datenbank nur die Beschreibung in Form eines Bloom-Filters und die Anzahl der enthaltenen unterschiedlichen Elemente gegeben sind. Zu einer Menge von Anfrageelementen $Q$, einer Menge in der Datenbank $S$, dem Bloom-Filter $F_S$ der Menge $S$ und der Anzahl $|S|$ der unterschiedlichen Elemente in $S$ kann diese untere Schranke $LB$ so bestimmt werden, dass gilt

$$LB(Q, F_S) \leq d_J(Q, S) = 1 - \frac{|Q \cap S|}{|Q \cup S|}.$$

Zunächst wird die Schnittmenge im Zähler $|Q \cap S|$ abgeschätzt, indem die Elemente gezählt werden, die in der Anfragemenge enthalten sind und für die der Filter eine positive Antwort liefert. Diese Zahl wird als $\varphi_1$ bezeichnet:

$$\varphi_1 = \#\{\text{Elemente aus } Q, \text{ die in } F_S \text{ enthalten sind}\} \geq |Q \cap S|.$$

Die Abschätzung über die größer-gleich-Relation ist deshalb gültig, weil die einzige Fehlerquelle hierbei die falschpositiven Filterantworten von $F_S$ sind. Würde ein Element, das in $Q$ enthalten ist, eine falschpositive Antwort von $F_S$ provozieren, würde es fälschlicherweise als Element aus $S$ und damit trotzdem als zur Schnittmenge $Q \cap S$ dazugehörig gezählt werden.

Analog dazu wird auch der Nenner $|Q \cup S|$ abgeschätzt. Dazu werden zu den Elementen, die in $S$ enthalten sind, diejenigen aus $Q$ gezählt, die nicht in dem Filter $F_S$ enthalten sind. Da auch hier die einzige Fehlerquelle die falschpositiven Filterantworten sind, für diese Abschätzung aber diejenigen Elemente gezählt werden, die nicht im Filter enthalten sind, wird der Nenner über $\varphi_2$ mit einer kleiner-gleich-Relation abgeschätzt:

$$\varphi_2 = |S| + \#\{\text{Elemente aus } Q, \text{ die nicht in } F_S \text{ enthalten sind}\} \leq |Q \cup S|.$$

Mit diesen beiden Abschätzungen kann die untere Schranke für die Mengenähnlichkeit zwischen $Q$ und $S$ mit gegebener Anfragemenge $Q$, dem Bloom-Filter $F_S$ zu $S$ und der Anzahl der Elemente in $S$ approximiert werden:

$$LB(Q, F_S) = 1 - \frac{\varphi_1}{\varphi_2} \leq d_J(Q, S) = 1 - \frac{|Q \cap S|}{|Q \cup S|}.$$

Über diese untere Schranke können einzelne Mengen im Suchraum für eine nähere Betrachtung ausgeschlossen werden (*pruning*). Das heißt, dass der Suchraum zwar noch linear durchlaufen werden muss, für jede Menge in der Datenbank aber entschieden werden kann, ob sie eine gewisse Ähnlichkeit zur Anfragemenge überschreitet. Darüber hinaus ist für manche Applikationen wünschenswert, wenn ganze Bereiche einer Datenbank für eine Suche ausgeschlossen werden könnten, wenn also mehrere Mengen von einem Bloom-Filter zusammenfassend beschrieben würden und die Jaccard-Ähnlichkeit zu dieser Zusammenfassung bestimmt werden könnte.

Ein solches distanzbasiertes *pruning* einer Anfragemenge $Q$ zu einer zusammenfassenden Repräsentation $S_{sum} = \{S_1, S_2, ..., S_n\}$ mehrerer Mengen $S_i$ der Datenbank würde eine Funktion voraussetzen, die häufig `mindist` genannt wird und für die gilt

$$d(Q, S_i) \leq mindist(Q, S_{sum}) \forall 1 \leq i \leq n. \tag{3.11}$$

Wie in [102] näher ausgeführt, kann diese Funktion nicht unmittelbar über die Abschätzung der Jaccard-Ähnlichkeit zu Bloom-Filtern ausgewertet werden, da die Jaccard-Ähnlichkeit die Vereinigung von Bloom-Filtern nicht im

gewünschten Maße unterstützt. In [102] wird deshalb vorgeschlagen, für diesen speziellen Fall den Dice-Koeffizenten zu verwenden, zu dem die Jaccard-Ähnlichkeit monoton ist, der aber ohne eine Vereinigungsoperation auskommt:

$$sim_{Dice}(A, B) = \frac{2\,|A \cap B|}{nA + nB}.$$

Dieser Dice-Koeffizent stellt dieselbe Ordnung über die Elemente her wie die Jaccard-Ähnlichkeit. Sein Inverses erfüllt jedoch die Dreiecksungleichung nicht und stellt deshalb keine Metrik oder gültige Distanzfunktion dar. Der Zusammenhang zur Jaccard-Distanz lässt sich folgendermaßen beschreiben:

$$sim_{Dice}(A, B) \leq sim_{Dice}(C, D) \Leftrightarrow sim_{Jacc}(A, B) \leq sim_{Jacc}(C, D).$$

Damit kann die in 3.11 beschriebene `mindist`-Funktion realisiert werden. Bloom-Filter können so auch zur Beschleunigung von $k$-nächste-Nachbarn-Suchen eingesetzt werden, indem größere Bereiche eines Suchraums über einen zusammenfassenden Filter ausgeschlossen werden können. Natürlich setzt das eine effiziente Organisation der Speicherbereiche voraus. Darauf soll an dieser Stelle aber nicht weiter eingegangen werden.

In diesem Abschnitt wurde gezeigt, wie erstmals die Jaccard-Ähnlichkeit zweier Mengen anhand ihrer Bloom-Filter-Beschreibungen approximiert werden kann. Damit ist ein generischer, privatsphäreschonender Vergleich von Mengen möglich, der theoretisch auch von nicht-vertrauenswürdigen Dritten durchgeführt werden kann. Darüber hinaus wurde eine Arbeit vorgestellt, die unmittelbar auf dieser Abschätzung aufbaut und so weiterentwickelt, dass damit eine $k$-nächste-Nachbar-Suche ermöglicht und optimiert werden kann.

## 3.4 Zusammenfassung

In diesem Kapitel wurden Bloom-Filter, spezielle Varianten und ihre jeweiligen Einsatzzwecke beschrieben. Ursprünglich als probabilistische Datenstruktur zur Beschreibung von Mengen entwickelt, eignen sich Bloom-Filter auch zur Beschreibung von Kontexten. Ein Kontext ist in der Regel eine Menge heterogener Daten, die sich durch unterschiedliche Datentypen, schwankende Größen der Datenobjekte und verschiedene Modalitäten der Elemente auszeichnt. Indem Kontexte mit Bloom-Filtern ausgedrückt werden, ergibt sich eine generische, typlose und privatsphäreschonende Kontextbeschreibung, wie in Abschnitt 3.2 dargelegt wurde.

Anhand dieser generischen Beschreibung können die Kontextausprägungen nun erstmals auch allgemeingültig verglichen werden. Wo die Heterogenität der Kontextinformationen sonst keinen typlosen Vergleich erlaubt, hilft die generische Beschreibung durch Bloom-Filter. Wie in Abschnitt 3.3 dargelegt wurde, kann an zwei Bloom-Filtern die Ähnlichkeit der zugrundeliegenden Mengen beschrieben werden. Die Ähnlichkeit von Kontexten wird in diesem Zusam-

menhang also festgestellt als Grad der Übereinstimmung der Kontextinformationen, der als Ähnlichkeit zwischen Bloom-Filtern erstmals typlos abgeschätzt wird.

Darüber hinaus wurden im Abschnitt 3.1 einige Varianten von Filtern vorgestellt, die entweder neue Operationen auf Bloom-Filtern einführen (Löschen, Histogrammberechnung und andere), die Definition der Filter modifizieren, um die Falschpositivrate zu reduzieren, oder die besondere Strukturen unter den zugrundeliegenden Daten ausnutzen, um bestimmte Anwendungsfälle abbilden zu können. Unter diesen wurden mit den Gaußschen und den OVSF-kodierten Filtern auch zwei Varianten beschrieben, die im Rahmen dieser Arbeit entstanden sind.

Das folgende Kapitel beschreibt nun ein spezielles Adressierungsschema, das auf Bloom-Filtern und den Vergleichsoperationen zwischen Bloom-Filtern aufbaut, um damit den Aufbau kontextzentrischer Sozialer Netze zu ermöglichen.

# 4 Kontextzentrische Soziale Netze und Authentizität

Eine kontextbasierte und dennoch modalitätenunabhängige Vernetzung ist das Thema dieses Kapitels. In den vorherigen Kapiteln wurde herausgearbeitet, dass Kontextinformationen für Routing in Sozialen Netzen noch kaum zum Einsatz kommen, weil notwendige Vergleichsoperationen die Privatsphäre der Nutzer beeinträchtigen können und ein pervasives Routingprotokoll nur schwer erweiterbar wäre. Dabei bieten Bloom-Filter eine Möglichkeit zur typlosen Beschreibung von Kontextinformationen und zum generischen privatsphäreerhaltenen Vergleich. In diesem Kapitel wird darauf aufbauend das Konzept der kontextzentrischen Sozialen Netze beschrieben.

In diesen neuartigen Sozialen Netzen wird erstmals eine informationszentrische Adressierung von Profilen realisiert. Eine Vernetzung kommt nur über den Vergleich von Kontexten zustande. Damit werden in kontextzentrischen Sozialen Netzen nur Verbindungen unter Personen etabliert, die in der Offline-Welt Kontexte miteinander teilen. So spiegeln die Verbindungen in diesen Sozialen Netzen gewissermaßen die Zugehörigkeiten zu Sozialen Gruppen der Offline-Welt wider. Die inhärente Trennung der Kommunikationskanäle zu unterschiedlichen Gruppen hat weitreichendes Potenzial für das Kommunikationsverhalten der Nutzer.

Gleichzeitig erlaubt der generische Aufbau dieses kontextzentrischen Routings neuartige Anwendungen und Dienste beispielsweise für das *Internet-of-things*, dynamische Kollaboration von Diensten oder andere Anwendungen mit Kontextbezug.

**Veröffentlicht** wurde dieses Konzept und insbesondere die Definition kontextzentrischer Sozialer Netze in Abschnitt 4.1.3 bereits in [241]. Abschnitt 4.1.5 gibt darüber hinaus einen Ausblick für den allgemeinen Nutzen des kontextzentrischen Adressierungsschemas. Der genannte Abschnitt basiert auf Vorarbeiten, die in [243] und [240] veröffentlicht wurden. Der letzte Abschnitt 4.2 widmet sich erstmals dem Aspekt der Authentizität in diesem neuartigen Typ von Sozialen Netzen.

## 4.1 Kontextzentrische Soziale Netze

Kontextzentrische Soziale Netze sind informationszentrische Netzwerke mit Kontextbezug, in denen der Austausch von Informationen an die Ähnlichkeit von Kontexten geknüpft ist. Bevor diese neuartige Architektur in Abschnitt

4.1.3 vorgestellt wird, werden zunächst die wichtigsten Bestandteile des Konzepts beschrieben: Das Paradigma informationszentrischer Netzwerke wird in Abschnitt 4.1.1 aufgegriffen und es folgt das Paradigma dezentraler Sozialer Netze in Abschnitt 4.1.2. Bloom-Filter als die zentrale Datenstruktur zur typlosen Beschreibung von Kontexten und dem Ähnlichkeitsvergleich zwischen diesen wurden bereits ausführlich in Kapitel 3 vorgestellt.

### 4.1.1 Informationszentrische Netzwerke

Netzwerke informationszentrisch zu organisieren, ist ein Vorschlag, der im Zusammenhang mit der rapide wachsenden Zahl an internetfähigen Endgeräten entwickelt wurde. Demnach wird die Architektur des Internets an die Anzahl verbundener Endgeräte und den prognostizierten Datenverkehr angepasst [103]. Während das hostbasierte Modell, auf dem die Struktur des Internets heutzutage basiert, ursprünglich dazu diente, Kommunikationskanäle zwischen zwei Endpunkten zu etablieren, hat sich die Nutzung des Internets inzwischen stark von diesem Anwendungsfall entfernt: Heutzutage werden Nutzer mit Inhalten verknüpft. Diese Inhalte können Videos bei YouTube [232], Dateien bei BitTorrent [233], aktuelle Meldungen von Nachrichtenportalen oder die Ergebnisse einer Google-Suche sein. Die Übertragung solcher Inhalte hat einen großen Anteil an den insgesamt über das Internet übertragenen Datenmengen. Inzwischen gibt es zahlreiche kommerzielle Anbieter sogenannter *content-delivery*-Infrastrukturen, die die Produzenten von Inhalten bei der Bereitstellung selbiger entlasten. Diese Anbieter betreiben häufig mehrere zehntausend Server weltweit, auf denen Multimedia-Inhalte, Webseiten oder Software-Aktualisierungen so verteilt werden, dass die Übertragungswege zum (mobilen) Konsumenten so weit wie möglich verkürzt werden.

Um dem enormen Overhead an Infrastruktur und Übertragungskapazitäten zu begegnen, beschäftigt sich die Forschung seit einiger Zeit mit informationszentrischen Architekturen von Netzwerken [104], [105]. In solchen Netzwerken werden Inhalte über ihren Namen und unabhängig von ihrem Speicherort adressiert. Die Adressierung der Inhalte unterscheidet sich damit grundlegend von den heute gebräuchlichen *Uniform Resource Locators (URLs)*, den global einheitlichen Identifikatoren für Ressourcen im World Wide Web, die immer auch den Anbieter eines Inhalts referenzieren: Um beispielsweise die Bedienungsanleitung der neuen Stereoanlage herunterzuladen, muss der genaue Speicherort des Dokuments auf den Servern des Herstellers in Form einer *URL* angegeben werden. Problematisch wird es vor allem, wenn sich der Inhalt der Datei ändert, die sich hinter der *URL* verbirgt. Dann ist erstmal unklar, woher die gewünschte Information bezogen werden kann.

In informationszentrischen Netzwerken wird eine Anfrage an den Namen eines Inhalts gestellt. Diese Anfrage wird dann im ganzen Netz verteilt. Es antwortet ein beliebiger Knoten, der eine autorisierte Kopie des gewünschten Inhalts gespeichert hat. Nach der Bedienungsanleitung kann zum Beispiel mit

der genauen Modellbezeichnung gefragt werden. Und wenn der Hersteller das Modell aus dem Sortiment nimmt und damit auch die Inhalte von seinen Servern löscht, würde ein beliebiger anderer Knoten im Netz auf die entsprechende Anfrage reagieren können.

An diesem simplen Beispiel wird allerdings auch eine große Herausforderung informationszentrischer Architekturen deutlich: Die Benennung der Informationsobjekte ist nicht trivial. Wie soll eine spezifische Anfrage nach der Bedienungsanleitung eines bestimmten Modells formuliert werden? Soll ein beliebiger Name aus Modellnummer und Hersteller generiert werden? Soll ein technischer Identifikator der gesuchten Datei eingesetzt werden? Soll die Bedienungsanleitung überhaupt als Datei ausgeliefert werden oder besser als strukturierter Text?

Um eine informationszentrische Architektur zu realisieren, sind drei Aspekte von zentraler Bedeutung: die Repräsentation von Inhalten als Informationsobjekte, das Namensschema, nach dem beliebige Informationsobjekte global eindeutig referenziert werden und eine namensbasierte Routing- und Weiterleitungsstrategie, nach der Inhalte zwischen Konsumenten und Anbietern vermittelt werden. Bisherige Arbeiten zu informationszentrischen Netzen konzentrieren sich daher stark auf Namensschemata und Repräsentation von Informationen [104]–[106].

Intelligente Routing- und Weiterleitungsstrategien wiederum sollen den Übertragungsaufwand effizient verteilen und reduzieren helfen. So kann beispielsweise die Auswahl des ausliefernden Hosts durch die Entfernung zum anfragenden Nutzer, die Auslastung des Servers oder die Belastung einzelner Teile des Netzwerks beeinflusst werden. Stark frequentierte Übertragungsrouten können zudem durch ein automatisches Zwischenspeichern (*caching*) bestimmter Inhalte entlastet werden. Dass dabei nur unveränderte, echte Kopien durchs Netzwerk zirkulieren, stellt das Namensschema und eine Überprüfung bei weiterleitenden Knoten sicher; die allgemeine Routingstrategie des Netzwerks sorgt dafür, dass der ausliefernde Knoten gegenüber dem Nutzer transparent bleibt. Insgesamt ist ein informationszentrisches Netzwerk damit ausfallsicherer, zuverlässiger gegenüber *denial-of-service*-Angriffen und besser an die Mobilität der Nutzer angepasst [97], [107]. Das automatische Caching innerhalb des Netzwerks und die inhärente Authentisierung der weitergeleiteten Inhalte sind zusätzliche Vorteile gegenüber heutigen Datenübertragungsstrategien im Internet [108].

In informationszentrischen Netzwerken können außerdem Abonnement-Modelle von Inhalten – sogenannte *publish-subscribe-pattern* – einfach umgesetzt werden: Die Anfrage nach einem Inhalt kann gestellt werden, bevor der Inhalt verfügbar ist. Sie wird beantwortet, sobald der Inhalt veröffentlicht wurde. Diese Eigenschaft ist vor allem für kontextzentrische Soziale Netze interessant, in denen Nachrichten zugestellt werden können, sobald ein Nutzer in den Kontext eintritt, in dem diese Nachrichten veröffentlicht wurden.

Gleichwohl das informationszentrische Paradigma effiziente Datenübertragungsstrategien für das Internet bietet und vielversprechende Anwendungen ermöglicht, sind noch zahlreiche Details ungeklärt. So gibt es noch keine Architektur, Protokolle oder Namensschemata, deren Standardisierung unmittelbar bevorstünde [108], [109]. Zudem gibt es einige Herausforderungen bezüglich der Stabilität und Sicherheit informationszentrischer Netze zu meistern, wie zum Beispiel einzelne Knoten vor Überlastung zu schützen, Inkonsistenzen unter den verteilt gespeicherten Kopien von Inhalten zu verhindern und fehlerhafte oder schädliche Inhalte vor deren Verbreitung zu erkennen und zu entfernen [110].

Vielen bisher vorgeschlagenen Architekturen informationszentrischer Netze ist gemein, dass Knoten sich über die zwischengespeicherten Inhalte in Form von Metdaten austauschen [109]. Als Metadaten werden Informationen über Daten bezeichnet; in informationszentrischen Netzwerken sind Metadaten also Repräsentationen von Informationsobjekten, die den Inhalt der Informationsobjekte möglichst umfassend beschreiben. Typische Metadaten sind die Größe des Informationsobjekts, sein Datentyp, ein Hash-Wert und andere technische Details, die direkt aus dem Informationsobjekt berechnet werden können. Darüber hinaus können Metadaten auch semantische Informationen enthalten, wie zum Beispiel „Dieses Informationsobjekt enthält ein Foto von Bob“. Entscheidend ist, dass die Metadaten deutlich kleiner sind als die Informationsobjekte selbst und dennoch genügend Informationen enthalten, um angemessene Routing- und Caching-Entscheidungen zu ermögliche.

Metadaten sollten darüber hinaus Operationen wie Aggregation, Vereinigung und verteilte Vergleiche unterstützen, um die Komplexität von Routing-Entscheidungen und weiterleitenden Infrastrukturknoten reduzieren zu können. Aggregierte Metadaten enthalten dabei eine zusammenfassende Beschreibung einer Menge von Informationsobjekten. So sollte ein Knoten in der Lage sein, alle bei diesem Knoten zwischengespeicherten Informationsobjekte über die Aggregation der entsprechenden Metadaten beschreiben zu können. Können aus den so aggregierten Metadaten zweier Knoten (also zweier Mengen von Informationsobjekten) die bei beiden Knoten gespeicherten Informationsobjekte wiederum über eine Aggregation zusammenfassend beschrieben werden, spricht man von einer Vereinigung der Metadaten. Mit einer verteilten Vergleichsoperation sollen zwei Knoten anhand der aggregierten Metadaten des jeweils anderen entscheiden können, welche Informationsobjekte nur bei einem der beiden Knoten gespeichert sind.

An dieser Stelle wird erneut die zentrale Rolle von Bloom-Filtern für kontextzentrische Netze deutlich: Neben der typlosen und modalitätenunabhängigen Beschreibung von Mengen unterstützen sie auch die zum Routing von Informationsobjekten notwendigen Operationen. Bloom-Filter erlauben also die informationszentrische Adressierung von Kontexten.

Bevor eine präzise Definition kontextzentrischer Sozialer Netze aufgestellt wird, widmet sich der folgende Abschnitt zunächst verteilten Sozialen Netzen.

### 4.1.2 Verteilte Soziale Netze

Soziale Netzwerke werden häufig als Client/Server-Architektur im Internet bereitgestellt. Die Verteilung auf mehrere physische Netzwerkknoten dient dann meist der Skalierbarkeit und Ausfallsicherheit eines zentralen Dienstes. In diesen Konstellationen kennt der zentrale Dienst den kompletten Sozialen Graphen und kann auf verschiedenste Arten helfen, Verbindungen zwischen Profilen zu etablieren. Obgleich das als großer Vorteil wahrgenommen werden kann, haben diese Sozialen Netze häufig auch Nachteile für ihre Nutzer: *Lock-in*-Effekte verhindern einen offenen Austausch mit anderen Sozialen Netzen, die Dienstanbieter verfolgen undurchsichtige kommerzielle Interessen oder die Nutzung birgt Risiken für die Privatsphäre der Nutzer.

Zahlreiche Forschungsarbeiten stellen deshalb verteilte Architekturen für Soziale Netze vor. Diese sind definiert als Soziale Netze, in denen keine zentrale (Rechen-)Einheit existiert. Motiviert sind diese Arbeiten vor allem dadurch, den Nutzern mehr Autonomie und Kontrolle über ihre privaten Daten zu geben [111]. Private Daten wie Profilinformationen bleiben meist nur auf den persönlichen Endgeräten der Nutzer gespeichert. Ihre Weitergabe ist entweder nur begrenzt möglich oder der Zugriff eingeschränkt – zum Beispiel nur auf Personen, zu denen eine Beziehung im Sozialen Graphen besteht.

Die bekannten Funktionalitäten Sozialer Netze wie Nachrichtenaustausch unter den Nutzern, Abrufen von Profilinformationen oder Austausch von Multimediainhalten werden häufig in direkter Kommunikation zwischen den Endgeräten über Peer-to-Peer-Verbindungen realisiert [112]. Die Komplexität in diesen Architekturen liegt in der permanenten Verfügbarkeit von Nutzerprofilen und der effizienten Suche in diesem verteilten Aufbau. Die permanente Verfügbarkeit wird meist durch Strategien angenähert, die die Profilinformationen entweder an Nachbarn im Sozialen Graphen oder verschlüsselt überall im Netzwerk verteilen [24], [64], [113]–[118]. Zur effizienten Suche wurden zudem Protokolle vorgeschlagen, die die Eigenschaften von Knoten im Sozialen Graphen berücksichtigen. Suchanfragen werden demnach beispielsweise über solche Profile geleitet, die besonders viele Verknüpfungen zu anderen Profilen aufweisen oder die wichtige Verbindungen zwischen sozialen Gruppen herstellen [119].

Neben diesen Arbeiten, die versuchen, Soziale Netze wie Facebook ([227]) oder LinkedIn ([226]) in eine verteilte Architektur zu übertragen, gibt es auch Arbeiten, die Mikroblogging-Dienste wie Twitter in einem verteilten Aufbau nachbilden. Im Allgemeinen erlauben Mikroblogging-Dienste, kurze Textnachrichten an interessierte Abonnenten zu verteilen. In diesen Anwendungen liegt die Komplexität vor allem im verteilten Routing der Nachrichten [120], [121].

### 4.1.3 Kontextzentrische Soziale Netze

Aus der kontextsensitiven Adaptivität informationszentrischer Netze, den generischen Kontextbeschreibungen durch Bloom-Filter und den positiven Eigen-

schaften verteilter Sozialer Netze für die Privatheit persönlicher Daten können kontextzentrische Soziale Netze folgendermaßen definiert werden:

**Definition**:
Ein kontextzentrisches Soziales Netz ist ein Soziales Netz, in dem die Kanten des Sozialen Graphen allein durch Kontextinformationen und Algorithmen zum Vergleich dieser Kontextinformationen etabliert werden. Eine Kante zwischen zwei Profilen existiert für ein definiertes Informationsobjekt genau dann, wenn diese Profile eine Kontextinformation teilen, die von dem Herausgeber des Informationsobjekts als relevant erachtet wird.

Ein kontextzentrisches Soziales Netz ist also in erster Linie ein verteiltes Soziales Netz. Darin wird die Kommunikation zwischen Profilen entsprechend dem Paradigma informationszentrischer Netzwerke ermöglicht. Jeder Knoten im Netzwerk ist in der Lage, seinen Kontext mit einem Bloom-Filter zu beschreiben. Die Kommunikation wird in Form eines Informationsobjekts an einen Bloom-Filter adressiert und im Netzwerk verteilt. Jeder Knoten, dessen Kontextbeschreibung eine gewisse Ähnlichkeit zu der Adresse des Informationsobjekts aufweist, erhält das Informationsobjekt und wird so zum Kommunikationspartner. In kontextzentrischen Sozialen Netzen werden Profile allein aufgrund der Ähnlichkeit ihrer Kontexte miteinander vernetzt – dies ist in Abbildung 4.1 schematisch dargestellt.

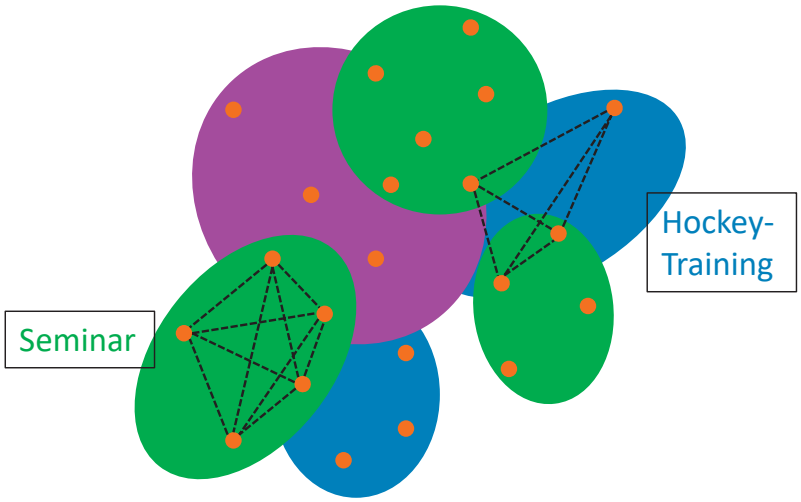



Abbildung 4.1: In kontextzentrischen Sozialen Netzen entstehen Verknüpfungen und Kommunikationskanäle nur in ähnlichen Kontexten

Der wichtigste Effekt dieser kontextzentrischen Vernetzung besteht darin, dass es keine *pollution*, also Verschmutzung, des Sozialen Graphen gibt (siehe 2.3). Die Kanten im Sozialen Graphen kommen nur durch tatsächliche, zeitgleiche Interaktion zustande. Das führt dazu, dass in dieser Art Sozialer Netze keine inaktiven Beziehungen oder virtuellen Freundschaften existieren.

Das heißt vor allem, dass die Kommunikation in kontextzentrischen Sozialen Netzen strengen Limitierungen unterworfen ist: Nur wenn Herausgeber und Abonnent eines Informationsobjekts gewisse Bedingungen zur Kontextähnlichkeit erfüllen, wird das Informationsobjekt ausgetauscht. Gegenüber Sozialen Netzen, die eine kontextabhängige Vernetzung über die Reichweite von bestimmten Funkübertragungsstandards wie Bluetooth etablieren, haben kontextzentrische Soziale Netze den entscheidenden Vorteil, dass die Ähnlichkeit von Kontexten nicht von räumlicher Nähe allein abhängt.

Der Einsatz von Bloom-Filtern für die kontextzentrische Vernetzung ist der zentrale Punkt für das beschriebene Konzept und gleichzeitig seine größte Stärke. Indem Kontexte durch Bloom-Filter beschrieben werden, ist die Adressierung völlig unabhängig von Modalitäten, Sensoren oder Datentypen möglich – ganz im Gegensatz zu den in Abschnitt 2.3.5 beschriebenen Überlegungen. Auch die Ähnlichkeit der Kontexte kann unabhängig von den Eigenschaften der zugrundeliegenden Quellen der Kontextinformationen allein anhand des Vergleichs von Bloom-Filtern beschrieben werden. Diesem Aspekt hat sich der Abschnitt 3.3 ausführlich gewidmet.

Bloom-Filter erlauben darüber hinaus auch die blinde Extraktion von Teilmengen der Kontextbeschreibungen, etwa über eine binäre UND-Verknüpfung mehrerer Filter oder den Vergleich zu bekannten Ausprägungen – in letzterem Fall mit geringer Wahrscheinlichkeit für falschpositive Annahmen über die Teilmengen.

Inwiefern der Einsatz von Bloom-Filtern die Stärke des kontextzentrischen Konzepts darstellt, illustrieren die folgenden Beispiele. Zum einen kann mit diesem Konzept ein Nachrichtenaustausch an Orte innerhalb von Gebäuden geknüpft werden. Dazu werden Ortsbeschreibungen konstruiert, die die MAC-Adressen der an einem Ort empfangbaren WLAN-Basisstationen verwenden. Nachrichten sind dann in zwei Fällen empfangbar: Entweder stimmen die Bloom-Filter dieser Ortsbeschreibung von Herausgeber und Abonnent zu einem gewissen Maße überein, oder die Bloom-Filter von Herausgeber und Empfänger enthalten beide eine bestimmte, bekannte MAC-Adresse. Im ersten Fall wird zur Adressierung ein Vergleich der Bloom-Filter verwendet, und Nachrichten können ausschließlich an dem beschriebenen Ort empfangen werden. Im zweiten Fall werden Nachrichten über eine gezielte Anfrage an die Bloom-Filter zugestellt. Damit können Nachrichten an alle Personen geschickt werden, die sich innerhalb des Gebäudes aufhalten, weil deren Ortsbeschreibungen eine der in diesem Gebäude verfügbaren MAC-Adressen enthalten.

Genauso können Nachrichten auch an Orte auf der Erde geschickt werden. Dazu bieten sich als Ortsbeschreibungen Z-Kurven-kodierte WGS84-Koordinaten an, die auch als GeoHashes bekannt sind [231]. Eine solche Kodierung begrenzter Länge beschreibt einen bestimmten Bereich auf der Erdkugel und ermöglicht damit eine entsprechende Adressierung bestimmter Orte, ähnlich zu sogenannten Geocookies [122]. Nachrichten werden hier aber über dieselben Mechanismen zugestellt, wie innerhalb von Gebäuden – nämlich über

den Vergleich von Bloom-Filtern oder gezielte Anfragen an Bloom-Filter –, obwohl die zugrundeliegenden Ortsbeschreibungen völlig anderen Semantiken und Modalitäten unterliegt.

Mit demselben Adressschema können aber auch Aktivitäten adressiert werden. Denkbar ist, dass ein Schritterkennungsalgorithmus auf einem mobilen Endgerät die Aktivität „Joggen“ erkennt, die Teil einer zuvor definierten Menge von Aktivitäten ist und durch einen bekannten String repräsentiert wird. Indem nun diese Aktivität in Form des Strings in den Filter eingefügt wird, können andere Jogger adressiert werden. Weil Nachrichten zu schreiben beim Joggen eine Herausforderung sein kann, liegt der Nutzen hier auch eher bei anderen Anwendungsfällen. Profitieren könnten beispielsweise kollaborative Musikempfehlungssysteme, die Empfehlungen für bestimmte Aktivitäten aussprechen und im Netz verteilen. Oder aber der Läufer wird auf andere Läufer in seiner Umgebung aufmerksam gemacht, indem zusätzlich zur Aktivitäts- auch eine Ortsbeschreibung in den Bloom-Filter einfließt.

Der Kontext eines Nuters kann aber auch unabhängig von dessen Aufenthaltsort oder Aktivität über bestimmte Eigenschaften beschrieben werden. Für eine firmeninterne Kommunikation kann beispielsweise der Arbeitgeber und die Abteilung eines Nutzers über einen Bloom-Filter beschrieben werden – die Kollegen könnten dann auch von unterschiedlichsten Orten auf der Welt Nachrichten miteinander austauschen.

Die bisher aufgeführten Beispiele erreichen eine Verknüpfung über den Vergleich der Bloom-Filter von Herausgeber und dem Abonnenten eines Informationsobjekts oder über eine gezielte Abfrage nach bestimmten diskreten Kontextwerten. Aus Bloom-Filtern können aber auch blind Teilmengen extrahiert werden, um etwa aus einer Historie von Kontextbeschreibungen diejenigen Elemente zu identifizieren, die über eine gewisse Zeit konstant geblieben sind. Dazu soll das Beispiel betrachtet werden, wonach aus einer Historie von Aufenthaltsorten ein allgemeiner, wichtiger Aufenthaltsort extrahiert werden soll. Die Aufenthaltsorte werden über Bloom-Filter beschrieben, die die Namen der an einem Ort verfügbaren WLAN-Netzwerke zusammenfassen. In einem Bürogebäude sind neben dem überall verfügbaren Firmennetzwerk auch zahlreiche weitere Netzwerke einzelner Abteilungen sichtar. Über die binäre UND-Verknüpfung aller Filter würden die kurzzeitig empfangenen Netzwerke der einzelnen Abteilungen als unwichtiger für die Beschreibung der Historie erkannt. Als wichtiges Element würde das überall verfügbare Firmennetzwerk übrig bleiben. Nachrichten können so an alle Personen adressiert werden, die sich aktuell im Gebäude aufhalten – trotz fehlender Kenntnis, was die einzelnen Bloom-Filter beschreiben und was die extrahierte Teilmenge genau bedeutet.

Mit der Vernetzung über Bloom-Filter können Nachrichten innerhalb von Kontexten auch gezielt an bestimmte Personen adressiert werden. Es ist schließlich denkbar, dass Nutzer nicht alle anderen Nutzer im gleichen Kontext adressieren wollen, sondern nur eine bekannte Gruppe – die Verabredung zum Mensabesuch etwa, die während der Vorlesung im Hörsaal ausgemacht

wird, soll sicher nicht an alle Kommilitonen und den Dozenten geschickt werden, sondern nur die Freunde erreichen, die oft zusammen essen gehen. Dazu einigen sich die Gruppenmitglieder auf einen geheimen String, der als Präfix oder *seed* der Hash-Funktionen bei der Konstruktion der Bloom-Filter verwendet wird. Das verändert die Ausprägung der Bloom-Filter der Gruppen für dieselben Kontextinformationen so deutlich, dass Nachrichten, die an diese speziellen Gruppen-Filter geschickt werden, auch nur die Gruppenmitglieder erreichen. Die Filter der anderen Nutzer innerhalb des Kontextes erfüllen die Anforderungen an die Ähnlichkeit dagegen nicht mehr.

Die kontextzentrische Vernetzung hat möglicherweise Auswirkungen auf die Geschwindigkeit, mit der sich Informationen in dem Netzwerk ausbreiten. Insbesondere bei wenigen Teilnehmern kann die Weiterleitung von Informationsobjekten verzögert oder unzuverlässig werden. Für diesen Fall kann das Netzwerk mit Infrastrukturknoten unterstützt werden, die nur zur Weiterleitung von Informationsobjekten eingesetzt werden. Da die Routing-Entscheidungen im Kern auf den Vergleich von Bit-Arrays hinauslaufen, werden keine hohen Anforderung an die Rechenkapazität oder Komplexität dieser Infrastrukturknoten gestellt.

Das bisher beschriebene Adressierungsschema basiert also auf den folgenden drei Komponenten, die sich an den allgemeinen Anforderungen an Metadaten aus Abschnitt 4.1.1 orientieren.

1. **Typlose Beschreibung:** Die Repräsentation einer Kontextinformation ist vollkommen unabhängig von den Datentypen oder der Größe der Informationsobjekte, weil Hash-Funktionen auf die Binärdarstellung der Informationen angewendet werden.

2. **Blinde Operationen auf Teilmengen:** Subsets der beschriebenen Kontextmenge können alleine anhand der Metadatenrepräsentation (also anhand eines Bloom-Filters) extrahiert werden – zwar mit geringer Wahrscheinlichkeit für Falschpositive, aber ohne den genauen Inhalt des Metadatenobjekts kennen zu müssen.

3. **Abschätzung der Jaccard-Ähnlichkeit:** Ohne den Inhalt der Metadatenobjekte zu kennen, ist es auch möglich, die Distanz zwischen zwei Bloom-Filtern abzuschätzen. Dazu wird die Jaccard-Metrik berechnet, die auf der Abschätzung des Jaccard-Indexes beruht.

Das hier beschriebene Konzept zur kontextzentrischen Vernetzung eignet sich mit diesem allgemein formulierten Adressierungsschema auch als Basis für andere kontextsensitive Anwendungen mit oder ohne sozialen Bezug. So kann diese Architektur beispielsweise im *Internet-of-Things*, dem Internet der Dinge, zum Einsatz kommen, um Haushaltsgeräte mit Nutzern kommunizieren zu lassen. In diesem Fall adressiert das Gerät seine Nachrichten an das Umfeld der Wohnung oder des Hauses und erreicht so automatisch alle Bewohner oder andere Hausgeräte bestimmten Typs.

Aus dieser kontextzentrischen Vernetzung könnten prinzipiell Informationen über die Kontexte von Nutzern gesammelt werden, deren Privatsphäre so stark kompromittiert würde. Der folgende Abschnitt widmet sich deshalb Mechanismen, mit denen die Privatsphäre der Nutzer angemessen geschützt werden kann.

### 4.1.4 Privatsphäre in kontextzentrischen Sozialen Netzen

Da Kontextbeschreibungen potenziell private Informationen enthalten, und diese Beschreibungen in kontextzentrischen Sozialen Netzen eine zentrale Rolle spielen, wird in diesem Abschnitt untersucht, wie das vorgestellte Konzept die Privatsphäre der Nutzer schützt.

Ein wichtiger Aspekt in Sozialen Netzen ist der Schutz der Identität von Nutzern. Verschaffen sich böswillige Nutzer Zugang zu einem fremden Nutzerprofil, können sie sich im Sozialen Netz als diese fremde Person ausgeben. Der Vollzugriff auf fremde Profilinformationen, Kontaktlisten und Kommunikationen erlaubt es, eine fremde Identität zu stehlen. Zahlreiche Einzelfälle berichten von solchem Identitätsdiebstahl und damit verbundenem betrügerischen Auftreten in anderen Sozialen Netzen oder Umgebungen (zum Beispiel [234]).

Ein weiteres verbreitetes Problem Sozialer Netze besteht in der Sammlung sekundärer Daten und der Verknüpfung dieser Daten mit Profilinformationen. Angreifer sind damit beispielsweise in der Lage, gezielt Profile in unterschiedlichen Sozialen Netzen wiederzuerkennen oder Einträge in anonymisiert veröffentlichten Datensätzen zu deanonymisieren.

Die Analyse der Interaktionen innerhalb eines Sozialen Netzes ist ein weiterer möglicher Angriffspunkt. Aus dem Wissen, wer mit wem in welcher Form interagiert hat, kann bei gezieltem Einsatz gegen einzelne Personen großer Schaden für deren Privatsphäre entstehen.

Diese Aspekte sind besonders in zentral organisierten Sozialen Netzen mit klassischer Client/Server-Architektur ein Problem. Sei es, weil neugierige Dienstanbieter ganz automatisch detaillierte Kenntnisse über Interessen, Vorlieben und Beziehungen aller ihrer Nutzer erhalten, oder weil Sicherheitslecks und Fehlkonfigurationen der Infrastruktur dafür sorgen, dass Dritte unerlaubt Zugang erlangen. Einem Großteil dieser Probleme kann also mit einer dezentralen Architektur des Netzwerks begegnet werden, weil damit der zentrale Punkt für Angriffe auf das Soziale Netz eliminiert wird.

Dezentrale Architekturen sind dadurch aber anderen Gefahren ausgesetzt. Ein häufig genannter Angriffspunkt sind die Kommunikationswege – insbesondere bei drahtloser Datenübertragung [71], [73], [123]. Gefahren für die Privatsphäre enstehen zum Beispiel durch sogenannte *wormhole*-Angriffe: Danach simulieren Angreifer eine Verbindung zu einem adressierten Knoten, die eigentlich nicht existiert. Indem der Datenverkehr aber über das „Wurmloch“ der Angreifer geleitet wird, können die transportierten Inhalte analysiert wer-

den, um daraus etwa die Identität Nutzers zu extrahieren. Angreifer können sich dann an anderer Stelle mit der fremden Identität ausgeben [43], [73].

Im Folgenden sollen aber nur die Gefahren für die Privatsphäre betrachtet werden, die allein durch die kontextzentrische Vernetzung entstehen können – das vorgeschlagene Adressierungsschema kann schließlich potenziell kompromittierende Informationen über den Kontext der Nutzer offenbaren. Ein Angreifer könnte beispielsweise versuchen, die Kontextbeschreibungen einzelner Nutzer abzufangen, um so Wissen über deren Aufenthaltsort, Aktivität oder andere Kontextinformationen zu sammeln. Zu einer gegebenen Kontextbeschreibung in Form eines Bloom-Filters müsste dieser Angreifer also die Informationen rekonstruieren, die in den Filter eingefügt wurden. Dazu muss aus den Ausgaben der Hash-Funktionen – in diesem Fall also der Bit-Positionen im Bloom-Filter – auf die Eingabewerte – die Kontextinformationen – geschlossen werden. Dieser Gefahr wird mit kryptographischen Hash-Funktionen begegnet. Das sind Hash-Funktionen, die per definitionem Einwegfunktionen sind. Aus deren Ausgabewerten kann in akzeptabler Zeit nicht auf Eingabewerte geschlossen werden.

Eine weitere Gefahr besteht in möglichen *replay*-Attacken, indem ein Angreifer eine Datenbank über realistische Kontextbeschreibungen erstellt, in der Ausprägungen von Bloom-Filtern gesammelt werden. Damit wäre der Angreifer in der Lage, gezielt Nachrichten aus beliebigen Kontexten zu empfangen, die Nutzer in den entsprechenden Kontexten aktuell austauschen. Diesem Problem wird mit zufälligen Präfixen oder *seeds* für die kryptographischen Hash-Funktionen begegnet. Diese Präfixe erlauben es, unterschiedliche Arten von Gruppen zu bilden und so die Anzahl der Personen zu verringern, die realistische Filterausprägungen lernen können.

Die Art der Gruppen wird über die Variation der Präfixstrings bestimmt. Denkbar ist neben den zufälligen Präfixen auch, zeitliche Epochen einzuführen, indem in regelmäßigen Abständen neue zufällige Präfixstrings gewählt werden. Die Kommunikation wird damit zwar an die Plattform gebunden, die die zeitlichen Epochen definiert. Das direkte Lernen von Ausprägungen von Bloom-Filtern ist dann aber nur mit Kenntnis des jeweils gültigen Präfixes möglich. Etwaige gelernte Ausprägungen sind auch nur für eine entsprechende Epoche gültig. Denkbar wäre auch, die Präfixstrings zusätzlich an Orte zu knüpfen; die Ausprägungen der Bloom-Filter wären dann nur für eine Epoche und nur an dem entsprechenden Ort gültig.

An dieser Stelle sei angemerkt, dass diese Überlegungen nicht die Sicherheit der Informationsobjekte betreffen: zufällige Nachrichten können leicht abgefangen werden. Das bedeutet nicht, dass die Informationsobjekte nicht geschützt werden können oder müssen. Diese Art von Schutz kann aber leicht durch Verschlüsselung der Kommunikation erreicht werden und ist damit keine besondere Gefahr für die Privatsphäre, die aus der kontextzentrischen Adressierung erwächst. Die beschriebenen Mechanismen sollen in erster Linie das gezielte Abfangen von Nachrichten oder Interakionen bestimmter Gruppen erheblich

erschweren. Damit entstehen durch das kontextzentrische Konzept keine zusätzlichen Gefahren für die Privatsphäre, mit denen andere verteilte Architekturen nicht konfrontiert wären.

### 4.1.5 Ausblick zum kontextzentrischen Adressierungsschema

Das hier vorgestellte kontextzentrische Adressierungsschema bietet großes Potenzial für weitere Anwendungen und Konzepte. Ein Anwendungsgebiet ist die Personalisierung. Mit kontextzentrischer Vernetzung können personalisierte Dienste verteilt realisiert werden.

Eine verteilte Dienstarchitektur für einen personalisierten Musikdienst wurde in [240] vorgeschlagen. Darin wird zunächst ein hybrides Radionetzwerk beschrieben, das aus der Kombination von DAB (*Digital Audio Broadcast*, Digitaler Rundfunk) und mobiler Internetverbindung einen personalisierten Radiosender simuliert. Ungewünschte Musik im Rundfunksignal wird durch Vorschläge überblendet, die einzeln über das Internet abgerufen oder aus einem lokalen Zwischenspeicher gespielt werden. Dem Nutzer gegenüber bleibt diese Überblendung transparent: Im digitalen DAB-Signal kodieren spezielle Bits die genauen Zeitpunkte der Übergange zwischen aufeinanderfolgenden Inhalten, sodass hörbare Überschneidungen bei der Personalisierung entweder durch passende Überblendungen vermieden oder mit Hilfe von lokalen Zwischenspeichern (*timeshift-buffer*) ausgeglichen werden können. Dem Nutzer präsentiert sich so ein personalisierter Radiosender, der bekannte Services wie Wetterbericht und Staumeldungen bietet und gleichzeitig weniger Daten über das Internet übertragen muss als reine Musik-Streaming-Dienste.

Da es das digitale Rundfunksignal durch eine enstprechende Bit-Kodierung erlaubt, Musikstücke passgenau aus dem Rundfunksignal auszuschneiden, konnte das Konzept um ein kooperatives Element erweitert werden. Die Clients der Radiohörer sind so in der Lage, eine lokale Datenbank beliebter Inhalte aufzubauen. Über die Gesamtheit der Clients entsteht in dem Empfangsgebiet eines Radiosenders – also in einer geografischen Region – ein verteilter Speicher der dort beliebten Inhalte. In [240] wurden deshalb Strategien beschrieben, mit denen Inhalte intelligent unter den Radiohörern ausgetauscht werden können. Damit sollte der Streaming-Aufwand des Radiosenders verringert werden, ohne die Datenübertragungskapazitäten einzelner Clients unverhältnismäßig stark zu beanspruchen. Die Disseminationsstrategien setzten vor allem auf die Adressierung geografischer Regionen über GeoHash-kodierte Bloom-Filter – GeoHashes sind Z-kurvenkodierte WGS84-Koordinaten, die begrenzte geografische Regionen auf der Erdkugel beschreiben [231].

Zusammen mit dem kontextzentrischen Adressierungsschema lässt sich dieses Konzept weiterdenken zu einem Netzwerk kontextabhängiger und kooperativer Personalisierung mit inhärentem Schutz der Privatsphäre der Nutzer. Anbieter können Präferenzen kontextabhängig lernen, indem Zusammenhän-

ge zwischen angefragten Inhalten und bestimmten Ausprägungen von Bloom-Filtern beobachtet werden. Durch die im vorherigen Abschnitt beschriebenen Mechanismen sind aus den Filterausprägungen aber keine Rückschlüsse auf die Kontexte der Nutzer möglich – deren Privatsphäre ist also geschützt. Der kooperative Aspekt tritt hervor, wenn Inhalte mit besonderer Relevanz in bestimmten Kontexten mit anderen Nutzern aus demselben Kontext geteilt werden können.

Weitreichende Implikationen ergeben sich für die kontextzentrische Vernetzung allgemein, wenn Nutzer beweisen können, dass sie einen bestimmten Kontext erlebt haben. Die bisher beschriebenen Mechanismen sorgen dafür, dass nur Knoten vernetzt werden, die wirklich einen Kontext miteinander teilen. Innerhalb dieser Subnetze, die womöglich zeitlich begrenzt oder nur einmalig bestehen, können die verbundenen Knoten ein gemeinsames Geheimnis aushandeln, das stark von diesem Kontext abhängig ist. Denkbar ist etwa, einen String zu verschlüsseln, den nur die Teilnehmer des Subnetzes kennen. Zur Generierung der nötigen Schlüssel können quasizufällige Kontextinformationen eingesetzt werden, die stark von Umwelteinflüssen abhängig sind und deshalb schwer rekonstruiert werden können. Dieses ausgehandelte und verschlüsselte Geheimnis muss dann vor nachträglicher Fälschung geschützt werden. Das gelingt zuverlässig, wenn das gesamte Netz über den aktuellen Stand des Geheimnisses einen Konsens bildet und ihn so bestätigt. Der Konsens kann dann in einer öffentlich einsehbaren Speicherstruktur abgelegt werden.

Als Vorbild dieser netzweiten Absicherung von Interaktionen zwischen wenigen Teilnehmern im Netz dient die *block chain*, die öffentlich einsehbare Historie aller Transaktionen, die in der virtuellen Währung *BitCoin* getätigt wurden [235]. Als *block chain* wird eine verteilte Datenbank bezeichnet, in der die Datensätze jeweils eine Prüfsümme des zuletzt abgelegten Datensatzes enthalten. Das erzeugt eine logische Kette der Datensätze, die in einer relativen zeitlichen Reihenfolge abgelegt sind. Dadurch wird insbesondere die Integrität der vorangegangenen Datensätze gesichert.

In Bitcoin dient die verteilte Architektur der *block chain* auch der Anonymität der Teilnehmer. Die gespeicherten Transaktionen enthalten keine Informationen zu den beteiligten Personen. Kompromittiert werden könnte die Anonymität eines Transaktionsteilnehmers höchstens durch zusätzliche Informationen, etwa seine IP-Adresse. Zum Bitcoin-Protokoll gehört deshalb auch ein Mechanismus, der den Ursprung von Transaktionen verschleiern soll, indem Informationen über neue Transaktionen möglichst schnell möglichst weit im Bitcoin-Netz verteilt werden. Der Kern dieses Mechanismus wurde in [243] und [237] genauer untersucht. Dabei zeigte sich, dass neue *peers* zwar schnell mit zahlreichen unterschiedlichen Teilnehmern bekannt gemacht werden, diese sich aber auf wenige Autonome Systeme im Internet verteilen. Für die Anonymität der Bitcoin-Teilnehmer kann das zur Gefahr werden – etwa wenn eine Transaktion eindeutig aus einem Autonomen System stammt, in dem nur ein einziger Bitcoin-Teilnehmer existiert.

Im Zusammenhang mit der kontextzentrischen Vernetzung könnte daraus aber ein Vorteil erwachsen. Werden die Autonomen Systeme mit den kontextabhängigen Subnetzen gleichgesetzt, in denen die Teilnehmer einen Kontextbeweis aushandeln, wird für alle Teilnehmer des Netzes transparent, wer einen Kontext geteilt hat. Dadurch ergeben sich beispielsweise Anwendungsmöglichkeiten für *smart contracts*, also elektronische Verträge, deren Entstehung so vom ganzen Netz bestätigt werden. Für Soziale Netze kann mit Hilfe dieser Transparenz auch die Integrität der Nutzerprofile sichergestellt werden. Ist der geteilte Kontext etwa die unmittelbare persönliche Anwesenheit, kann etwa die Integrität des Profilfotos des Gegenübers bestätigt und in einem netzweiten Konsens festgehalten werden. Die Integrität eines Profils wächst dann mit steigender Anzahl solcher bestätigenden Interaktionen mit möglichst vielen verschiedenen anderen Profilen.

Der folgende Abschnitt erläutert, wie die bisher beschrieben Architektur kontextzentrischer Netze eine authentische Kommunikation in Sozialen Netzen ermöglichen kann.

## 4.2 Authentizität in kontextzentrischen Sozialen Netzen

Die im vorangegangenen Abschnitt beschriebene Architektur eines kontextzentrischen Sozialen Netzes erzeugt erstmals eine Kommunikationssituation, die ein vergleichbares Maß an Authentizität wie die *Face-to-Face*-Kommunikation in der Offline-Welt erlaubt.

Eingangs wurden drei wichtige Dynamiken erläutert, die die Online-Kommunikation in Sozialen Medien bisher fundamental von der Offline-Kommunikation unterschieden haben: das unbekannte Publikum, die verschwindende Grenze zwischen Öffentlichem und Privatem und der sogenannte *context collapse*, mit dem das Zusammenfallen eigentlich getrennter sozialer Gruppen in einer großen Freundesliste bezeichnet wird [12]. Insbesondere dieser Zusammenfall getrennter sozialer Gruppen wurde als problematisch für eine authentische Kommunikation identifiziert. Weil das Einschränken des Adressatenkreises in bekannten Sozialen Netzen kompliziert und zeitaufwändig ist, versenden Nutzer nicht-öffentliche Kommunikation (sogenannte Status-Updates) in der Regel an ihre gesamte Freundesliste [20]. Verfasst werden diese Statusnachrichten häufig trotzdem mit einem bestimmten Adressatenkreis im Hinterkopf, für den auch eine bestimmte Form der Selbstdarstellung und damit eine eigene Form von Authentizität gewählt wird. Allerdings erreichen diese Statusnachrichten auch Personen, die als Mitglieder bestimmter Gruppen hätten ausgeschlossen werden sollen. Von diesen wird die gewählte Selbstdarstellung womöglich als nicht authentisch empfunden. Das führt zu Konflikten oder Spannungen und kann die Bindungen der Gruppenmitglieder untereinander schwächen [13].

Das hier vorgestellte Konzept wirkt den beschriebenen Dynamiken deutlich entgegen. Sobald eine soziale Gruppe einen Kontext teilt und darin Nachrichten austauscht, ist für diese Gruppe ein dedizierter Kommunikationskanal entstanden. Der *context collapse* wird so verhindert. Dieser Kontext kann das gemeinsame Hockey-Training oder die Comic-Messe sein. Durch die Möglichkeit, die Gültigkeit von Nachrichten räumlich, zeitlich und sozial über Präfixstrings (ort- und zeitabhängige Präfixe, beziehungsweise „Passwörter“, die nur Mitgliedern einer Gruppe bekannt sind) einzuschränken, kann wirksam verhindert werden, dass sich Empfänger unerwünscht in die Kommunikation sozialer Gruppen einschleichen. Damit wird auch wirkungsvoll vermieden, dass Nachrichten ein unbekanntes Publikum erreichen, beziehungsweise umgekehrt innerhalb von gleichen Kontexten nur intendierte Personen adressiert werden. Es müssen damit nicht zwangsläufig alle Besucher der Comic-Messe angeschrieben werden, sondern es können entweder die eigenen Freunde ausgewählt oder unbekannte Personen mit derselben Vorliebe für Superhelden-Comics angeschrieben werden.

Vor allem erfordert die Beschränkung des Adressatenkreises keinen großen Aufwand. Es müssen nicht umständlich die gewünschten Personen in langen Freundeslisten gesucht und ausgewählt werden. Die Kontextbeschreibungen können automatisch aus bestimmten verfügbaren Kontextinformationen konstruiert und beispielsweise durch bestimmte Profilinformationen verfeinert werden.

Durch diese Architektur kann allerdings nicht verhindert werden, dass Personen Nachrichten mutwillig an unerwünschte Dritte weiterleiten. Deshalb kann das Problem des unbekannten Publikums und damit die verschwindende Grenze zwischen Öffentlichem und Privatem nicht vollständig behoben werden. Es sei aber angemerkt, dass diese Dynamiken für jede Form der elektronisch vermittelten Kommunikation gelten und prinzipiell nicht verhindert werden können.

Dennoch bietet die kontextzentrische Vernetzung eine neuartige Abbildung der *Face-to-Face*-Kommunikation hinsichtlich der Trennung sozialer Gruppen. So wie Personen in der Offline-Welt ihr Verhalten an ihre jeweiligen Gesprächspartner anpassen, ist dies nun auch in dedizierten Kommunikationskanälen zu bestimmten Personen möglich. Nutzern wird ein neuartiger Rahmen für authentische Kommunikation geschaffen, der ihnen die mentale Arbeit der Selbstzensur erspart. Der virtuellen Identität werden nun Räume gegeben, sich so zu entfalten, wie es in der Offline-Welt möglich ist.

Die kontextzentrische Architektur macht indes keine Vorgaben, wie erfasst werden soll, dass eine soziale Gruppe einen Kontext teilt. Dies ist in zahlreichen Arbeiten zur Erkennung von Kontextgleichheit untersucht worden. Diese Arbeiten setzen beispielsweise auf ein Funktechnologie wie Bluetooth, deren Reichweite so stark begrenzt ist, dass eine Vernetzung nur in unmittelbarer Nähe möglich ist [42]–[44], [60], [62], [63]. Andere Vorarbeiten widmen sich der Erkennung räumlicher Nähe durch den Vergleich von Kamerabildern [124],

indem sie einen „Fingerabdruck“ eines Ortes über dort verfügbare WLAN-Netzwerke erstellen [125]–[127] oder indem sie auf spezielle WLAN-Signale und deren Signalstärke anderer mobiler Endgeräte hören [128].

Die Stärke des hier beschriebenen Adressierungsschemas ist die generische Kontextbeschreibung. Es können also alle eben beschriebenen Verfahren zur Erkennung unmittelbarer räumlicher Nähe eingesetzt werden, um innerhalb einer Gruppe eine gemeinsame Kontextbeschreibung zu ermitteln. Der Nachrichtenaustausch ist dann jedoch unabhängig von lokalen Beschränkungen solcher Verfahren, wie etwa der Reichweite der Bluetooth-Funktechnologie. Außerdem können Kombinationen mehrerer Verfahren eingesetzt werden, um eine möglichst umfassende, eindeutige Kontextbeschreibung zu formulieren.

## 4.3 Zusammenfassung

In diesem Abschnitt wurde die Architektur kontextzentrischer Sozialer Netze und die Bedeutung dieser Architektur für die authentische Kommunikation beschrieben.

Kontextzentrische Soziale Netze sind zunächst einmal informationszentrische verteilte Netzwerke. Für diese Netzwerke wurde ein neuartiges Adressierungsschema vorgestellt, das es ermöglicht, Nachrichten an Kontexte zu schicken. Ein Kontext wird dazu als eine Menge von Kontextinformationen aufgefasst und durch einen Bloom-Filter beschrieben. Indem die Jaccard-Ähnlichkeit zwischen zwei Bloom-Filtern berechnet wird, wird die Ähnlichkeit der zugrundeliegenden Mengen – also der Kontexte – ermittelt. Nachrichten werden zugestellt, wenn zwei Knoten im Netzwerk Kontextbeschreibungen aufweisen, deren Ähnlichkeit über einem definierten Grenzwert liegen. Da ein Bloom-Filter wegen der Verwendung kryptographischer Hash-Funktionen keine Rückschlüsse auf die verwendeten Kontextinformationen erlaubt, kann der Vergleich der Bloom-Filter von einem beliebigen Knoten im Netzwerk durchgeführt werden, ohne die Privatsphäre einzelner Knoten zu verletzen.

Die kontextzentrische Vernetzung kann durch geschickte Wahl der Kontextbeschreibung dedizierte Kommunikationskanäle für soziale Gruppen etablieren. Damit ist insbesondere der eingangs beschriebene *context collapse* verhindert, der bisher ein Problem für die authentische Kommunikation in Sozialen Medien darstellt. Die hier vorgestellte Architektur erlaubt somit neuartige Formen der Selbstentfaltung und Authentizität, die so bisher nur unzureichend ermöglicht wurden.

Der folgende Abschnitt untersucht ein simuliertes kontextzentrisches Netz mit Methoden der Sozialen Netzwerkanalyse. Es wird gezeigt, dass darin Strukturen entstehen, die zu einem herkömmlichen Sozialen Netz vergleichbar sind. Anschließend werden Verfahren zur privatsphäreschonenenden Datenerhebung in einem verteilten Netzwerk vorgestellt, um die praktische Einsetzbarkeit dieser Architektur auch für *Data Mining*-Anwendungen zu demonstrieren.

# 5 Data Mining in kontextzentrischen Sozialen Netzen

Zur Praxistauglichkeit Sozialer Netze gehört zweierlei: ein zuverlässiger Dienst und die Möglichkeit zum *Data Mining*. An der Zuverlässigkeit des Dienstes sind vor allem die Nutzer interessiert. Sie möchten sich mit ihren Freunden oder Bekannten vernetzen und mit diesen Nachrichten austauschen. Zuverlässig ist der Dienst vor allem dann, wenn er Nachrichtenaustausch mit den gewünschten Adressaten ermöglicht. An der Möglichkeit zum *Data Mining*, also der Erhebung von Daten innerhalb des Dienstes, sind in erster Linie die Anbieter des Dienstes interessiert. Sie könnten etwa Profilinformationen erheben, Informationen zur Dienstnutzung sammeln oder Abstimmungen unter den Nutzern durchführen wollen.

Dieses Kapitel beschäftigt sich mit diesen beiden Aspekten im Zusammenhang kontextzentrischer Sozialer Netze. Im Abschnitt 5.1 wird untersucht, ob und wie sich das vorgeschlagene Adressierungsschema für Soziale Netze eignet. Dazu wird ein Soziales Netz von eMail-Kommunikationen, das in einem öffentlich zugänglichen Datensatz enthalten ist, mit kontextzentrischer Adressierung nachgebildet. Der resultierende Soziale Graph wird mit Methoden der Graphentheorie analysiert, wie es für die Analyse Sozialer Netze üblich ist. Es wird gezeigt, dass der resultierende Soziale Graph in seiner Struktur vergleichbar ist zum tatsächlichen Sozialen Graphen des eMail-Netzwerks.

Im darauf folgenden Abschnitt 5.2 werden spezielle *Data Mining* Methoden für verteilte Systeme vorgestellt, die Garantien an die Privatheit der Daten und damit an die Privatsphäre der Nutzer geben: Die erhobenen Daten lassen keine Rückschlüsse auf einzelne Personen mehr zu. In dem genannten Abschnitt wird ein Algorithmus zur verteilten Schätzung der Verteilung eines numerischen Werts mit starken Garantien an die Privatsphäre der Nutzer beschrieben.

**Veröffentlicht** wurden Teile dieses Kapitels und insbesondere des Abschnitts 5.2.3 in [244]. Bisher unveröffentlicht sind die Untersuchungen und Ergebnisse in 5.1.2.

## 5.1 Soziale Netzwerkanalyse in kontextzentrischen Sozialen Netzen

Kontextzentrische Soziale Netze ermöglichen eine neuartige spontane, persönliche und authentische Kommunikation mit einer kontextabhängigen *publish-/subscribe*-Architektur: Nachrichten werden darin an Kontextbeschreibungen adressiert. Die Kommunikation zwischen Nutzern – auch die persönliche – erfolgt unter Umständen also über mehrere unterschiedliche Kanäle. Der spontane Nachrichtenaustausch im Hörsaal während einer Vorlesung und nachmittags im Hockey-Verein kann dieselben Personen über eine unterschiedliche Adressen erreichen. Trotzdem sollte es auch in diesen neuartigen Sozialen Netzen möglich sein, die sozialen Bindungen der Nutzer mit bekannten Methoden zur Sozialen Netzwerkanalyse zu analysieren.

Soziale Strukturen wie sie in einem Sozialen Netz entstehen, werden in der Regel am Sozialen Graphen untersucht, der die Beziehungen der Profile zueinander abbildet. Solche Untersuchungen fördern zum Beispiel zu Tage, welche Nutzer sich zu Cliquen zusammenfinden, welche Nutzer in besonders vielen Cliquen eingebunden sind und welche Nutzer durch ihre Präsenz in unterschiedlichen Gruppen für den Zusammenhalt der gesamten Gruppe von besonderer Bedeutung sind. Für diese Untersuchungen wird auf Methoden der Graphentheorie zurückgegriffen, die sich in neuerer sozialwissenschaftlicher Forschung zur gängigen Praxis entwickelt haben.

Im Folgenden werden die Methoden vorgestellt, die zur Analyse Sozialer Netze typischerweise eingesetzt werden. Anschließend wird der Soziale Graph eines kontextzentrischen Sozialen Netzes mit diesen Methoden analysiert. Dazu wird ein Experiment vorgestellt, in dem mit dem kontextzentrischen Adressierungsschema aus Kapitel 4 ein existierendes Soziales Netz aus eMail-Kommunikationen simuliert wurde.

### 5.1.1 Metriken zur Analyse Sozialer Netze

Indem die Freundschaftsbeziehungen zwischen Profilen als ungerichtete Verbindungen zwischen Knoten in einem Graphen interpretiert werden, kann aus einem Sozialen Netz der sogenannte *Soziale Graph* extrahiert werden. Das ist ein Graph $G = (V, E)$, in dem $V$ die Menge aller Nutzer und $E = \{(u, v) : u, v \in V\}$ die Menge aller Beziehungen zwischen zwei Nutzern $u$ und $v$ repräsentieren.

Die Analyse Sozialer Netze hinsichtlich ihrer Struktur ist dadurch mit Methoden der Graphentheorie möglich. Soziale Gruppen können zum Beispiel über den Grad ihrer Verbundenheit beschrieben, einzelne Knoten über ihre Verbindungen zu anderen charakterisiert werden. Diese und weitere Eigenschaften liefern Algorithmen der Graphentheorie, die im Folgenden vorgestellt werden. Diese Vorgehensweise zur Untersuchung Sozialer Geflechte ist auch in der sozialwissenschaftlichen Forschung seit langem anerkannt [129].

Mit dem Aufkommen von großen Sozialen Netzen wie Facebook, in denen Freundschaftsbeziehungen öffentlich einsehbar sind, hat sich die Analyse der Beziehungen etwas verlagert. In vielen dieser Sozialen Netze sind die geknüpften Freundschaften überwiegend inaktiv – für Facebook wurde etwa gezeigt, dass an 70% der Interaktionen eines Nutzers lediglich 20% von dessen Freunden beteiligt waren. Diese Erkenntnis konnte für 90% aller Nutzer bestätigt werden. An 100% der Interaktionen waren nur 60% der Freunde eines Nutzers involviert [130]. Für die Analyse solcher Netze wird deshalb der *Interaktionsgraph* dem Sozialen Graphen vorgezogen. Im Interaktionsgraph werden nur tatsächlich stattgefundene, direkte Interaktionen zwischen Mitgliedern eines Sozialen Netzes festgehalten.

Anhand des sozialen oder des Interaktionsgraphen können diverse graphentheoretische Kennzahlen berechnet werden, die auch im Kontext sozialer Beziehungen sinnvolle Interpretationen zulassen [129], [131], [132].

#### 5.1.1.1 Knotengrad und Dichte

Der *Grad* eines Knotens $u$ ist die Anzahl der Kanten, die mit diesem Knoten $u$ inzidieren, also mit diesem Knoten verbunden sind. In einem gerichteten Graphen kann zudem zwischen Eingangs- und Ausgangsgrad unterschieden werden. Der *Eingangsgrad* eines Knotens $u$ ist die Anzahl der Kanten, die $u$ als Endpunkt haben, sein *Ausgangsgrad* ist dementsprechend die Anzahl der Kanten, die von $u$ ausgehen. Alle Gradmaße können für einen Knoten $u$ in einem Graphen mit $g$ unterschiedlichen Knoten maximal den Wert $g-1$ annehmen – genau dann, wenn $u$ mit allen anderen $g-1$ Knoten verbunden ist.

Der Grad eines Knotens in einem (ungerichteten) sozialen Graphen entspricht letztlich der Anzahl seiner Freunde. In einem (gerichteten) Interaktionsgraphen gibt der Knotengrad durch die Unterscheidung in Eingangs- und Ausgangsgrad auch Aufschluss über die Anzahl der Interaktionen, die ein Knoten kontrollieren kann [133].

Der durchschnittliche Knotengrad $deg(G)$ eines Graphen $G$, der als Durchschnitt über die Knotengrade aller Knoten über

$$deg(G) = \frac{\sum_{u \in U} deg(u)}{|V|}$$

bestimmt werden kann, ist unter anderem hilfreich, wenn besondere Knoten in einem Netzwerk identifiziert werden sollen. Sogenannte *hubs* weisen zum Beispiel überdurchschnittlich viele Verbindungen zu anderen Knoten auf. In Sozialen Netzen (und allgemeiner in verteilten Systemen) sind sie für die effiziente Verbreitung von Informationen von zentraler Bedeutung [119].

Aus dem Verhältnis aller Knoten $V$ und Kanten $E$ in einem Graph $G$ kann dessen *Dichte* $d(G)$ bestimmt werden:

$$d(G) = \frac{2\,|E|}{|V|\,(|V|-1)}.$$

Dieses simple Maß wird im Zusammenhang mit Sozialen Netzen häufig als Indikator für den Zusammenhalt von Gruppen im gesamten sozialen Geflecht angesehen. Gleichzeitig ist die Dichte als Durchschnittswert insbesondere für größere Netze schwer zu intepretieren. In der einschlägigen Literatur wird deshalb häufig geraten, zusätzlich auch andere Maße, zum Beispiel für die Zentralität von Knoten, mit anzugeben. Diese werden weiter unten detailliert vorgestellt.

#### 5.1.1.2 Kürzeste Pfade und Distanz

Ein Pfad durch einen Graphen bezeichnet den Weg zwischen zwei Knoten $u$ und $v$, der über mehrere benachbarte Knoten führen kann, auf dem jeder Knoten aber nur einmal besucht wird. Dementsprechend kann es mehrere unterschiedliche Pfade von $u$ nach $v$ geben. Aus dieser Menge der Pfade ist derjenige der kürzeste, der von $u$ ausgehend $v$ in einer minimalen Anzahl an Schritten erreicht. Mit dieser minimalen Anzahl von Schritten wird auch die *Distanz* $d(u, v)$ zwischen zwei Knoten $u$ und $v$ bezeichnet.

#### 5.1.1.3 Zentralität und Prestige

Die Zentralität und das Prestige von Knoten sind Kennzahlen, mit der die Wichtigkeit einzelner Knoten in einem Graphen beschrieben werden. Insbesondere für sozialwissenschaftliche Untersuchungen sind diese Kennzahlen ein entscheidender Grund, warum überhaupt auf graphentheoretische Methoden zurückgegriffen wird. Beide Kennzahlen können für jeden Knoten in einem Graphen einzeln berechnet werden. Während die Zentralität sowohl in gerichteten als auch ungerichteten Graphen bestimmt werden kann, ist das Prestige eines Knotens nur in einem gerichteten Graphen zu bestimmen, da sich dieses Maß vor allem auf die eingehenden Kanten eines Knotens stützt.

Sowohl Zentralität als auch Prestige werden auf der Basis unterschiedlicher Aspekte berechnet. So wird die Grad-Zentralität $C_d$ über den Grad $d(u)$ eines Knoten $u$ als $C_d(u) = d(u)$ bestimmt. Da diese Zahl von der Anzahl der Knoten $g$ im Graph abhängt, wird häufig die normalisierte Form der Grad-Zentralität angegeben, um eine Vergleichbarkeit für unterschiedliche Graphen zu erreichen:

$$C'_d(u) = \frac{d(u)}{g-1}.$$

Knoten mit einer hohen Grad-Zentralität sind besonders sichtbar für andere Knoten. Sie nehmen in einem Graphen eine sehr zentrale, exponierte Position

ein. Demgegenüber stehen Knoten mit geringer Grad-Zentralität häufig am Rand des Netzwerks – sie sind periphere Knoten im Graphen und werden von wenigen anderen wahrgenommen.

In gerichteten Graphen wird anstelle des Knotengrads $d(u)$ die Anzahl der ausgehenden Kanten $x_{u+}$ von $u$ verwendet, da vornehmlich die von $u$ initiierten Interaktionen über die Sichtbarkeit von $u$ entscheiden.

Die *closeness*-Zentralität basiert anders als die Grad-Zentralität auf der Distanz eines Knotens zu allen anderen Knoten im Graph. In diesem Sinne ist ein Knoten besonders zentral, wenn er alle anderen Knoten im Netzwerk über möglichst wenige Zwischenschritte erreichen kann. Zur Berechnung der *closeness*-Zentralität $C_c$ kommt daher die Distanz $d(u, v)$ zum Einsatz:

$$C_c(u) = \left[ \sum_{v \in V, v \neq u} d(u, v) \right]^{-1}.$$

Dieses Maß bezieht nicht mehr nur die Knoten in der direkten Nachbarschaft mit ein, sondern alle anderen. Nimmt die Distanz eines Knotens $u$ zu allen anderen Knoten im Graph zu, nimmt die *closeness*-Zentralität von $u$ dementsprechend ab.

Da auch diese Zentralität stark von der maximal möglichen Distanz und damit von der Anzahl der Knoten $g$ im Graph abhängt, wird auch die *closeness*-Zentralität häufig in normalisierter Form angegeben:

$$\begin{aligned} C'_c(u) &= \frac{g-1}{\left[\sum_{v \in V, v \neq u} d(u, v)\right]} \\ &= (g-1)(C_c(u)). \end{aligned}$$

Interpretiert wird dieses Maß beispielsweise als Effektivität, mit der ein Knoten Informationen verbreiten kann. Repräsentiert der Graph etwa die Kommunikationswege in einer Organisation, sind Knoten mit hoher *closeness*-Zentralität besonders effektiv im Verbreiten von Informationen, weil sie viele Knoten über wenige Zwischenschritte erreichen können.

Ein Problem kann bei der Berechnung der *closeness*-Zentralität auftreten, wenn der Graph nicht vollständig oder gerichtet ist. In beiden Fällen kann es vorkommen, dass ein Knoten $v$ von einem Knoten $u$ nicht erreicht werden kann. Dann ist $d(u, v) = \infty$ und die obige Gleichung genau genommen unbestimmt. In diesen Fällen wird nur der *Einflussbereich* eines Knotens zur Berechnung herangezogen. Damit ist die Menge der Knoten gemeint, die von $u$ ausgehend in einer endlichen Anzahl von Schritten erreicht werden kann. Mit $J_u$ als der Anzahl der Knoten im Einflussbereich von $u$ kann die obige Gleichung umformuliert werden zu

$$C^*_c(u) = \frac{J_u/(g-1)}{\sum d(u, v)/J_u}.$$

Beschrieben wird damit, wie „nah“ ein Knoten $u$ den Knoten in seinem Einflussbereich ist.

Die dritte wichtige Variante von Zentralität, die hier vorgestellt werden soll, heißt *betweenness*-Zentralität. Sie drückt aus, wie wichtig ein Knoten $u$ für die Verbindung aller anderen Knoten untereinander ist. Demnach nehmen diejenigen Knoten eine hohe *betweenness*-Zentralität ein, die auf vielen Pfaden zwischen zwei anderen Knoten liegen. Dieses Maß gibt also an, wie sehr ein Knoten „zwischen“ allen anderen Knoten liegt.

Berechnet wird die *betweenness*-Zentralität eines Knotens $u$ mit Hilfe der Anzahl aller möglichen Pfade $g_{vw}$, über die die beiden Knoten $v$ und $w$ miteinander verbunden sind. Diejenigen dieser Pfade, die den Knoten $u$ mit einschließen, werden als $g_{vw}(u)$ bezeichnet. Unter der Annahme, dass alle Pfade mit gleicher Wahrscheinlichkeit ausgewählt werden, wird die *betweenness*-Zentralität $C_b$ eines Knotens $u$ bestimmt als

$$C_b(u) = \sum_{v \neq u \neq w} \frac{g_{vw}(u)}{g_{vw}}.$$

Die größtmögliche *betweenness*-Zentralität $(g-1)(g-2)/2$ wird erreicht, wenn $u$ auf allen Pfaden zwischen allen Paaren von insgesamt $g$ Knoten liegt, die $u$ nicht einschließen. Daher wird auch hier häufig die normalisierte Form angegeben, um so eine Vergleichbarkeit zu anderen Graphen herzustellen:

$$C'_b(u) = C_b(u) / \left[(g-1)(g-2)/2\right].$$

In gerichteten Graphen ist die größtmögliche *betweenness*-Zentralität $(g-1)(g-2)$. Die obige Gleichung für $C'_b(u)$ muss also nur im Nenner angepasst werden.

Die *betweenness*-Zentralität eines Knotens kann als Kontrolle interpretiert werden, die ein Knoten auf die Interaktionen der anderen Knoten untereinander ausüben kann. Je mehr der Interaktionswege zweier Knoten $v$ und $w$ über den Knoten $u$ laufen, desto mehr kann $u$ beeinflussen, wie oder ob $v$ und $w$ miteinander interagieren können.

Im Gegensatz zur *closeness*-Zentralität kann die *betweenness*-Zentralität auch für Graphen berechnet werden, die nicht zusammenhängend sind.

Für manche Analysen, insbesondere im Zusammenhang mit sozialen Strukturen, sind die eingehenden Kanten eines Knotens von größerer Bedeutung als ausgehende. Je mehr Interaktionspfade bei einem Knoten $u$ enden, desto wichtiger scheint er für die Knoten zu sein, die eine Interaktion mit $u$ initiieren. Um speziell diese Bedeutung auszudrücken, gibt es die Notation des *Prestige*. Sie ist ausschließlich in gerichteten Graphen definiert.

Die simpelste Form des Prestige basiert ebenso wie die Grad-Zentralität auf dem Eingangsgrad eines Knotens. Ist der Eingangsgrad von $u$ durch $d_I(u)$ gegeben, ist das Grad-Prestige $P_d(u)$ von $u$ definiert als $P_d(u) = d_I(u)$.

Die normalisierte Form des Prestige

$$P'_d(u) = \frac{d_I(u)}{g-1}$$

ist unabhängig von der Anzahl $g$ der Knoten im Graph und wird häufig als relativer Eingangsgrad bezeichnet. Diese Form bezeichnet den Anteil der Knoten, die $u$ „auswählen". Interpretiert wird dieses Maß deshalb häufig als Wichtigkeit eines Knotens $u$ für dessen direkte Nachbarn.

Um das Prestige eines Knotens innerhalb seines Einflussbereichs auszudrücken, kann das sogenannte *proximity*-Prestige berechnet werden. Anders als für die *closeness*-Zentralität wird der Einflussbereich eines Knotens $u$ über die Knoten definiert, von denen *ausgehend* eine Verbindung *zu* dem Knoten $u$ existiert. Für die Größe $I_u$ des Einflussbereichs von $u$ werden also die Knoten $v$ berücksichtigt, für die die Distanz $d(v,u) \neq \infty$ und damit finit ist. Das proximity-Prestige $P_P(u)$ eines Knotens $u$ wird dann definiert als

$$P_P(u) = \frac{I_u/(g-1)}{\sum d(v,u)/I_u}.$$

Sowohl für die Zentralität als auch das Prestige gibt es in der einschlägigen Literatur noch einige weitere Varianten. Dort werden zum Beispiel die Informations-Zentralität oder das Rang-Prestige genannt [129]. Für die folgenden Betrachtungen und insbesondere das in 5.1.2 beschriebene Experiment sollen die hier vorgestellten Maße aber genügen. Alleine mit diesen Maßen können aus den relationalen Eigenschaften einzelner Knoten bereits Rückschlüsse auf deren Wichtigkeit, Eigenschaften und Rollen in dem jeweiligen sozialen Geflecht gezogen werden. Akteure, die effizient kommunizieren stechen ebenso heraus wie solche, die Informationswege in einer sozialen Struktur kontrollieren. Darüber hinaus können auch strukturelle Eigenschaften des ganzen Netzes benannt werden. So wird aus den hier vorgestellten Maßen schnell deutlich, wie viele wichtige Akteure es in einem sozialen Geflecht gibt, wie viele Akteure isoliert werden und bis zu welchem Grad solche Außenseiter isoliert werden. Im Folgenden sollen nun weitere strukturelle Eigenschaften von Graphen benannt werden.

#### 5.1.1.4 Cliquen und Cluster

Für die Betrachtung sozialer Gruppen ist häufig die Betrachtung von Teilgruppen interessant. Gibt es in einem sozialen Geflecht Gruppen, die sich von anderen Gruppen abgrenzen? Wie sehr sind die Mitglieder dieser Gruppen untereinander verbunden? Wieviele Verbindungen bestehen zu Mitgliedern anderer Gruppen?

Eine Gruppe von Akteuren, die besonders stark untereinander verbunden sind, nennt man *Clique*. Eine Clique $C$ ist eine Teilmenge $C \subseteq V$ der Knoten eines ungerichteten Graphen, in der alle Knoten paarweise miteinander

verbunden sind. Existiert im Graphen zudem kein weiterer Knoten, der mit allen Mitgliedern der Clique verbunden ist, heißt eine Clique *maximal*. Eine maximale Clique ist also ein maximaler vollständiger Teilgraph und damit die am strengsten definierte Art einer Teilgruppe. In sozialwissenschaftlichen Betrachtungen wird zudem gefordert, dass eine Clique aus mindestens drei Knoten besteht, um triviale Cliquen aus zwei Knoten und sogenannte Dyaden aus dieser Definition auszuschließen. Die Definition einer Clique lässt sich auf gerichtete Graphen übertragen.

Weniger streng definiert sind *Komponenten*. Eine Komponente ist auch ein maximaler verbundener Teilgraph – allerdings ohne die Einschränkung, dass alle Knoten dieses Teilgraphen miteinander verbunden sind. Für Komponenten genügt die Erreichbarkeit, also Pfade beliebiger, endlicher Länge zwischen allen Knoten in einer Komponente. Falls nicht alle Knoten einer Komponente füreinander erreichbar sind, zerfällt der Graph in mehrere disjunkte Komponenten.

Als Maß für die Cliquenbildung in einem Graphen dient vor allem der *Cluster-Koeffizient*. Dieser kann sowohl lokal für jeden Knoten als auch global als Eigenschaft des gesamten Graphen bestimmt werden. In einem ungerichteten Graphen wird der lokale Cluster-Koeffizient $C(u)$ eines Knotens $u$ berechnet aus der Anzahl der Kanten $n$, die zwischen den $g_u$ Nachbarn von $u$ tatsächlich verlaufen und den $\frac{1}{2}g_u(g_u - 1)$ Kanten, die zwischen den $g_u$ Nachbarn maximal verlaufen könnten:

$$C_l(u) = \frac{2n}{g_u(g_u - 1)}.$$

In einem gerichteten Graphen entfällt der Faktor 2, da zwischen den Nachbarn $g_u$ eines Knotens maximal $g_u(g_u - 1)$ Kanten verlaufen können. Der Cluster-Koeffizient wird häufig auch definiert als

$$C_l(u) = \frac{\text{Zahl der Dreiecke mit } u}{\text{Zahl der Tripel (Pfade der Länge 2) mit } u \text{ als zentralem Knoten}}.$$

Analog dazu lautet die Definition des globalen Cluster-Koeffizienten

$$C_g(u) = \frac{3 \cdot \text{Zahl der geschlossenen Dreiecke}}{\text{Zahl der zusammenhängenden Dreiergruppen}}.$$

Der Cluster-Koeffizient liefert so also ein Maß für die Dichte von Dreiecken im Umfeld eines Knotens oder im gesamten Graphen. Er dient damit auch als Indiz für den Typ eines Netzwerks. In Sozialen Netzen ist der Cluster-Koeffizient häufig in einem zweistelligen Bereich, wohingegen das Netz der Autonomen Systeme im Internet einen Cluster-Koeffizienten von circa $C = 0,01$ aufweist.

#### 5.1.1.5 Zusammenfassung

Mit den hier vorgestellten Maßen aus der Graphentheorie sind strukturelle Analysen von Graphen möglich. Im Zusammenhang Sozialer Netze lassen einige dieser Maße auch Rückschlüsse auf die Eigenschaften und Rollen von Akteuren innerhalb einer sozialen Gruppe zu.

Diese Maße können auch dazu eingesetzt werden, um Graphen miteinander zu vergleichen. Im folgenden Abschnitt wird eine solche vergleichende Analyse mit den hier vorgestellten Maßen für ein beispielhaftes kontextzentrisches Soziales Netz durchgeführt.

### 5.1.2 Simulation eines kontextzentrischen Sozialen Netzes

Die große Stärke der kontextzentrischen Vernetzung, nämlich die typlose und universelle Kontextbeschreibung, soll auch eine direkte Kommunikation ermöglichen. Die persönliche Kommunikation zwischen denselben Nutzern kann in verschiedenen Kontexten und in unterschiedlichen sozialen Konstellationen womöglich unterschiedlich adressiert werden. Trotzdem soll der Nachrichtenaustausch unter den „richtigen“ Kommunikationspartnern passieren.

Um dies zeigen zu können, dass nämlich mit dem kontextzentrischen Aufbau eines Sozialen Netzes ein natürlich strukturierter Sozialer Graph zwischen den Akteuren des Netzes entstehen kann, wurden die folgenden Untersuchungen auf einem öffentlich zugänglichen Datensatz eines Sozialen Netzes aufgebaut – dem Enron eMail-Korpus. Das ist ein Datensatz, der die eMail-Kommunikation von etwa 150 Mitarbeitern der Firma Enron enthält. Die eMails wurden im Zuge von Ermittlungen der amerikanischen Behörde *Federal Energy Regulatory Commission* gesammelt, die 2002 wegen des Bankrotts von Enron initiiert wurden. Bis zu dem Zeitpunkt war das immerhin der größte Bankrott der amerikanischen Geschichte, weil Besitzstände im Wert von über 63 Milliarden US-Dollar reorganisiert werden mussten (dieser zweifelhafte Rekord wurde aber bereits im darauffolgenden Jahr von der Pleite des Telekommunikationsanbieters WorldCom gebrochen). Die bei den Ermittlungen gesammelten eMails wurden der Öffentlichkeit zu wissenschaftlichen Zwecken zur Verfügung gestellt. Der Datensatz erfreut sich seitdem großer Beliebtheit, da er unter natürlichen Bedingungen entstanden ist, echte Soziale Strukturen abbildet und eine große Fülle an Untersuchungen zum Fluss von Informationen, dem Entstehen und der Verbreitung von Wissen und Entwicklungen zwischenmenschlicher Beziehungen zulässt. So findet sich in dem Datensatz auch viel private Kommunikation: Erzählungen von Urlaubsreisen oder unterhaltsame Rezepte für beschwipste Weihnachtskuchen(bäcker) sind unverändert in dem Datensatz zu finden.

Aus diesem eMail-Verkehr lässt sich ein Sozialer Graph extrahieren, der eine natürliche soziale Struktur repräsentiert. Zu diesem Sozialen Graphen wurden bereits zahlreiche Untersuchungen durchgeführt. Zum Beispiel wurde

analysiert, wie sich die Struktur der Sozialen Gruppen während der Krise der Firma entwickelt haben. Allein mit den im vorherigen Abschnitt vorgestellten Metriken aus der Graphentheorie konnte gezeigt werden, dass die leitenden Angestellten zu einer engen Clique zusammengewachsen sind, während Enron auf den Konkurs zusteuerte [134], [135].

Mit dem Sozialen Graphen aus dem Enron-Korpus steht eine umfangreiche Grundwahrheit für ein Soziales Netzwerk bereit. Die Untersuchungen, die im Rahmen dieser Arbeit angestellt wurden, drehten sich daher um die Frage, wie sich der Soziale Graph entwickeln würde, und ob er mit dieser Grundwahrheit in Größe und Struktur vergleichbar wäre, wenn sich die Akteure im Netz nicht per eMail gegenseitig anschreiben könnten, sondern kontextzentrisch miteinander verknüpft wären.

Für die folgenden Experimente wurde deshalb für jeden Akteur aus dem Datensatz eine Kontextbeschreibung formuliert. Die Kontextbeschreibungen aller Akteure wurden dann paarweise miteinander verglichen. Waren sich die Beschreibungen ähnlicher als ein zuvor festgelegter Grenzwert, wurde zwischen den Akteuren eine Kante im Sozialen Graphen eingefügt. Die Kontextbeschreibung selbst muss natürlich vollständig in dem Datensatz enthalten sein, sodass keine externen Faktoren die Entstehung des Sozialen Graphen beeinflussen können. Im Kern entspricht dieses Vorgehen der tatsächlichen kontextzentrischen Vernetzung. Die Kontextbeschreibungen werden in Form von Bloom-Filtern verglichen; Nachrichten werden nur weitergeleitet, wenn die Ähnlichkeit der Filter groß genug ist.

Der Kontext wurde hier über die Kommunikationspartner der letzten empfangenen und gesendeten eMails beschrieben. Die Kontextbeschreibung eines Akteurs bestand also aus einer Liste von eMail-Adressen. Die Liste hatte eine begrenzte Länge, sodass nur die neuesten Kommunikationspartner berücksichtigt wurden.

Mit einer solchen Kontextbeschreibung kann ein direkter Nachrichtenaustausch natürlich erstmal nicht realisiert werden. Dazu müsste der Kontext über Themen oder Schlagwörter aus den einzelnen eMail-Kommunikationen konstruiert werden. Damit läge der Fokus aber beim Mining von Themen aus Texten, um damit den eMail-Austausch möglichst zuverlässig abbilden zu können. Hier soll vielmehr gezeigt werden, dass aus einer Menge von Kontextbeschreibungen ein natürlich ausgeprägter Sozialer Graph gewonnen werden kann, in dem zentrale Rollen von denselben Akteuren eingenommen werden. Das hier beschriebene Vorgehen entspricht also eher einer Metaanalyse eines kontextzentrischen Netzes: Darin würden alle bekannten Kontextbeschreibungen einer Zeitspanne zusammengefasst und ebenfalls paarweise miteinander verglichen. So könnte eine globale Sicht auf das Netzwerk über alle Kontexte hinweg entstehen und zentrale Teilnehmer des Netzes kontextübergreifend identifiziert werden. Über dieses Vorgehen ist also erkennbar, welche Akteure wie stark in dieser sozialen Gruppe auftreten. Die eMail-Adresse eines aktuellen Kommunikationspartners entspricht daher einem kontextabhängigen Kom-

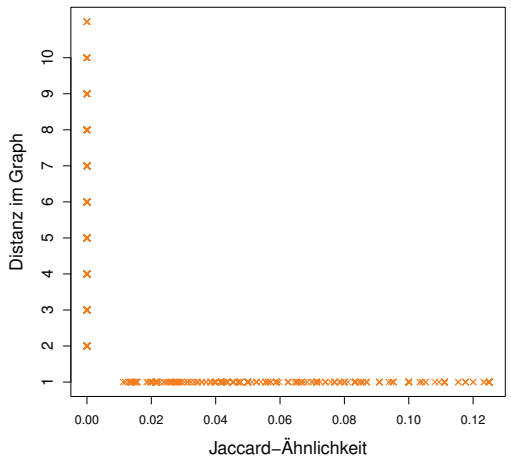
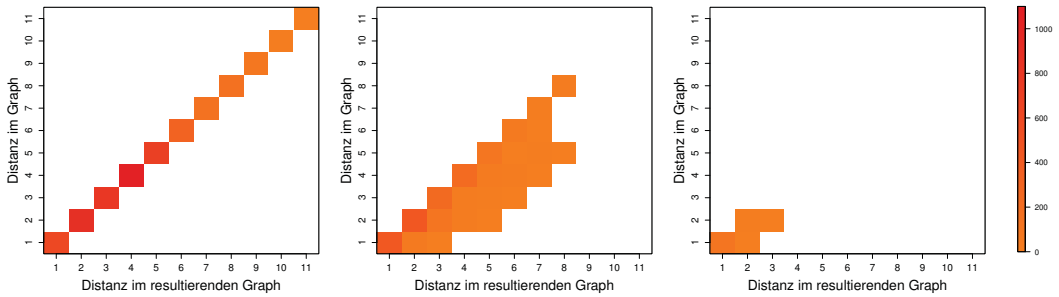



Abbildung 5.1: Zusammenhang zwischen Jaccard-Ähnlichkeiten der Kontextbeschreibungen und Distanz im sozialen Graphen der Grundwahrheit

munikationskanal, und die Kontextbeschreibung mit mehreren eMail-Adressen entspricht der Zusammenfassung mehrer Kontexte – die Untersuchungen stellen also eine Metaanalyse über mehrere Kommunikationskanäle dar.

Die folgenden Untersuchungen basieren auf der Kommunikation aus einem Zeitraum von 7 Tagen im Oktober 2001. In diesem Zeitraum wurde der Enron Skandal bekannt; unter den Mitarbeitern wurden deshalb in dieser Zeit besonders viele eMails ausgetauscht [134]. Insgesamt enthielt der Datensatz für die 7 Tage zwischen Montag, dem 8. Oktober 2001 ab Tagesbeginn (0 Uhr), und Sonntag, dem 14. Oktober 2001 bis Tagesende (23 : 59 Uhr und 59 Sekunden), 1534 eMails. Diese eMails wurden von 32 Mitarbeitern verschickt; das entspricht etwa 48 eMails pro Nutzer für den angegebenen Zeitraum, also etwa 7 eMails pro Nutzer pro Tag. Der überwiegende Teil der Akteure, die in den eMails in diesem Zeitraum in Erscheinung traten, waren isolierte Knoten am Rande des Sozialen Graphen. Sie haben entweder selbst keine eMails versendet oder wurden in dem Zeitraum der Untersuchung mit nur einer einzigen eMail adressiert. Um die Untersuchungen auf die wichtigen Akteure zu fokussieren, wurden daher alle eMail-Adressen ignoriert, die an weniger als 2 Kommunikationen beteiligt waren. So blieben für diesen Zeitraum 97 Akteure übrig, die entweder 2 oder mehr Nachrichten empfangen oder selbst gesendet haben.

Für diese 97 Akteure wurden nun die Empfänger und Absender der aus- und eingehenden eMails aus dem Untersuchungszeitraum gesammelt. Daraus wurden die Kontextbeschreibungen der einzelnen Akteure formuliert und paarweise miteinander verglichen. Lag die Jaccard-Ähnlichkeit zweier Beschreibungen über einem bestimmten Grenzwert, wurde eine Kante zwischen den beteiligten Akteuren im Sozialen Graphen eingefügt. Die Akteure selbst wurden als Knoten erst mit der ersten Kante in den Sozialen Graphen eingefügt – der resultierende Graph enthält also keine isolierten, sondern allenfalls weniger Knoten.




(a) Grenzwert 0,01 (b) Grenzwert 0,05 (c) Grenzwert 0,115

Abbildung 5.2: Analyse der Knotendistanzen in den resultierenden Sozialen Graphen für unterschiedliche untere Grenzwerte der Jaccard-Ähnlichkeiten

Der Grenzwert der Jaccard-Ähnlichkeit wurde schrittweise um $0,005$ von $0,0$ bis $1,0$ erhöht. Für jeden dieser Grenzwerte wurde ein eigener Sozialer Graph konstruiert und analysiert. Ab einem Grenzwert von $0,25$ blieben zu wenige Kanten und Knoten für eine sinnvolle Analyse übrig, sodass die Auswertungen sich nur auf Grenzwerte bis $0,25$ beziehen.

Für die ermittelten 97 Akteure wurde zusätzlich ein Sozialer Graph aus der Grundwahrheit konstruiert. Dazu wurden alle eMails, die diese Akteure in dem Untersuchungszeitraum untereinander ausgetauscht haben, als Kanten in den Sozialen Graphen eingefügt. Bis auf Abbildung 5.1 enthalten alle hier abgebildeten Grafiken die entsprechende Auswertung dieses Graphen zum Vergleich. In der Abbildung 5.1 sind die Jaccard-Ähnlichkeiten zwischen den Kontextbeschreibungen zweier Knoten im Vergleich zur Distanz zwischen diesen Knoten im Graph der Grundwahrheit abgebildet. Es ist deutlich erkennbar, dass die Kontextbeschreibungen von unverbundenen Knoten keine oder nur sehr geringe Ähnlichkeiten aufweisen. Die Jaccard-Ähnlichkeiten der Beschreibungen verbundener Knoten lassen sich davon zudem gut trennen. Für eine Distanz von mehr als 2 – also für nicht direkt miteinander verbundene Knoten – sind nur Jaccard-Ähnlichkeiten von ungefähr $0,0$ aufgetreten, während die Ähnlichkeiten für eine Graph-Distanz von 1 alle größer waren als $0,01$. Das zeigt, dass die Konstruktion der Kontextbeschreibung als eine Art Metaanalyse der Kommunikationen prinzipiell geeignet ist, den Sozialen Graphen nachzubilden – schließlich sind die Jaccard-Ähnlichkeiten verbundener und unverbundener Knoten deutlich voneinander getrennt.

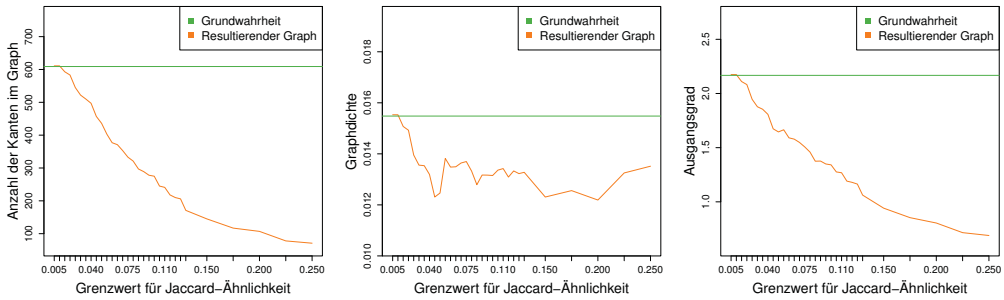
Für die folgenden Untersuchungen wurden nun die Kontextbeschreibungen aller Akteure paarweise miteinander verglichen, um aus diesem Vergleich einen Sozialen Graphen aufbauen zu können. Variiert wurden jeweils die unteren Grenzwerte der Jaccard-Ähnlichkeiten, die zwischen zwei Beschreibungen mindestens bestehen mussten, damit eine Kante im Graphen eingefügt würde. Die resultierenen Graphen wurden dann jeweils mit dem Graphen verglichen, der direkt aus dem Datensatz extrahiert wurde. Abbildung 5.2 zeigt nun einige

Analysen von Knotendistanzen, für die die Distanz zwischen zwei Akteuren im tatsächlichen Graphen mit der Distanz im resultierenden Graphen verglichen wurde. Die analysierten Graphen sind mit unteren Grenzwerten für Jaccard-Ähnlichkeiten von $0,01$, $0,05$ und $0,1$ entstanden. Der Graph in Abbildung 5.2(a) scheint dem Graphen der Grundwahrheit am Ähnlichsten – die verglichenen Knotendistanzen bewegen sich hier alle entlang der Diagonalen. In der mittleren Abbildung scheint der Graph sich schon deutlich anders ausgeprägt zu haben. Der höhere Jaccard-Grenzwert von $0,05$ hat dazu geführt, dass einige Kanten im resultierenden Graphen fehlen. Das führt dazu, dass die gemessenen Knotendistanzen stellenweise höher sind als im eigentlichen Graphen und zu höheren Knotendistanzen unterhalb der Diagonale hin abweichen. Es ist zudem erkennbar, dass der resultierende Graph aus deutlich weniger Kanten besteht, als der aus Abbildung 5.2(a). Diese beiden Effekte – höhere gemessene Knotendistanzen und insgesamt weniger Kanten – sind im dritten Graphen in Abbildung 5.2(c) noch einmal deutlich stärker ausgeprägt zu erkennen.

Abbildung 5.3 zeigt nun eine graphtheoretische Auswertung der resultierenden Sozialen Graphen jeweils im Vergleich mit der Grundwahrheit als horizontaler grüner Linie. Die Grafik in 5.3(a) zeigt die Anzahl der Kanten in den Sozialen Graphen für unterschiedliche untere Grenzwerte der Jaccard-Ähnlichkeiten. Es zeigt sich, dass mit Grenzwerten zwischen $0,005$ und $0,01$ eine zur Grundwahrheit ähnliche Anzahl an Kanten in den jeweiligen Sozialen Graphen etabliert wurden. Mit steigenden Grenzwerten blieben immer weniger zueinander ähnliche Kontextbeschreibungen übrig, sodass die entsprechenden Graphen zunehmend weniger Kanten enthielten.

Mit den Grenzwerten zwischen $0,005$ und $0,01$ enthielt der entsprechende Graph auch eine zur Grundwahrheit ähnliche Graphdichte, wie die Grafik 5.3(b) zeigt. Die Dichte $d(G)$ eines Graphen $G = (V, E)$ ist gegeben als $d(G) = 2\,|E|\,/(|V|\,(|V| - 1))$ und beschreibt das Verhältnis zwischen den tatsächlich vorhandenen und potenziell möglichen Kanten, indem die Anzahl der Knoten $|E|$ mit der Anzahl der vorhandenen Kanten $|V|$ ins Verhältnis gesetzt wird. Da die ermittelte Dichte zwischen $0,005$ und $0,01$ ebenso wie die Anzahl der Kanten zur Grundwahrheit ähnlich ist, heißt das, dass für diese Grenzwerte auch die Anzahl der Knoten vergleichbar zur Grundwahrheit ist. Für steigende Grenzwerte weicht die Dichte allerdings zunehmend ab, weil neben immer weniger Kanten auch immer weniger Knoten in den Graphen enthalten sind.

Die Analyse der durchschnittlichen Ausgangsgrade der Knoten in Grafik 5.3(c) zeigt, dass mit Grenzwerten für die nötige Jaccard-Ähnlichkeit zwischen $0,005$ und $0,01$ wiederum ein der Grundwahrheit ähnlicher Wert erreicht wurde, während die Ausgangsgrade für steigende Grenzwerte im Schnitt abnehmen. Daraus kann geschlossen werden, dass der Soziale Graph für die genannten Grenzwerte insgesamt gut angenähert werden konnte: Die Anzahl der Kanten und Knoten entspricht nahezu der tatsächlichen Anzahl und die




(a) Anzahl der Kanten (b) Graph-Dichte (c) Durchschnittliche Ausgangsgrade

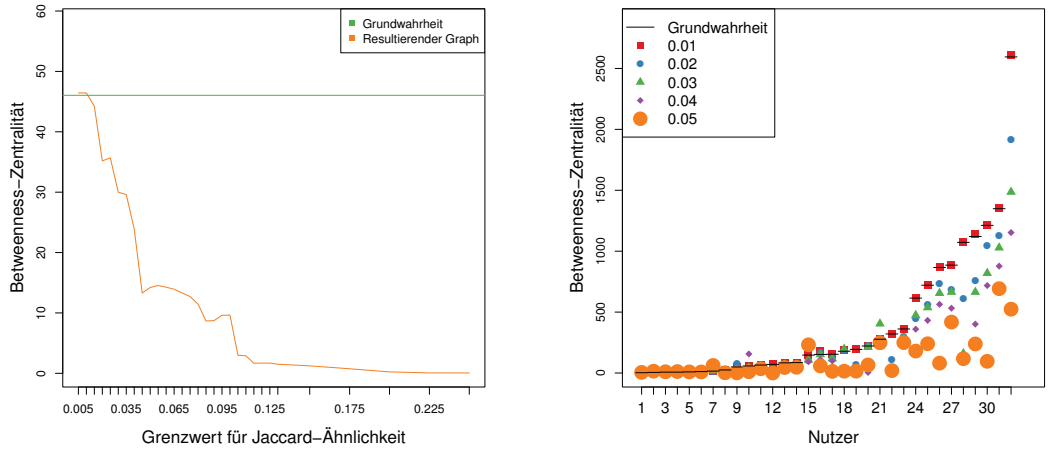
Abbildung 5.3: Analyse der resultierenden Sozialen Graphen für unterschiedliche untere Grenzwerte der Jaccard-Ähnlichkeiten

Anzahl der ausgehenden Kanten pro Knoten ist im Durchschnitt auch vergleichbar. Es sieht also so aus, als seien die zugelassenen Kanten auch zwischen den „richtigen“ Knoten gefunden worden. Diese Vermutung kam auch schon in den Betrachtungen zu Grafik 5.2(a) auf. In dem hier simulierten kontextzentrischen Netz sollten jedenfalls auch dieselben Knoten an zentralen Stellen zu finden sein, was durch die folgenden Analysen näher untersucht wurde.

Bestimmt wurden für dieses kontextzentrische Netz die *betweenness*- und die *closeness*-Zentralitäten sowie die Cluster-Koeffizienten aller Knoten. Es zeigte sich zunächst, dass die *closeness*-Zentralitäten für die hier betrachtete Gruppe der 97 Akteure nicht bestimmt werden konnten, da das die Beschaffenheit des Graphen nicht zuließ. Da für die *closeness*-Zentralität eines Knotens die kürzesten Pfade zu allen anderen Knoten im Graph bestimmt werden müssen, bedeutet das auch, dass jeder Knoten jeden anderen im Netz erreichen können muss. Das war in diesem Szenario weder im Graph der Grundwahrheit, noch in den experimentell aufgebauten Graphen der Fall. Daher werden hier nur Auswertungen zu *betweenness*-Zentralität und Cluster-Koeffizienten beschrieben.

Für diese Untersuchungen wurden nur die 32 Akteure betrachtet, die selbst auch eMails versendet haben; da hier stets gerichtete Graphen erzeugt und analysiert wurden, konnten auch nur Akteure mit ausgehenden Kanten auf dem Pfad zwischen zwei anderen Akteuren liegen. Diejenigen Akteure, die ausschließlich eMails empfangen haben, konnten demgegenüber nur an Endpunkten von Pfaden liegen.

Abbildung 5.4 zeigt die Auswertung der *betweenness*-Zentralitäten. Dabei enthält die linke Grafik die durchschnittlichen *betweenness*-Zentralitäten aller Knoten über steigende untere Grenzwerte der Jaccard-Ähnlichkeit. Für diese Grafik wurden also die *betweenness*-Zentralitäten aller Knoten für jeden entstandenen Graphen bestimmt und die durchschnittliche Zenträlität abgetragen. Es ist erkennbar, dass die durchschnittliche *betweenness*-Zentralitäten für Jaccard-Ähnlichkeiten von $0,005$ und $0,01$ nahe an der Grundwahrheit liegen



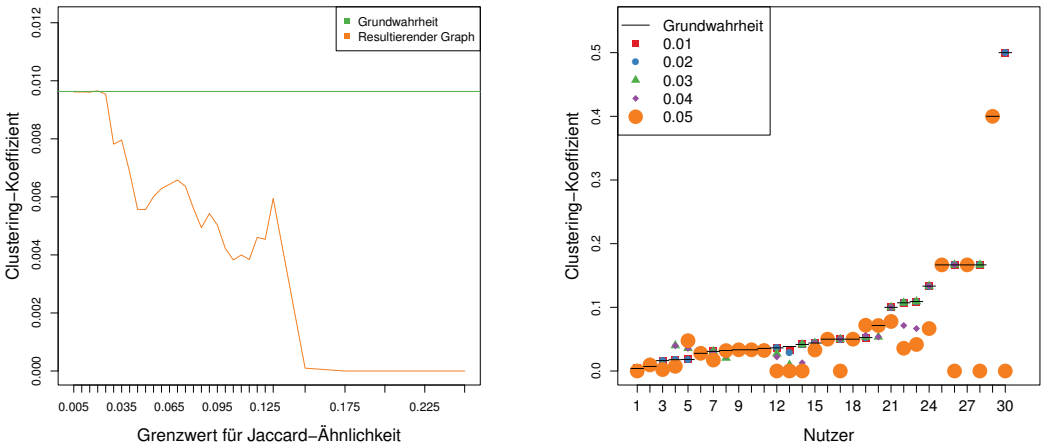

(a) *Betweenness*-Zentralität im Durchschnitt

(b) *Betweenness*-Zentralitäten aller Knoten

Abbildung 5.4: Analyse der *betweenness*-Zentralität aller Knoten für unterschiedliche untere Grenzwerte der Jaccard-Ähnlichkeiten

und für steigende Grenzwerte zunehmen abweichen. Die Vermutung scheint also bestätigt, dass in den Graphen, die mit den genannten Grenzwerten erzeugt wurden, dieselben Knoten an denselben Positionen im Graph erkannt werden. Die Abbildung 5.4(b) unterstreicht dies. In der Grafik sind die *betweenness*-Zentralitäten der 32 „aktiven“ Knoten abgetragen. Diese Knoten liegen im tatsächlichen Graph also auf zahlreichen kürzesten Pfaden zwischen anderen Knoten – die Kommunikation im Unternehmen verlief häufig über diese Akteure. Die Grundwahrheit, die aus dem tatsächlichen Sozialen Graph erstellt wurde, ist in diesem Fall durch einen schwarzen horizontalen Strich erkennbar; die unterschiedlichen Symbole repräsentieren die *betweenness*-Zentralität eines Knotens im jeweils resultierenden Graphen für unterschiedliche Grenzwerte der Jaccard-Ähnlichkeiten. Die Grafik zeigt zunächst, dass die Zentralitäten der Knoten auch über die kontextzentrische Vernetzung erkannt werden konnte – insbesondere für den Grenzwert der Jaccard-Ähnlichkeit von $0,01$. Zum einen wurden die genannten Akteure in diesen Graphen als wichtige Akteure identifiziert. Zum anderen konnte die Struktur des Netzes mit diesen Jaccard-Grenzwerten so weit abgebildet werden, dass die Zentralitäten dieser Akteure den tatsächlichen Zentralitäten angenähert werden konnten. Für steigende untere Grenzwerte der Jaccard-Ähnlichkeit wichen die Zentralitäten allerdings zunehmend stärker ab.

Vergleichbares zeigt sich für die Clustering-Koeffizienten der Knoten. Dazu sind in den Abbildungen 5.5(a) und 5.5(b) die durschnittlichen Clustering-Koeffizienten der jeweiligen Sozialen Graphen, beziehungsweise die Clustering-Koeffizienten einzelner Knoten für unterschiedliche Grenzwerte der Jaccard-Ähnlichkeit abgetragen. Es ist erkennbar, dass die durchschnittlichen Clustering-Koeffizienten für die Jaccard-Ähnlichkeiten von $0,005$ und $0,01$ den




(a) Clustering-Koeffizienten im Durschnitt (b) Clustering-Koeffizienten aller Knoten

Abbildung 5.5: Analyse der Clustering-Koeffizienten aller Knoten für unterschiedliche untere Grenzwerte der Jaccard-Ähnlichkeiten

tatsächlichen Graphen wieder sehr gut annähern. In diesem Fall konnten sogar die Graphen, die mit Grenzwerten von bis zu $0,02$ erzeugt wurden, die Grundwahrheit gut annähern. Das wird in der Abbildung 5.5(b) nochmals verdeutlicht, in der die Koeffizienten zu den einzelnen Knoten abgetragen wurden. Hier ist nun erkennbar, dass in den Graphen der Grenzwerte bis einschließlich $0,02$ kaum wirkliche Abweichungen von der Grundwahrheit auftraten. Lediglich die Clique(n) von Nutzer 13 scheinen nicht vollständig rekonstruiert worden zu sein. Stärkere Abweichungen sind zudem nur für den Graphen des Grenzwerts $0,05$ erkennbar. Diese Ergebnisse zeigen zum einen, dass die Cliquen und Cluster auch durch die kontextzentrische Vernetzung gut wiedergefunden werden können. Darüber hinaus entsteht die Vermutung, dass zahlreiche der untersuchten eMails, die von diesen 32 aktiven Nutzern gesendet wurden, auch unter diesen Nutzern untereinander ausgetauscht wurden. Denn während in den Graphen, die mit Jaccard-Grenzwerten von $>= 0,04$ konstruiert wurden, schon deutlich weniger Kanten und Knoten enthalten waren (siehe Abbildung 5.3), bleiben die Clustering-Koeffizienten bis zu diesem Grenzwert weitgehend stabil.

Insgesamt zeigen die hier beschriebenen Experimente, dass in kontextzentrischen Sozialen Netzen herkömmliche Analysen Sozialer Netze angewendet werden können und verwertbare Ergebnisse liefern. In diesem Fall wurde ein eMail-Netzwerk mit kontextzentrischer Struktur so abgebildet, wie für eine Metaanalyse eines echten kontextzentrischen Kommunikationsnetzes. Die Kontextbeschreibungen der Akteure waren in diesem Fall nicht zur direkten Kommunikation geeignet, sondern entsprachen einer Zusammenfassung oder einer Historie von Kontexten. Zwischen diesen Beschreibungen wurden Jaccard-Ähnlichkeiten bestimmt, um damit einen Sozialen Graphen zu erzeugen. Es zeigte sich, dass der tatsächliche Soziale Graph strukturell gut abgebildet wer-

den kann und die zentralen Akteure auch im kontextzentrischen Netz zentrale Rollen einnehmen.

An dieser Stelle bieten sich weitere Untersuchungen zur geeigneten Wahl der Kontextbeschreibungen an. So können mit Methoden des *Text Mining* etwa Themen aus den eMails bestimmt werden, über die dann Kontextbeschreibungen konstruiert werden, die zur persönlichen Kommunikation geeignet sind. Nachrichten werden dann zum Beispiel an das Thema „Konkurs“ adressiert werden. Indem diese Beschreibungen nur für begrenzte Zeitspannen gültig sind und durch weitere Kontextinformationen wie Abteilung oder Hierarchieebene angereichert werden, sollten Nachrichten ähnlich zielgerichtet ausgetauscht werden können.

Außerdem kann geklärt werden, ob die Mindestähnlichkeit zwischen Kontextbeschreibungen in Form eines Grenzwerts der Jaccard-Ähnlichkeit automatisch bestimmt werden könnte. Womöglich kann ein solcher Grenzwert aus der Menge von Kontextbeschreibungen mit zeitlich begrenzter Gültigkeit heraus bestimmt werden, weil beispielsweise die Kommunikation zeitlich besser nachvollzogen werden kann.

Insbesondere die Erkenntnis zum durchschnittlichen Clustering-Koeffizient aus Abbildung 5.5(a) scheinen aber zu bestätigen, dass diese Art der Metaanalyse dazu geeignet ist, die generelle Zugehörigkeit zu Cliquen zu ermitteln. Es wurde weiter oben festgestellt, dass der Clustering-Koeffizient bis zu einer höheren unteren Grenze der Jaccard-Ähnlichkeit stabiler und näher an der Grundwahrheit bleibt als die *betweenness*-Zentralität. Das ist auf die Konstruktion der Kontextbeschreibung zurückzuführen. Als Aggregation mehrerer Kontexte – eben einer Metaanalyse – bewirkt sie, dass Randknoten, die in wenigen Kontexten in Erscheinung getreten sind, bei steigender Grenze der Jaccard-Ähnlichkeit früher eliminiert werden. Deshalb ist die *betweenness*-Zentralität der aktiven Knoten auch empfindlicher gegenüber diesem Grenzwert: Mit steigender Grenze fehlen den resultierenden Graphen Knoten und damit auch kürzeste Pfade, auf denen die aktiven Knoten liegen könnten. Teilen diese Knoten aber mehrere Kontexte gleichzeitig miteinander, ändert sich die Konstellation ihrer Cliquen offenbar deutlich langsamer. Es liegt deshalb die Vermutung nahe, dass die Verbindungen unter den Knoten, die in mehreren Kontexten zusammen aktiv waren, besonders stark sind: Arbeitskollegen, die abends auch im gleichen Sport-Club aktiv sind, sind womöglich befreundet. Ihre Verbindung ist deshalb stärker als zu Personen, mit denen sie nur das Büro teilen.

Womöglich stellt der Vergleich von aggregierten Kontextbeschreibungen deshalb allgemein ein gutes Maß für die qualitative Bewertung von Beziehungen dar. Eine solche qualitative Bewertung auch der Intensität von Beziehungen fehlt der Sozialen Netzwerkanalyse und den graphentheoretischen Methoden häufig; der Zusammenhang zur hier beschriebenen Metaanalyse eröffnet deshalb spannende Perspektiven für anschließende Untersuchungen.

Nachdem dieser Abschnitt Analysemöglichkeiten Sozialer Netze vorgestellt hat, widmet sich der folgende Abschnitt einer anderen Form der Datenerhe-

bung. Darin geht es um *Data Mining* unter den Nutzern. Es werden Verfahren vorgestellt, die für die verteilte Architektur der kontextzentrischen Netze entwickelt wurden und Garantien an die Privatheit der erhobenen Nutzerdaten geben können.

## 5.2 Datenerhebung mit Garantien an die Privatsphäre

Neben der strukturellen Analyse eines Sozialen Netzes, die im vorangegangenen Abschnitt beschrieben wurde, sind auch statistische Erhebungen von Profilinformationen oder Nutzerdaten von Interesse – klassisches *Data Mining* also. Diese Erhebungen wären auch in der verteilten Architektur kontextzentrischer Netze trivial, wenn man eine vertrauenswürdige zentrale Entität etablieren würde, die üblicherweise *trusted third party*, vertrauensvolle dritte Partei, genannt wird. Das soll aber vermieden werden. Mit dieser zentralen Entität entstünde zum einen ein allwissender Knoten, dem ganz allein die Sammlung, Archivierung und Freigabe von Nutzerinformationen obläge. Damit wäre die dezentrale Haltung von Profilinformationen, die einen wesentlichen Vorteil einer verteilten Architektur für die Privatsphäre von Nutzern darstellt, obsolet – die Nutzer hätten dann keine Kontrolle mehr über die Weitergabe ihrer Daten. Zum anderen entstünde mit dieser zentralen Entität auch eine zentrale Schwachstelle für Angriffe auf das Netz und insbesondere auf die Daten der Nutzer.

Diese Tatsache gilt natürlich in kontextzentrischen Sozialen Netzen so, wie in ubiquitären und kontextsensitiven Systemen allgemein. In diesen kommt erschwerend hinzu, dass die Menge der produzierten Daten so weit zunehmen wird, dass eine zentrale Speicherung und Verarbeitung kaum mehr wirtschaftlich zu realisieren sein wird. Man denke dabei nur an die Vision vom Internet der Dinge (*Internet of Things*), nach der eine ungeahnte Vielzahl von unterschiedlichen Systemen und Geräten miteinander kommunizieren können wird, um Menschen ihren Alltag zu erleichtern [136]. Dabei wird eine ungeheure Datenmenge erzeugt, die es nötig macht, dass die datenverarbeitende Intelligenz verteilt und womöglich in die Endgeräte ausgelagert wird.

Durch die Verarbeitung von Kontextinformationen entstehen aber – sowohl im vorverarbeitenden Sensorknoten wie auch bei nachgelagerten Rechensystemen – höchstsensible Daten mit sehr privatem Bezug. Der Schutz der Privatsphäre sollte für kontextsensitive und kontextzentrische Systeme also höchste Priorität genießen. In diesem Abschnitt werden deshalb Verfahren vorgestellt, die Datenerhebung in verteilten Systemen ermöglichen und dabei den Schutz der Privatsphäre in den Vordergrund stellen. Diese Verfahren eignen sich für die Datenerhebung in kontextzentrischen Sozialen Netzen ebenso wie für verteilte Systeme allgemein – beispielsweise solche aus dem Internet der Dinge.

Für die folgenden Betrachtungen soll deshalb ganz allgemein von einem verteilten System ausgegangen werden, an das eine Anfrage nach einer Schätzung der Verteilung eines numerischen Wertes gestellt wird. Das verteilte System kann auch eine verteilte Datenbank sein, von der aggregierte sensible Daten abgerufen werden. Der numerische Wert wiederum kann ein Sensormesswert sein, eine Profilinformation wie das Alter der Nutzer oder die Antwort eines Nutzers auf eine Umfrage. Denkbar ist etwa, dass ein Dienst eine Umfrage an ein verteiltes System von Nutzern initiiert, die in numerischer Form beantwortet werden muss – etwa mit einem Wert auf einer Likert Skala.

In allen diesen Szenarien soll von einem Dienstanbieter ausgegegangen werden, der an dem aggregierten Gesamtergebnis interessiert ist – also an der durchschnittlich gemessenen Temperatur, der Altersverteilung der Nutzer oder eben dem Ergebnis der Umfrage mit Mittelwert und Standardabweichung. Gleichzeitig soll davon ausgegangen werden, dass die individuellen Werte privat bleiben sollen und die Nutzer womöglich selbst an dem Gesamtergebnis interessiert sind.

Mit diesem Abschnitt soll nun beantwortet werden, wie die Knoten des Netzes kollaborieren können, um eine Schätzung der gewünschten Verteilung zu ermitteln, ohne private Informationen preiszugeben. Dazu wird zunächst auf Kommunikationsprotokolle und Verfahren eingegangen, mit denen Informationen effizient in einem Netz verbreitet und ein Konses unter allen beteiligten Kommunikationspartnern hergestellt werden kann. Die effiziente Informationsverbreitung ist notwendig, um die Schätzung einer Verteilung möglichst schnell zu ermitteln. Ein Konsens wäre zudem sinnvoll, wenn allen Nutzern das Gesamtergebnis bekannt gegeben werden soll. Anschließend werden Methoden zum Schutz der Privatsphäre vorgestellt, die auch für großangelegte verteilte Systeme geeignet sind. Der Teil 5.2.3 beantwortet schließlich, wie Kollaboration und Konsens privatsphäreschonend bewerkstelligt werden können.

### 5.2.1 Konsensprotokolle für verteilte Systeme

Ein Konsens in einem verteilten System ist dann hergestellt, wenn alle Knoten dieselbe, einheitliche Sicht auf eine Information haben, zu der auch alle Knoten des Netzes beigetragen haben. Ein Konsensprotokoll regelt neben der Verbreitung von Informationen meist auch die Kooperation unter den Knoten. Oft werden mit solchen Protokollen globale Durschnittswerte von lokalen Messgrößen bestimmt. Solche Protokolle sind vor allem aus dem Bereich der drahtlosen Sensornetze (*Wireless Sensor Networks*) oder allgemeiner Multiagentensystemen bekannt, in denen phsyisch verteilte Knoten Informationen über ein Kommunikationsnetzwerk zu Kooperationszwecken austauschen. Ziel ist, dass nicht nur eine zentrale Entität (in diesen Netzen Senke genannt) eine Gesamtsicht auf gemessene Daten erhält, sondern alle Sensoren gleichermaßen. So können sich Sensoren im Netzwerk beispielsweise selbstständig kalibrieren, also eine lokale Abweichung vom globalen Durchschnitt ermitteln [137].

Zur Verbreitung von Informationen in Rechnernetzen und verteilten Systemen im Allgemeinen oder Sensornetzen im Speziellen gibt es eine Vielzahl von Routingprotokollen [138]. Im Hinblick auf Konsensprotokolle sind davon insbesondere Flooding und Gossiping interessant. Das Wort *flooding* lässt sich mit Überschwemmung übersetzen und beschreibt damit auch optimal den Ablauf des Protokolls: Datenpakete werden an alle verfügbaren Nachbarknoten weitergeleitet, außer an die, von denen das aktuelle Paket empfangen wurde. Damit ist sichergestellt, dass Informationen überall im Netz ankommen, während gleichzeitig einiges an überflüssiger Datenübertragung (*overhead*) in Kauf genommen wird. Demgegenüber reduziert Gossiping diese überflüssige Übertragung spürbar, indem Datenpakete an verfügbare Nachbarknoten nur mit einer gewissen Wahrscheinlichkeit weitergeleitet werden. Damit soll das unter Menschen hocheffiziente „Klatschen und Tratschen“ (*gossip* ist der englische Begriff für Tratschen) nachgebildet werden, bei dem zufällige Adressaten mit kleinen und häufig unvollständigen Bruchstücken von Informationen versorgt werden, an denen die Adressaten nicht zwingend interessiert sein müssen. Trotz reduzierter überflüssiger Übertragungen können Informationen auch mittels Gossiping gut im Netz verbreitet werden. Eine Kombination von Flooding und Gossiping wird in [139] als ein Mittelweg zwischen überflüssiger Übertragung und bestmöglicher Verbreitung von Informationen vorgeschlagen.

Konsensprotokolle und insbesondere solche, die einen globalen Durchschnitt bestimmen, setzen vor allem auf eine Kombination von Gossiping und Broadcasting, das heißt, es werden zufällige Nachbarknoten ausgewählt, an die dieselben Nachrichten simultan verschickt wird. Diese Protokolle laufen zudem meist rundenbasiert ab. Aus den in einer Runde empfangenen Daten berechnet ein Knoten dann einen Durchschnitt und gibt diesen an die nächsten Kommunikationspartner weiter. Ein Konsens wird damit fast immer erreicht, während die Anzahl an Übertragungen akzeptabel gering ist. Dieses Vorgehen ist in der einschlägigen Literatur deshalb verbreitet und akzeptiert [140]–[145]. In weiteren Arbeiten wurden unter anderem Einschränkungen der erlaubten Übertragungen pro Runde untersucht und insbesondere deren Auswirkungen auf die Abschätzung und Konsensbildung über zeitlich variierenden Signale – damit eignen sich Konsensprotokolle auch für eine verteilte Kalman-Filterung oder verteilte Klassifikation von abweichenden Sensorknoten [137], [146]–[149].

Im Allgemeinen basieren die erwähnten Verfahren stets auf der Annahme, dass alle Knoten im Netzwerk konstruktiv kooperieren wollen und demnach allen Knoten vertraut werden kann. Für die erdachten Einsatzzwecke ist diese Annahme sicherlich akzeptabel; im verteilten *Data Mining* hingegen hält diese Annahme nicht. *Data Mining* wird durchgeführt, damit ein Dienstanbieter ein möglichst umfassendes und genaues Abbild seiner Nutzer erhält, um darauf aufbauend zum Beispiel wirtschaftliche Entscheidungen zu treffen. In diesem Szenario kooperieren die Knoten nicht; vielmehr sollen sie lokale (private oder gar geheime) Informationen preisgeben. Es ist deshalb leicht vorstellbar, dass einzelne Knoten Informationen nicht weitergeben oder das Gesamtergebnis

beeinflussen wollen. Zur verteilten Datenerhebung muss den Nutzern also zumindest der Schutz ihrer Privatsphäre garantiert werden, sodass preisgegebene private Informationen zumindest nicht mit Identitäten oder Nutzerprofilen in Verbindung gebracht werden können. Gleichzeitig muss sichergestellt werden, dass böswillige Knoten das Ergebnis nicht beeinflussen können.

### 5.2.2 Privatsphäreschonende Datenerhebung

Dieser Abschnitt widmet sich dem Schutz privater Informationen in *Data Mining*-Verfahren. Alle hier vorgestellten Techniken und Algorithmen sind mit dem Ziel entwickelt worden, erhobene Daten nicht so veröffentlichen zu müssen, dass ein Zusammenhang zu einem einzelnen Datensatz oder gar zu einer einzelnen Person hergestellt werden kann – nicht einmal von der Partei, die die Datenerhebung initiiert. Herangehensweisen an dieses Problem existieren zahlreich und in unterschiedlichen Fachbereichen. So finden sich Arbeiten aus dem Bereich statistischer Datenbanken ebenso wie solche aus dem Bereich der Verschlüsselung oder statistischer Veröffentlichung von Daten.

Trotz verschiedener Hintergründe ähneln sich die Verfahren und es können zwei klassische Szenarien identifiziert werden. Das erste Szenario beschreibt, wie Daten unter zwei oder mehr Parteien aufgeteilt werden, sodass auf der Gesamtheit der Daten die gewünschten Untersuchungen durchgeführt werden können ohne jedoch einer Partei Einblick in private Daten einzelner anderer zu gewähren. Für dieses Vorgehen wird auf Protokolle zur sicheren kooperativen Berechnung – sogenannte *secure multiparty computation* – gesetzt. Die Privatheit der Daten wird in diesen Verfahren dadurch gewährleistet, dass sie nie den eigenen Speicherbereich verlassen. Darauf aufbauende Berechnungen werden unter den Parteien wechselseitig ausgehandelt. So könnte eine persönliche Shopping-Historie in einem Online-Kaufhaus auf dem Endgerät eines Nutzers gespeichert sein. Soll diesem Nutzer eine personalisierte Empfehlung auf der Basis der getätigten Einkäufe gegeben werden, kann diese mit einem entsprechenden Algorithmus berechnet werden, ohne, dass die Shopping-Historie zum Kaufhaus übertragen werden muss. Daher werden solche Verfahren vor allem in medizinischer Forschung eingesetzt, in der beispielsweise Patientendaten das Krankenhaus, in dem sie erhoben wurden, nie verlassen dürfen. Aus Sicht der Privatheit der Daten sind diese Verfahren deshalb äußerst attraktiv. Für echtes *Data Mining* eignen sie sich allerdings nur begrenzt: die Dienstanbieter lernen damit nichts über ihre Nutzer [150].

Für die weiteren Betrachtungen ist deshalb das zweite klassische Szenario interessant. Darin werden die zu untersuchenden Daten so verändert, dass ihre Veröffentlichung niemandes Privatsphäre mehr kompromittieren kann, während weiterhin aussagefähige Schlüsse aus den veränderten Daten gezogen werden können. Diese Transformationen reduzieren im Allgemeinen die Granularität der Daten, um dadurch die Privatsphäre der Individuen zu schützen. Da durch solche Transformationen auch Informationen verloren gehen, muss stets

abgewogen werden, in welchem Umfang die Privatsphäre geschützt und gleichzeitig die Effektivität der *Data Mining*-Algorithmen reduziert werden soll. Als Transformationen werden meist entweder gruppenbasierte Anonymisierungs- oder Randomisierungsverfahren eingesetzt [151], [152]. Beide Vorgehensweisen werden im folgenden vorgestellt.

#### 5.2.2.1 Gruppenbasierte Anonymisierungsverfahren

Die Privatsphäre in Datenbeständen fußt häufig auf der Anonymität der Einträge. In der Öffentlichkeit ist beispielsweise gerne von „anonymisierten Datensätzen“ die Rede. Darin werden Klarnamen einfach durch Pseudonyme ersetzt, weil davon ausgegangen wird, dass einzelne Einträge dann nicht mehr mit Individuen in Verbindung gebracht werden können.

Tatsächlich reicht es aber nicht aus, die Identifikatoren der Datensätze – etwa die Primärschlüssel – zu entfernen. Aus der Kombination mehrerer nichtsensitiver Attribute können Individuen dennoch eindeutig in einer Population identifiziert werden. Ein häufig zitiertes Beispiel ist die Kombination aus Geburtsort, Geburtsdatum und Geschlecht, mit der Personen auch ohne Kenntnis ihres Namens in einem entsprechenden Datensatz ermittelt werden können. Eine solche Kombination von Attributen, die eine eindeutige Identifikation in einem Datensatz ermöglichen, nennt man Quasi-Identifikatoren. Auf der Ebene der Anfragen an solche Datenbestände wurde ein simples Verfahren entwickelt, mit dem die Anonymität der Einträge im Hinblick auf solche Quasi-Identifikatoren geschützt werden kann: $k$-Anonymität [153]. Eine Tabelle $T$ etwa genügt der $k$-Anonymität, wenn für jeden Eintrag $t$ mindestens $k-1$ andere Einträge $t_{a_1}...t_{a_{k-1}}$ existieren, die in der Ausprägung ihrer Attribute nicht unterscheidbar sind.

Ein Weg, $k$-Anonymität herzustellen, führt über Generalisierung von nichtsensiblen Attributen. Diese werden solange zusammengefasst, bis in jeder Gruppe gleichwertiger Attribute mindestens $k$ Einträge enthalten sind. Bezogen auf das Beispiel mit Geburtsdatum, -ort und Geschlecht könnte $k$-Anonymität etwa über das Zusammenfassen von Postleitzahlbereichen erreicht werden. Für den Postleitzahlbereich $80xxx$ könnten damit eher $k$ männliche Kinder gefunden werden, die an Silvester geboren wurden, als in 80333 alleine.

So einfach verständlich und simpel die $k$-Anonymität umzusetzen ist, so leicht ist sie auch zu brechen. Verfügt ein Angreifer etwa über Hintergrundwissen bezüglich der nicht-sensiblen Attribute eines gesuchten Individuums, kann er damit entweder direkt die generalisierten Instanzen identifizieren, in denen das Individuum enthalten sein muss, oder Instanzen ausschließen, sodass effektiv weniger als $k-1$ gleichwertige Instanzen übrig bleiben. Dieses Vorgehen wird in der Literatur als positive, beziehungsweise negative Enthüllung (*positive / negative disclosure*) bezeichnet [151]. Alternative Angriffe bestehen in sequentiellen oder leicht abgewandelten Anfragen an denselben Datenbestand, aus deren geschickter Kombination bestimmte Einträge womöglich eindeutig identifiziert werden können.

Zwar gibt es zahlreiche Protokollerweiterungen der $k$-Anonymität, mit der die beschriebenen Angriffe erschwert werden können – für eine Übersicht sei hier etwa auf [152] verwiesen. Es lässt sich aber festhalten, dass mit der $k$-Anonymität vor allem die Identität eines Eintrags geschützt werden kann; unter Umständen sind aber Rückschlüsse auf dessen sensiblen Attribute möglich.

Das gruppenbasiertes Anonymisierungsverfahren, mit dem innerhalb der Gruppen mit minimal $k$ Einträgen auch eine schützende Diversität unter den Attributen hergestellt wird, heißt $l$-Diversität [154]. Danach ist ein generalisierter Datenblock $l$-divers, wenn er mindestens $l$ wohl-repräsentierte Werte für ein sensibles Attribut $S$ umfasst. Eine Tabelle ist l-divers, wenn jeder enthaltene Datenblock selbst l-divers ist. Die Definition beschreibt nicht, was genau mit wohlrepräsentierten Werten (*well-represented values*) gemeint ist. Ziel dieses Verfahrens ist jedenfalls, die positive Enthüllung von individuellen Attributen zu erschweren, indem die Diversität unter den sensiblen Attributen erhöht wird. Demgegenüber verändert sich die Komplexität einer negativen Enthüllung nicht, da keine Annahmen über das Hintergrundwissen eines Angreifers gemacht werden können.

Ein großes Problem beim Erzeugen von $l$-Diversität ist die Spärlichkeit von Daten. In vielen echten Datensätzen sind bestimmte Attribute für eine Vielzahl von Individuen gar nicht gefüllt – man denke da etwa an den Anteil des gesamten Sortiments des Online-Kaufhauses Amazon ([236]), für den ein Nutzer überhaupt eine Bewertung abgegeben hat. Deshalb wurde mit [155] ein weiteres gruppenbasiertes Anonymisierungsverfahren vorgestellt, das *t-closeness* heißt. Dieses fordert, dass die Distanz zwischen der Verteilung eines sensiblen Attributs innerhalb einer anonymisierten Gruppe und der Verteilung desselben Attributs innerhalb der gesamten Population nur einen Grenzwert $t$ betragen darf. Als Distanzmaß kommt in diesem Verfahren die *Earth Mover*-Distanz zwischen zwei Verteilungen zum Einsatz. In der Literatur gilt die *t-closeness* als sehr mächtiges Werkzeug für die privatsphäreschonende Erhebung von numerischen Werten [152].

#### 5.2.2.2 Privatsphäreschutz durch Randomisierung

In der letzten Dekade hat sich in der Literatur ein weiteres starkes Modell für die Auffassung von Privatsphäre etabliert, das im Gegensatz zu den im vorherigen Abschnitt vorgestellen Methoden nicht als Eigenschaft von Tabellen formuliert ist. Dieses Modell ist als Bedingung für die Ausgabe von Funktionen definiert, die auf schützenswerten Daten operieren und deren Eingabewerte nur geringfügig verändert werden. Dieses Prinzip heißt *$\epsilon$-differential privacy*, oder $\epsilon$-differentielle Privatheit [156]–[158].

Für die Definition von $\epsilon$-differentieller Privatheit seien zunächst $x$ und $x'$ zwei Vektoren, die sich nur in einem einzigen Wert unterscheiden. Weiter sei $\hat{f}$ eine randomisierte Funktion, die erzeugt wird, indem beispielsweise auf die Funktionswerte einer Funktion $f$ zufälliges Rauschen addiert wird. Die $\epsilon$-differentielle Privatheit ist dann wie folgt definiert:

*Die Funktion $\hat{f} : D^n \mapsto R$, die Daten einer Domain $D^n$ auf eine Menge $R$ abbildet, heißt $\epsilon$-differentiell privat, wenn für alle Vektoren $x$ und $x'$, die sich nur in einem einzigen Wert unterscheiden, und für alle Ausgabewerte $O \subseteq R$ gilt:*

$$P\left(\hat{f}\left(x\right) \in O\right) \leq e^{\epsilon} P\left(\hat{f}\left(x'\right) \in O\right)$$

Meist wird die $\epsilon$-differentielle Privatheit hergestellt, indem Rauschen einer Laplace-Verteilung hinzuaddiert wird:

$$\hat{f} = f + v,$$

wobei $v$ entsprechend der Sensitivität der Funktion $f$ über alle Paare von Vektoren $x$ und $y$ aus $D$ kalibriert wird, die sich ihrerseits nur in einem Wert unterscheiden:

$$v \sim Lap\left(\frac{max_{x,y}\left|f(x) - f(y)\right|}{\epsilon}\right).$$

In einfachen Fällen kann $\hat{f}$ erzeugt werden, indem für $v$

$$v \sim Lap\left(\frac{1}{\epsilon}\right)$$

angenommen wird.

Für Funktionen $f$, die im $\mathbb{R}^n$ operieren, ist diese Definition simpel anzuwenden; schließlich sind zahlreiche statistische Funktionen kaum sensibel gegenüber Rauschen. Sind genügend Datenpunkte vorhanden, kann Laplace-Rauschen addiert werden, wenn etwa die Summe über die Datenpunkte oder ihren Durchschnitt berechnet werden sollen. Durch diese Konstruktion wird garantiert, dass aus zwei aufeinanderfolgenden Ausgaben derselben Funktion mit minimal veränderten Eingabewerten nicht erkannt werden kann, ob ein individueller Datensatz zu dem Ergebnis beigetragen hat, oder nicht.

Allgemein wird Randomisierung zur privatsphäreschonenden Datenerhebung in zahlreichen Arbeiten eingesetzt [159]–[162]. Allerdings basieren zahlreiche dieser Lösungen auf Architekturen mit einer zentralen vertrauenswürdigen Entität [156], [163]–[165]. Diese zentrale Entität kann etwa eine statistische Datenbank sein, die Informationen von Peers sammelt und über das beschriebene Prinzip der differentiellen Privatheit statistische Informationen für nicht-vertrauenswürdige Dritte veröffentlicht. Im Hinblick auf die Privatsphäre der Individuen ermöglicht eine solche Konstruktion starke Garantien. Wie bereits beschrieben, eignet sich eine zentrale Datenbank in kontextsensitiven ubiquitären Umgebungen häufig nicht. Zudem etabliert sich mit dieser zentralen Datenbank auch stets ein zentraler Angriffspunkt auf das Netz. Im folgenden wird daher eine vollständig verteilte Lösung zur privatsphäreschonenden Datenerhebung vorgestellt.

### 5.2.3 Verteilte Datenerhebung und Konsens mit Garantien an die Privatsphäre

In diesem Abschnitt wird ein Verfahren beschrieben, mit dem die Verteilung eines geheimen, numerischen Wertes unter Peers kollaborativ erhoben werden kann, wobei die Privatheit der individuellen Informationen in Bezug auf das Endergebnis garantiert wird und auf eine vertrauenswürdige zentrale Entität, die Einsicht in die kommunizierten Daten erhalten muss, vollständig verzichtet werden kann. Dieser geheime, numerische Wert kann etwa eine Profilinformation sein (wie zum Beispiel das Alter) oder die Antwort auf eine Umfrage auf einer Likert-Skala. Für die folgenden Betrachtungen wird diese Metapher von einer Umfrage verwendet.

Abstrakter formuliert, handelt es sich um einen Konsensalgorithmus, der globale Statistiken über das Netzwerk zusammenträgt, indem Knoten lokale schützenswerte Daten propagieren. Die lokale Berechnung der globalen Funktion führt dazu, dass die Knoten auch etwas über das Gesamtergebnis lernen. Im Netz entsteht unter bestimmten Bedingungen ein Konsens über das Gesamtergebnis.

Die Privatheit der lokalen Daten eines Knotens ist geschützt, weil aus den ausgetauschten und global ermittelten Statistiken kein Rückschluss auf die Daten einzelner Knoten möglich ist. Analog zur $\epsilon$-differentiellen Privatsphäre, werden die kommunizierten Daten dazu mit einer zufälligen Störverteilung perturbiert. Um einen Konsens erreichen zu können, operiert ein Konsensalgorithmus dann entweder nur auf gestörten Daten. In diesem Fall erhöht das Rauschen den Schutz der Daten, während die Genauigkeit des Gesamtergebnis darunter leidet. Alternativ könnte die Störung der Daten mit der Zeit oder innerhalb des Netzwerks verändert werden, sodass der Einfluss der Störung im arithmetischen Mittel über die gesamte Laufzeit oder über alle Knoten des Netzwerks hinweg verschwindet. In diesem Fall ist der Konsens wesentlich genauer, während private Daten stellenweise weniger geschützt sein können. Für den hier vorgestellten Algorithmus kommt diese zweite Strategie mit zeitabhängiger Störung zum Einsatz.

Zurück zur Metapher der Umfrage: In dem hier beschriebenen Verfahren wird eine Umfrage von einer zentralen Entität initiiert, um die Verteilung des numerischen Werts unter den Teilnehmern zu liefern. Jeder Peer verwendet während der Beantwortung der Umfrage eine Fehlerverteilung zum Schutz seiner eigenen sensiblen Antworten vor böswilligem Abhören. Der Einfluss der Störverteilung wird sukkzessive reduziert, sodass das System insgesamt gegen einen Konsens konvergiert. Nachdem die Peers ausreichend häufig miteinander kommuniziert haben, hält keiner mehr identifizierbare private Informationen. Daher können die Knoten dann die Gesamtverteilung verteilt abschätzen, indem zum Beispiel der Algorithmus von Chan [166] angewendet wird. Aus dem Endergebnis der Umfrage sind keine Knoten identifizierbar und keine Rückschlüsse auf individuelle Antworten möglich. Damit trägt dieses Verfahren so-

wohl dem Interesse eines Dienstanbieters Rechnung, der an einer Datenerhebung unter seinen Nutzern interessiert ist, als auch dem Interesse der Nutzer an der Privatheit ihrer persönlichen Daten. Dieses Verfahren wurde bereits in [244] veröffentlicht.

Dieses Vorgehen ist durch ein gewisses Misstrauen gegenüber jedwedem zentralisierten Dienstanbieter motiviert, der persönliche oder sensible Daten willkürlich sammeln könnte. Für das hier beschrieben Verfahren ist eine zentrale Einheit deshalb nur im Sinne eines Plattform- oder Dienstanbieters vorhanden. Dieser kann Umfragen unter den verbundenen Peers initiieren. Zur Kommunikation unter den Peers stellt dieser Anbieter einen verschlüsselten Kanal zur Verfügung, den der Anbieter selbst nicht einsehen kann.

Bisher sind einige Verfahren bekannt, die $\epsilon$-differentielle Privatheit in einem vergleichbaren verteilten Umfeld zur Datenerhebung oder Konsensbildung einsetzen. Das hier beschriebene Verfahren basiert auf der Addition von Rauschen. Dieser Ansatz unterscheidet sich aber insbesondere in der Analyse des möglichen Wissensgewinns eines Angreifer, der die Kommunikation unter den Teilnehmern abhört. Im Gegensatz zu bekannten Arbeiten kommt hier der Standardfehler, auch Stichprobenfehler genannt, als Maß für den möglichen Verlust an Privatheit zu einem beliebigen Zeitpunkt während der Laufzeit der Abstimmung zum Einsatz. Damit lässt sich hier der Einfluss des Rauschens auf den Informationsgewinn für einen Angreifer simpler erfassen als in bekannten Arbeiten.

#### 5.2.3.1 Genereller Ablauf des Verfahrens

Das Verfahren läuft prinzipiell in drei Schritten ab:

1. Konstruktion eines Overlay-Netzes

2. Rundenbasierte, $\epsilon$-differentiell private Kommunikation über das Overlay-Netz

3. Verteilte oder zentrale Aggregation der Daten

Im ersten Schritt iniitiert eine zentrale Entität eine Umfrage und steht den Knoten bei der Konstruktion des Overlay-Netzes zur Verfügung. Das Overlay-Netz wird während des zweiten Schrittes mehrmals neu aufgebaut und besteht stets nur für eine begrenzte Anzahl Runden. In jeder Runde verteilen die Knoten untereinander ihre Sicht auf das Gesamtergebnis, das ihre privaten Informationen enthält und deshalb stets durch eine Fehlerverteilung gestört wird. Die Sicht eines Knotens integriert allerdings nur eine begrenzte Anzahl anderer Teilnehmer. Dieses Vorgehen entspricht daher einem lokalen Algorithmus, der mit den Eingaben weniger benachbarter Peers sehr genaue Ergebnisse erzielt. Gleichzeitig skaliert das System so unabhängig von der Anzahl der teilnehmenden Peers [167]. In einem letzten Schritt werden die verteilten und

gestörten Daten schließlich zu einem Durchschnitt aller individuellen Ergebnisse zusammengefasst. Die einzelnen Schritte werden im folgenden detaillierter beschrieben.

#### 5.2.3.2 Konstruktion des Overlay-Netzwerks

Das intendierte Overlay-Netzwerk ist ein verbundener zufälliger Graph mit einem konstanten Knotengrad $m$. Dementsprechend kommuniziert jeder Knoten in jeder Runde mit genau $m$ Nachbarn.

Damit diese Kommunikation privat und gegen Abhören geschützt erfolgen kann, kommt das Konzept der *Locagramm Exchanger* zum Einsatz [168]. Dieses Konzept basiert auf der Verknüpfung von Identitäten mit öffentlichen Schlüsseln. Jeder Knoten erzeugt ein RSA-Schlüsselpaar und veröffentlicht seinen öffentlichen Schlüssel beim Dienstanbieter. Dieser muss eine Plattform bereitstellen, die *Exchanger* genannt wird und über die der Austausch von Locagrammen erfolgt, bei der also Locagramme abgelegt und abgerufen werden können, ohne dass eine zusätzliche Authentifikation nötig ist. Die Locagramme, die über diesen Exchanger ausgetauscht werden, sind kleine Datenpakete, die mit dem öffentlichen Schlüssel des Empfängers verschlüsselt und mit diesem Schlüssel als Zieladresse abgelegt werden.

Die zentrale Entität, die den Exchanger bereitstellt, kann als eine Art *Microblogging*-Plattform gesehen werden. Darauf sind allgemeine Aktivitäten generell einsehbar – etwa wenn Locagramme abgelegt werden. Alle weiteren Informationen, wie kommunizierte Daten oder Empfangsstatus bleiben wegen der starken Verschlüsselung aber vertraulich – auch gegenüber dem Plattformanbieter. Diese Kommunikationsplattform kann dahingehend erweitert werden, dass Abfragen nur von bekannten öffentlichen Schlüsseln zugelassen werden und die Antwortgeschwindigkeit begrenzt wird oder gar exponentiell verzögert wird, sodass nur der Plattformanbieter selbst die genauen Aktivitätsinformationen hält.

Die zentrale Entität ist jedenfalls nur eine logische zentrale Komponente, die selbst auch in hohem Maße verteilt realisiert werden kann. Sie muss lediglich mit allen Knoten kommunizieren können, um den Austausch der Locagramme zu ermöglichen. Daten, die die Peers im weiteren Verlauf der Umfrage kommunizieren, bleiben gegenüber der zentralen Einheit allerdings verborgen.

Das Overlay-Netzwerk wird aufgebaut, indem jeder Knoten zunächst ein RSA-Schlüsselpaar erzeugt und seinen öffentlichen Schlüssel beim Plattformanbieter ablegt. Anschließend lädt jeder Knoten alle öffentlichen Schlüsel herunter und wählt zufällig einen öffentlichen Schlüssel eines Kommunikationspartners aus. Mit diesem Schlüssel und dem zentralen Exchanger ist eine Kommunikation zunächst in eine Richtung möglich. Für eine symmetrische Kommunikation erzeugen die Kommunikationspartner einen virtuellen, vertraulichen und authentifizierten Kommunikationskanal, indem der initiierende Knoten mit einer Koppelungsnachricht zunächst seinen eigenen öffentlichen Schlüssel verschlüsselt übermittelt.

Auf diese Weise verknüpft sich jeder Knoten mit $m$ zufälligen Nachbarn. Die zentrale Einheit kann dabei die Anzahl der Datenübertragungen an einen Knoten überprüfen, um die korrekte Anzahl an Kommunikationspartnern durchzusetzen. Die zentrale Einheit kann auch verhindern, dass im Overlay-Graphen unverbundene Komponenten entstehen. Die Wahrscheinlichkeit dafür ist vom Knotengrad $m$ und der Anzahl von Knoten $N$ abhängig und kann damit in echten Anwendungsszenarien soweit reduziert werden, dass entsprechende Gegenmaßnahmen nicht nötig werden.

Insgesamt haben sich die Knoten im ersten Schritt der Umfrage auf einen zufälligen, vollverbundenen Graphen mit Grad $m$ geeinigt.

#### 5.2.3.3 $\epsilon$-differentiell private Kommunikation

Im Kern basiert der Algorithmus auf der Verteilung weniger Stichproben, die in jeder der $R$ Runden aus mehreren spezifischen Verteilungen gezogen werden.

Die erste Verteilung repräsentiert die aktuelle Schätzung des Gesamtergebnisses. Diese Schätzung verwaltet jeder Knoten separat, indem er die Verteilungsparameter $\mathcal{N}_1 = \mathcal{N}(\mu_E, \sigma_E)$ in jeder Runde aktualisiert. Der Mittelwert $\mu_E$ dieser Schätzung wird mit dem privaten und korrekten, geheimen Wert des Knotens initialisiert: $\mu_E = \mu_S$. Die Standardabweichung der Schätzung wird anfangs mit $\sigma_E = 0$ belegt.

Da der Algorithmus die Privatheit der geheimen Daten erhalten soll, muss jeder Knoten an der Umfrage teilnehmen können, ohne seinen privaten Eingabewert offenbaren zu müssen. Das wird entsprechend der $\epsilon$-differentiellen Privatheit über die Addition einer Störverteilung realisiert, die jeder Knoten zu seiner geheimen Eingabe hinzuaddiert, bevor der Knoten seine Eingabe kommuniziert.

Diese Störverteilung ist die zweite Verteilung, die jeder Knoten hält. In diesem Fall handelt es sich um eine Gaußsche Normalverteilung mit Mittelwert Null und beliebig hoher Standardabweichung $\mathcal{N}_2 = \mathcal{N}(0, \sigma_t)$. Diese Standardabweichung $\sigma_t$ wird in der ersten Hälfte der $R$ Runden sukzessive auf Null reduziert. Anfangs wird eine starke, im weiteren Verlauf gar keine Störung mehr angewandt.

Ist die Streuung der Störverteilung groß genug und die Größe der Stichproben pro Nachricht gleichzeitig klein genug, kann aus einer einzelnen Nachricht kein Rückschluss auf die privaten Informationen eines Knotens gezogen werden. Mit zunehmender Anzahl der Nachrichten desselben Knotens ist allerdings eine zunehmend genauere Schätzung dessen privater Daten möglich. Daher muss das Overlay-Netz in jeder Runde zufällig neu aufgebaut werden.

Aus den zwei genannten Verteilungen werden jedenfalls die kommunizierten Datenpakete konstruiert. Jeder Knoten zieht eine Stichprobe der festen Größe $k$ aus der Summe der zwei Verteilungen, die dann übertragen werden:

$$\mathcal{N}_1 + \mathcal{N}_2 = \mathcal{N}(\mu_E, \sigma_E) + \mathcal{N}(0, \sigma_t).$$

Bei der Addition der beiden Verteilungen bleibt der Mittelwert $\mu_E$ erhalten. Das bedeutet, dass der Mittelwert dieser summierten Verteilung anfangs dem privaten Wert des Knotens entspricht – $\mu_E = \mu_S$. Wird die Größe der Stichprobe $k$ allerdings klein genug und die Standardabweichung der Störverteilung $\sigma_t$ groß genug gewählt, ist der erwartete Fehler des Mittelwerts auch zu Beginn der Umfrage beliebig hoch.

Ein Knoten, der ein derart konstruiertes Datenpaket empfängt, aktualisiert damit seine eigene Schätzung $\mathcal{N}_1$ wie folgt. Der empfangende Knoten konstruiert eine temporäre Verteilung $\mathcal{N}_t$, die er mit $l$ Wiederholungen seines eigenen, geheimen Werts initialisiert und mit einer Stichprobe der Größe $\frac{\beta}{1-\beta}l$ aus allen bisher empfangenen Daten befüllt. Die Verteilung $\mathcal{N}_t$ besteht also zu einem Anteil von $1-\beta$ aus dem geheimen Wert des Knotens und zu einem Anteil von $\beta$ aus den bisher empfangenen Daten. Aus dieser Verteilung $\mathcal{N}_t$ aktualisiert der Knoten schließlich die Parameter seiner Schätzung $\mathcal{N}_1$.

Typischerweise sollte $\beta$ so gewählt werden, dass der Einfluss der empfangenen Daten auf die Verteilung $\mathcal{N}_t$ deutlich höher ist, als der des geheimen Werts eines Knotens. Dadurch ist sichergestellt, dass auch Daten mehrerer Kommunikationspartner ausreichend in die Schätzung jedes Knotens mit einfließen. Das wird zusätzlich durch Agrawal und Aggarwal motiviert, die gezeigt haben, dass eine bekannte Störverteilung nahezu fehlerlos rekonstruiert werden kann, wenn nur genug Stichproben aus dieser Verteilung gesammelt werden können [169]. Ein Angreifer, der mehrere Pakete abhören kann, kann bei geschickter Wahl eines $\beta$ also nur unsicheres Wissen über einen nichtkonstanten Wert erlangen, in den die geheimen Eingabewerte zudem nur zu geringem Teil einfließen.

Insgesamt konvergiert das gesammelte Wissen aller Knoten gegen das korrekte Ergebnis. Im Gegensatz zu Konsensalgorithmen wie [146] erhält ein Peer zunächst nur eine begrenzte Sicht auf das Gesamtergebnis. In bestimmten Szenarien wie verteilten Umfragen ist das ein gewünschter Seiteneffekt; insbesondere, wenn aus wirtschaftlichen Interessen das tatsächliche Ergebnis gegenüber Angreifern oder teilnehmenden Wettbewerbern verborgen bleiben soll. Wird allerdings der Knotengrad $m$ des Overlay-Graphen im Verhältnis zur Gesamtzahl der Knoten $N$ erhöht, erhält jeder einzelne Knoten auch eine zunehmend genaue Schätzung des Gesamtergebnisses.

#### 5.2.3.4 Aggregation

Im letzten Schritt wird das gesammelte Wissen aller Knoten zusammengetragen. Der Durchschnitt aller individuellen Schätzungen bildet das Gesamtergebnis der Umfrage. Der wichtigste Schritt des Algorithmus besteht darin, die Schätzungen der einzelnen Knoten möglichst weit zu verbreiten, um ihnen zum Schutz ihrer geheimen Werte so möglichst stark abweichende Werte zuzuordnen. Daher ist die Zeit, bis ein Konsens erreicht ist, eher hoch. Es wird deshalb vorgeschlagen, den Ablauf nach einer fixen Anzahl von Runden zu beenden und alle von den Teilnehmern ermittelten Mittelwerte $\mu_E$ gegenüber der zentralen Entität zu veröffentlichen. Da der Einfluss des Zufalls auf die Schätzung

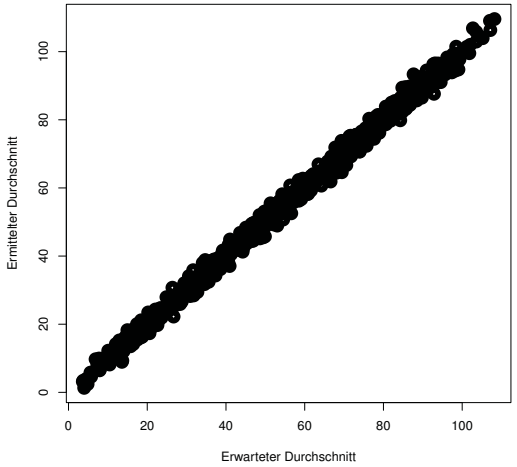



Abbildung 5.6: Erwartete und ermittelte Mittelwerte

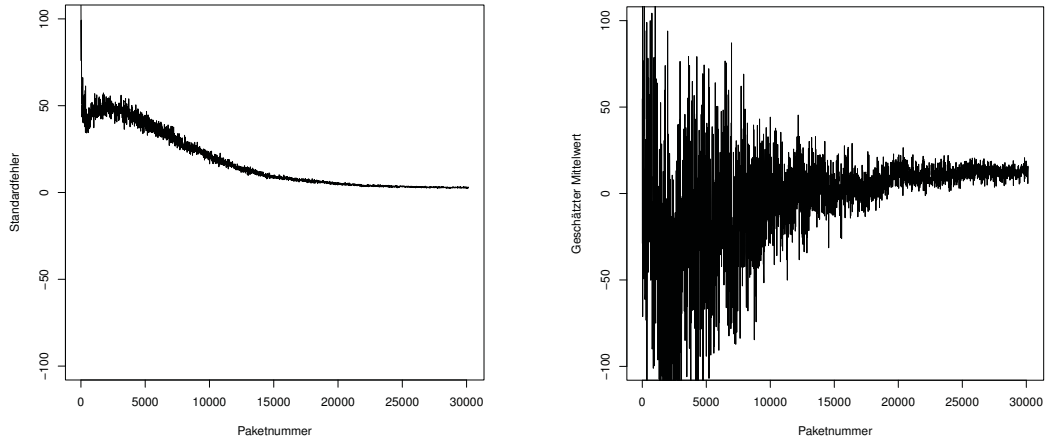
eines individuellen Knotens nach einer ausreichenden Anzahl an Runden hoch ist, stellt die Veröffentlichung zu diesem Zeitpunkt keine Anforderungen mehr an die Vertrauenswürdigkeit der zentralen Entität. Es kann an dieser Stelle also auch direkt ein Algorithmus zum verteilten Berechnen eines Mittelwerts eingesetzt werden, wie etwa der Algorithmus von Chan [166].

Insgesamt wurde ein Weg beschrieben, über den eine Datenerhebung von privaten Informationen möglich ist, ohne dass eine zentrale Entität in der Lage ist, die Privatheit der Daten eines Knoten zu kompromittieren. Vielmehr ist die Privatheit der Nutzer auch gegenüber anderen Knoten im Netz geschützt, da die Informationen, die einem einzelnen Knoten zur Verfügung stehen, zu keiner Zeit die Enthüllung von privaten Daten anderer Knoten erlauben.

#### 5.2.3.5 Simulation

Zur Evaluation des Verfahrens wurden mehrere Simulationen durchgeführt. Dabei wurden Mittelwert und Standardabweichung der Zielverteilung für jede Simulation verändert und zufällige Stichproben aus dieser Verteilung an die Knoten als geheime Informationen verteilt. Die erste Beobachtung ist, dass der Algorithmus insgesamt ziemlich gut abschneidet – vor allem in Anbetracht des starken zufälligen Störeinflusses auf einzelne Datenpakete. In Abbildung 5.6 ist die Qualität einer Reihe von Simulationen mit variierenden Mittelwerten und Standardabweichungen dargestellt. Es ist sofort erkennbar, wie das Gesamtergebnis das Optimum von $f(x) = y$ recht gut annähert.

Abbildung 5.7 verdeutlicht den zeitlichen Ablauf einer Simulation. Grafik 5.7(a) zeigt den Standardfehler, der angibt, wie genau der Mittelwert einer Verteilung anhand einzelner Beobachtungen abgeschätzt werden kann. Eine detaillierte Diskussion dieses Maßes auch im Hinblick auf die Privatheit der Daten folgt im nächsten Abschnitt. Grafik 5.7(b) zeigt das Gesamtergebnis im Zeitverlauf, wenn der finale Aggregationsschritt nach jeder Runde durchgeführt würde. Es ist gut erkennbar, wie die Varianz des Mittelwerts über die Zeit




(a) Standardfehler der beobachteten Pakete im Zeitverlauf

(b) Geschätzter Mittelwert einer Stichprobe beobachteter Pakete im Zeitverlauf

Abbildung 5.7: Entwicklung und Auswirkung des Standardfehlers

abnimmt, da der Einfluss der Störverteilung $\mathcal{N}_2$ rundenweise reduziert wird. Ab einem gewissen Zeitpunkt schwankt das Ergebnis zufällig und mit geringer Varianz um den Mittelwert der Zielverteilung. Insgesamt wäre dies auch die Sicht eines optimalen Angreifers, der die Kommunikation aller Knoten über alle Runden hinweg abhören könnte. Eine eingehende Analyse dessen, was ein (optimaler) Angreifer aus der Kommunikation einzelner Knoten lernen kann, folgt in den nächsten Abschnitten.

#### 5.2.3.6 Analyse der Privatsphäre der Umfrageteilnehmer

Das beschriebene Verfahren widmet sich der Privatsphäre der beteiligten Individuen hinsichtlich

- ihrer direkten Kommunikationspartner,
- der Gesamtheit der Teilnehmer und
- dem Dienstanbieter.

Wenn im Folgenden der Informationsfluss durch das Overlay-Netzwerk untersucht wird, wird zwischen zwei Arten von Informationen unterschieden: Metainformationen und Dienstinformationen. Metainformationen sind jegliche Art von öffentlich einsehbaren Faktoren zur Netzwerkaktivität, wie etwa die adressierten Knoten, technische Adressen und Zeitstempel. Dienstinformationen sind demgegenüber die algorithmischen Eingabewerte und damit die eigentlich schützenswerten Informationen.

Metainformationen stellen eine nur geringe Gefahr für die Privatheit der Nutzerdaten dar. Selbst eine umfassende Sicht auf das Netzwerk über die RSA-Schlüsselpaare enthält wenig kompromittierende Information, da die Schlüs-

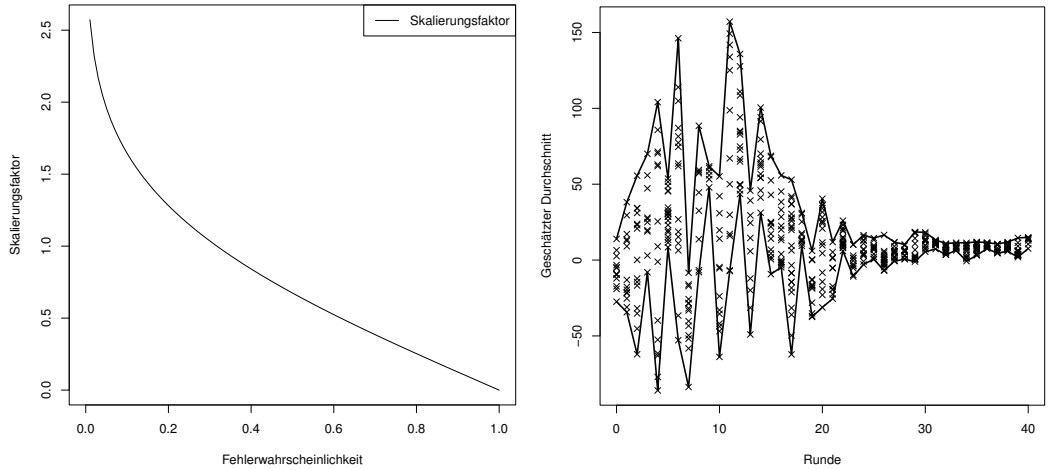
selpaare für jede Umfrage neu erzeugt und Kommunikationspartner zufällig ausgewählt werden.

Dienstinformationen werden zwischen maximal $m$ zufälligen Knoten ausgetauscht. Diese Informationen sind mit paarweise unkorrelierten Fehlern perturbiert, die aus einer häufig verwendeten Störverteilung gezogen werden. Wie von Agrawal und Aggarval in [169] gezeigt, bietet dieses Vorgehen einen angemessenen Schutz der Daten, solange ein Angreifer nur eine begrenzte Anzahl an Stichproben aus kommunizierten Datenpaketen ergattern kann.

Nur zu Beginn kommuniziert jeder Knoten hauptsächlich seinen eigenen geheimen Wert; diese initiale Verteilung wird daher durch besonders starke Störung verzerrt. Dennoch bietet diese initiale Kommunikation prinzipiell eine gute Schätzung des geheimen Werts eines Knotens, da die kommunizierte Verteilung ihren Mittelwert bei dem geheimen Wert des Knotens hat: Jeder Knoten startet zunächst mit einer Verteilung, die ihren Mittelwert bei dem jeweiligen geheimen Wert hat und deren Standardabweichung Null ist. Zu dieser Verteilung wird eine Störverteilung mit sehr hoher Standardabweichung hinzuaddiert; da beide Verteilungen unabhängig voneinander sind, entsteht eine Verteilung mit dem bekannten Mittelwert und der Standardabweichung der Störverteilung.

Das motiviert den Standardfehler oder Stichprobenfehler als Maß für den Schutz privater Daten. Dieses Maß gibt an, wie genau der Mittelwert einer unbekannten Verteilung anhand einer begrenzten Anzahl an Stichproben abgeschätzt werden kann. Definiert ist er als die Standardabweichung der Differenzen zwischen dem tatsächlichen Mittelwert und allen Mittelwerten, die mit Hilfe von $n$ Stichproben beobachtet werden können. Mit zunehmender Größe einer Stichprobe nimmt der Standardfehler ab, während er mit steigender Standardabweichung der zu schätzenden Verteilung zunimmt. Diesem Umstand wird in dem beschriebenen Protokoll Rechnung getragen: Während der geheime Wert als Mittelwert der anfangs kommunizierten Verteilungen direkt abgeschätzt werden kann, ist die Anzahl der verfügbaren Stichproben gering und die Standardabweichung sehr hoch. Im weiteren Verlauf, wenn einem Angreifer prinzipiell mehr Stichproben zur Verfügung stehen, bewegt sich der Mittelwert zunehmend von dem privaten Wert eines Knotens weg. Die Privatheit der Daten ist damit folgendermaßen zu erklären: Um den Standardfehler um den Faktor zehn zu senken, sind 100 mal mehr Stichproben notwendig.

Mit Hilfe des Standardfehlers können Konfidenzintervalle für probabilistische Angriffe definiert werden, die als erfolgreich gelten, wenn die Schätzung des echten Mittelwerts mit ausreichender Wahrscheinlichkeit auf ein minimales Interval eingegrenzt werden kann. Ein Angriff gilt also als probabilistisch erfolgreich, wenn der geschätzte Mittelwert mit einer festen Fehlerwahrscheinlichkeit $p$ durch ein Interval der Länge $l$ oder kürzer beschrieben werden kann. Die Länge dieses Intervals kann mit Hilfe des Standardfehlers ausgedrückt werden.




(a) Skalierungsfaktoren der Konfidenzinter-valle des Standardfehlers

(b) Geschätzte Mittelwerte $\mu_E$ eines Knotens in jeder Runde im Zeitverlauf.

Abbildung 5.8: Skalierungsfaktoren des Standardfehlers und Sicht eines Angreifers, der die Kommunikation eines einzelnen Knotens abhört

Mit dem $(1-p)$-Perzentil $z$ der Standardnormalverteilung kann dieses Interval durch

$$[\bar{x} - (z * \mathrm{SF}), \bar{x} + (z * \mathrm{SF})]$$

beschrieben werden, wobei $SF$ hier für den Standardfehler steht. Der beobachtete Mittelwert liegt mit einer Wahrscheinlichkeit von $1-p$ innerhalb dieses Intervals.

Abbildung 5.8(a) zeigt einen Graphen mit den Faktoren $z$ auf der Y-Achse und der Fehlerwahrscheinlichkeit auf der X-Achse. Für eine hohe Konfidenz von 95% nimmt der Wert $z \approx 1,96$ an. Die Länge des Intervals kann damit auf $3,92 * \mathrm{SF}$ begrenzt werden. Für eine geringe Konfidenz von 50% liegt der Wert $z \approx 0,675$ und das Interval hat damit eine Länge von $1,35 * \mathrm{SF}$. Ein ausreichend hoher Standardfehler sorgt also für beliebig hohe Privatheit der Daten.

#### 5.2.3.7 Szenarien als Angriff auf die Privatheit der Umfrageteilnehmer

Nachdem die Privatheit der Daten der Umfrageteilnehmer beschrieben wurde, sollen nun einige Angriffsszenarien genauer untersucht werden. Es wird gezeigt, was ein Angreifer wissen oder unternehmen muss, um Wissen über die geheimen Informationen eines einzelnen Knotens oder über das Gesamtergebnis der Umfrage zu erlangen, oder um die Umfrage mutwillig zu beeinflussen. Das Szenario, in dem ein Angreifer jeden Knoten kontrolliert, wird dabei außen vor gelassen. In diesem Fall ist er in der Lage, die Umfrage beliebig zu beeinflussen und Wissen über jeden Knoten zu erlangen.

**Das Abhören von Nachrichten** ist eine verbreitete Angriffsmethode auf verteilte Systeme. Das Ziel ist in diesem Fall, die geheimen Informationen eines Einzelnen abzufangen oder das Gesamtergebnis der Umfrage zu ermitteln.

Die Verteilung, die ein Angreifer zu einem beliebigen Zeitpunkt abhören oder beobachten würde, hat ihren Mittelwert $\mu_c$ bei

$$\mu_c = \beta\mu_R + (1 - \beta)\mu_s,$$

wobei $\mu_R$ für den Mittelwert steht, den ein teilnehmender Knoten aus seinen bisher empfangenen Verteilungen berechnet, und $\mu_s$ für seinen eigenen geheimen Wert. Wie in Abschnitt 5.2.3.3 erläutert, kontrolliert $\beta$ das Verhältnis aus privaten und empfangenen Informationen beim Resampling der Verteilung, die dieser Knoten an seine Nachbarn kommuniziert. Um den geheimen Wert $\mu_s$ eines Knotens offenzulegen, muss dieser mittels Stichproben aus der kommunizierten Verteilung mit Mittelwert $\mu_c$ abgeschätzt werden.

Eine hinreichend genaue Abschätzung von $\mu_s$ ist nahezu unmöglich, da der Standardfehler der kommunizierten Pakete hoch ist. Erschwerend kommt hinzu, dass die Varianz der Stichproben mit jedem abgehörten Datenpaket weiter zunimmt – schließlich sollten die Mittelwert $\mu_R$ und $\mu_s$ zunehmend voneinander abweichen.

Ein Angreifer könnte höchstens Wissen über das Gesamtergebnis der Umfrage erlangen. Um das wiederum mit hinreichender Genauigkeit bestimmen zu können, genügt es aber nicht, nur einen einzelnen Knoten abzuhören, da das Gesamtergebnis nur durch die lokalen Schätzungen aller Knoten zustande kommt.

Abbildung 5.8(b) zeigt die Abschätzungen $\mu_E$ eines einzelnen Knotens in einer Simulation mit 50 Knoten, 40 Runden, $\beta = 0.9$ und einem geheimen Wert dieses Knotens von $\mu_s = -11$. Das Gesamtergebnis dieser Umfrage lag bei 10.35. Anhand der Grafik ist deutlich zu erkennen, wie variabel sich die Schätzungen eines einzelnen Knotens entwickeln. Die Punkte innerhalb der umfassenden Kurven zeigen die Abschätzungen, die auf der Basis der eingehenden Pakete in jeder Runde berechnet wurden und die damit auch zur Berechnung der ausgehenden Pakete gedient haben.

Der einzige Zeitpunkt, der für einen Angriff vielversprechend sein könnte, ist die erste Runde. Die ausgehende Verteilung eines Knotens aus dieser Runde könnte eine gute Abschätzung für dessen geheimen Wert bieten. Allerdings ist der Standardfehler $SF$ einer Stichprobe der Länge $k$ gegeben durch

$$SF = \frac{\sigma}{\sqrt{k}}.$$

In den ersten Runden des Algorithmus ist $\sigma \approx \sigma_t$. Die Standardabweichung und damit auch der Standardfehler sind damit wiederum ausreichend hoch.

Ein Angreifer kann also entweder den geheimen Wert eines Knotens mit hohem Standardfehler während der ersten Runden abhören oder im weiteren

Verlauf einen anderen Wert, der zufällig von den Nachbarn des Knotens beeinflusst ist und dessen geheimen Wert nur zu einem kleinen Bruchteil enthält. Der geheime Wert eines Knotens ist also zu keiner Zeit durch Abhören mit hinreichender Zuverlässigkeit zu bestimmen.

**Die Isolation von Knoten** ist eine weitere Angriffsmöglichkeit. Sie zielt darauf ab, private Informationen eines einzelnen Knotens zu enthüllen. Während diese Angriffsmethode sehr genaue Ergebnisse verspricht, ist sie in dem beschriebenen Overlay-Netzwerk kaum zu bewerkstelligen.

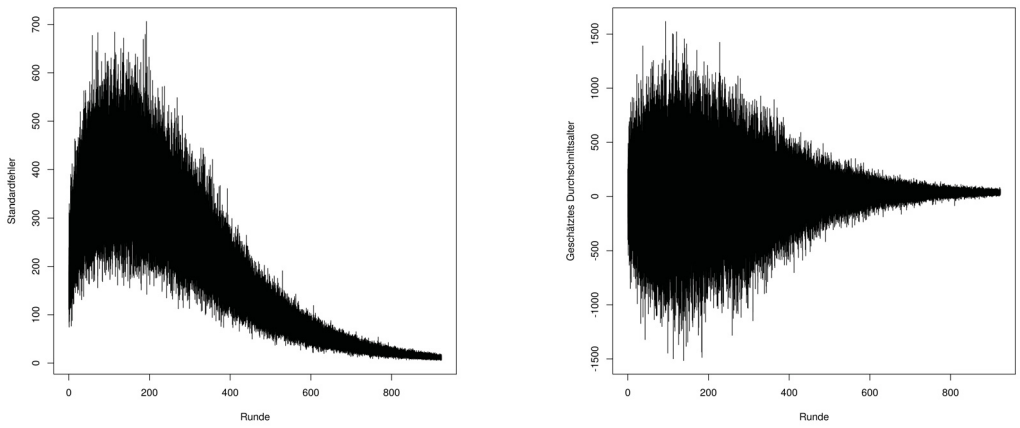
Ausgehend von der oben genannten Definition von $\mu_c$ ist leicht ersichtlich, dass $\mu_s$ berechnet werden kann, wenn einem Angreifer $\mu_R$ bekannt ist. Dieser Angreifer müsste also alle Kommunikationspartner eines Knotens ersetzen – ihn isolieren –, um ihm so nur bekannte $\mu_R$ schicken zu können. Der isolierte Knoten würde protokollkonform mit einer Verteilung antworten, die seinen geheimen Wert enthält. Der Angreifer wäre dann in der Lage, die Störverteilung aus den returnierten Datenpaketen zu subtrahieren, da diese genau dem $\mu_R$ entspricht, das der Angreifer selbst gesendet hat. Der Erfolg dieses Angriffs hängt maßgeblich von der Anzahl der vollständigen Kommunikationen ab, für die der Angreifer ein Datenpaket mit bekanntem Inhalt an sein Opfer schicken und von diesem wieder zurückbekommen muss.

Dieses Szenario erscheint für die Privatheit von geheimen Daten zwar äußerst kritisch, ist in dem hier beschriebenen Verfahren aber ausgeschlossen, solange ein Angreifer nicht in der Lage ist, die gesamte Umfrageplattform und -infrastruktur nachzubilden. Das beschriebene Overlay-Netzwerk und das Aufbauen von Kommunikationskanälen mit zufälligen Nachbarn verhindert einen Isolationsangriff wirkungsvoll – insbesondere, da die Auswahl der Kommunikationspartner lokal bei jedem Knoten aus der Liste der verfügbaren öffentlichen Schlüssel erfolgt. Solange ein Angreifer also keine Mehrheit des Netzes oder die Liste der öffentlichen Schlüssel kontrolliert, ist die Wahrscheinlichkeit sehr gering, dass er in der Lage ist, einen Knoten vollständig zu isolieren.

**Die Kontrolle über eine Minderheit**, mit der das Ergebnis einer Umfrage beeinflusst werden kann, ist eine kritische Frage. Insbesondere, da durch die beschriebene Methoden zum Schutz der Privatheit von Informationen gleichzeitig die Anonymität der teilnehmenden Knoten sichergestellt wird und somit auch die Identität von Angreifern nicht wirkungsvoll festgestellt werden kann. Das hier beschriebene Umfragesystem kann also durchaus mit Hilfe einer böswillig kooperierenden Menge von Knoten beeinflusst werden. Es sollte an dieser Stelle also weiter untersucht werden, wie *Proof-of-Work*, Anomalieerkennung, *Captchas*, Reputationsmechanismen der beteiligten Knoten oder weitere anonym lieferbare Beweise der Ehrlichkeit in das beschriebene System integriert werden können.

#### 5.2.3.8 Anwendungsbeispiel

In einem Experiment mit echten Daten einer Altersumfrage unter Radiohörern eines großen bayerischen Radiosenders wurde der beschriebene Algorithmus




(a) Standardfehler beobachteter Pakete

(b) Geschätzter Mittelwert einer Stichprobe beobachteter Pakete

Abbildung 5.9: Echte Altersumfrage unter Radiohörern im Zeitverlauf

auf seine Praxistauglichkeit hin evaluiert. Der Radiosender hat seine Umfrage am Morgen eines Werktags zwischen 6 und 7 Uhr unter 1761 Radiohörern durchgeführt und ein Durchschnittsalter von $37,522$ Jahren ermittelt – die Hörer waren zwischen 10 und 69 Jahren alt. Mit Hilfe dieser Umfragedaten wurde die Simulation intiialisiert. Die Störverteilung wurde definiert durch $\mathcal{N}_2 = \mathcal{N}(0, 600)$ und 20 Runden für den Ablauf des Algorithmus festgelegt. Jedes Datenpaket, das ein Knoten versendet hat, enthielt eine Stichprobe der Größe $k = 15$, der Resampling-Parameter $\beta$ wurde auf 0.9 festgelegt, und pro Runde hat jeder Knoten mit 25 zufällig ausgewählten Nachbarn kommuniziert. Der Standardfehler des ersten gesendeten Pakets liegt in dieser Konfiguration bei

$$\text{SEM} \geq \frac{600}{\sqrt{15}} \approx 154,9 \text{ Jahren.}$$

Wie in Abbildung 5.9 ersichtlich, verhält sich der Standardfehler der beobachteten Pakete ähnlich zum Gesamtergebnis der Umfrage. In Anbetracht des Wertebereichs einer Altersumfrage liegt der Standardfehler sehr hoch. Das Alter eines Radiohörers kann durch Abhören seiner Kommunikation nicht effektiver bestimmt werden als durch zufälliges Raten eines Alters mit einem angenommenen Durchschnittsalter von 50. Nachdem allerdings ausreichend Perturbation durch Austausch stattgefunden hat, stabilisiert sich der Mittelwert schnell. Die Simulation ermittelte einen Altersdurchschnitt von $38,886$ Jahren bei dem korrekten Ergebnis von den oben genannten $37,522$ Jahren.

#### 5.2.3.9 Schlussbemerkung

Das hier beschriebene Verfahren ermöglicht die privatsphäreschonende Erhebung von gaußverteilten numerischen Werten mit den folgenden Eigenschaften:

Der Schutz der Privatheit von Informationen ist messbar und konfigurierbar mit Hilfe des Standard- oder Stichprobenfehlers. Es kann eingesetzt werden, um private Informationen mit geringem zusätzlichen Rechenaufwand zu anonymisieren, wobei diese Anonymisierung vollständig verteilt erfolgt.

Einen interessanten Ansatz für weitere Betrachtungen bietet eine aktuelle Arbeit von Ran Wolff [170]. Darin stellt der Autor insbesondere eine Abbruchbedingung für diese Art von lokalen Algorithmen vor, die nicht mehr auf zyklenfreie Kommunikationsgraphen angewiesen ist. Eine Integration dieser Idee etwa in Kombination mit dem eingesetzten Standardfehler kann den beschriebenen Algorithmus dann beenden, wenn keine weitere Kommunikation nötig ist.

## 5.3 Zusammenfassung

Dieses Kapitel hat die Praxistauglichkeit kontextzentrischer Sozialer Netze in zwei zentralen Aspekten beleuchtet. In Abschnitt 5.1 wurde ein Experiment vorgestellt, in dem ein existierendes Soziales Netz aus eMail-Kommunikationen mit Hilfe kontextzentrischer Vernetzung simuliert und einer Metaanalyse unterzogen wurde. Der Vergleich zwischen dem resultierenden und dem echten Sozialen Graphen zeigte dieselben Akteure an zentralen Positionen im Netz. Die Ergebnisse lassen vermuten, dass über die beschriebene Metaanalyse über mehrere Kontexte hinweg erstmals auch die Intensität von Beziehungen in der Sozialen Netzwerkanalyse beschrieben werden kann.

Im Abschnitt 5.2 wurden Methoden zur Datenerhebung in verteilten Systemen mit starken Garantien an die Privatsphäre der Knoten vorgestellt. Insbesondere das in 5.2.3 beschriebene Verfahren eignet sich zum *Data Mining* von Profilinformationen oder sonstigen privaten Informationen, die durch numerische Werte ausgedrückt werden können. Aus der mit Hilfe dieses Verfahrens ermittelten Schätzung der Verteilung von numerischen Werten sind keine Rückschlüsse auf private Daten individueller Nutzer mehr möglich.

Damit wurden in diesem Kapitel zwei Aspekte zur Datenerhebung in kontextzentrischen Sozialen Netzen vorgestellt, die die Praxistauglichkeit dieses Konzepts verdeutlichen.

# 6 Zusammenfassung und Ausblick

Die Errungenschaften und positiven Eigenschaften Sozialer Netze im Internet, über die Freundschaften und Kontakte rund um den Globus gepflegt werden, sind unumstritten. Nie zuvor konnten Menschen mit allen ihren Freunden und Bekannten den Alltag so miteinander teilen, wie es heutzutage über Soziale Netze möglich ist.

Diese verführerischen Möglichkeiten sind allerdings nicht frei von Schattenseiten. Soziale Netze bieten noch kein (einfach handhabbares) Abbild der Offline-Kommunikation von Angesicht zu Angesicht. Denn Menschen passen sich ihrem Gegenüber an; sie verhalten sich Freunden gegenüber anders als Arbeitskollegen, und Vorgesetzten gegenüber anders als Familienangehörigen. Diese Anpassung kann auch als ein Sich-Selbst-Ausprobieren gesehen werden, ein Spiel mit der eigenen Identität. Notwendig und förderlich ist dieses Spiel, um eigene Ansichten zu schärfen, eine Haltung zu entwickeln und eine Persönlichkeit auszuprägen. Nur über solche zwischenmenschlichen Interaktionen formt sich eine eigene Identität und festigt sich eine Persönlichkeit.

In Sozialen Netzen ist dieses Spiel bisher kaum möglich. Statusnachrichten werden in der Regel an die gesamte Freundesliste „ausposaunt“ und erreichen somit auch Mitglieder verschiedener Cliquen und Freundeskreise gleichermaßen. Durch diese uneingeschränkte Kommunikation geht die Möglichkeit des Spielens und Ausprobierens innerhalb mehr oder weniger geschützter sozialer Kontexte verloren. Die Nutzer sind sich dieser Tatsache durchaus bewusst. Sie haben zwar ein genaues Bild von den intendierten Adressaten im Kopf. Umständliche Einstellungen oder Eingabemasken verhindern oft genug, dass nur die Wunschadressaten angesprochen werden. Dieser *context collapse*, dieser Zusammenfall sozialer Gruppen, den die Nutzer im Moment des Nachrichtenschreibens erleben, führt dazu, dass sie sich einer strengen Zensur unterwerfen. Nachrichten werden entweder verworfen oder so umformuliert, dass auch ein unbekanntes Publikum nichts anstößiges an den – unweigerlich – kommunizierten Persönlichkeitsaspekten finden kann. Die Kommunikation droht, ihre Authentizität zu verlieren.

Mit der vorliegenden Arbeit wurde ein Konzept vorgestellt, das sich dem *context collapse* widmet. In kontextzentrischen Sozialen Netzen können Nachrichten nur über geteilte Kontexte ausgetauscht werden. Der Zusammenfall sozialer Gruppen wird also durch das vorgeschlagene Adressschema technisch verhindert. Indem für jeden Kontext dedizierte Nachrichtenkanäle etabliert

werden, entstehen für jede Form sozialer Gruppen implizit separate Kanäle. Damit ist erstmals ein Rahmen geschaffen, der ohne Zutun der Nutzer eine authentische Kommunikation mit verschiedenen Freundeskreisen ermöglicht.

Der Kern dieses Konzepts liegt in der informationszentrischen Adressierung und der generischen Kontextbeschreibung. Zur informationszentrischen Vernetzung werden Nachrichten an Kontextbeschreibungen adressiert; zum Nachrichtenempfang werden Kontexte abonniert. Eine Nachricht wird nur zugestellt, wenn die Kontextbeschreibung des Empfängers eine gewisse Ähnlichkeit zur Kontextbeschreibung der Nachricht übersteigt. Beschrieben werden Kontexte durch Bloom-Filter. Diese spezielle probabilistische Datenstruktur dient zur Beschreibung von Mengen und wird in der Regel zum Optimieren von aufwändigen Suchanfragen eingesetzt. Hier werden mit Bloom-Filtern Kontextinformationen zusammengefasst, damit sie so den Kontext eines Nutzers beschreiben. Da anhand eines Bloom-Filters die Größe der zugrundeliegenden Menge abgeschätzt werden kann, können bekannte Maße zum Vergleich von Mengen direkt auf Bloom-Filtern berechnet werden – etwa die Jaccard-Ähnlichkeit. Damit ist ein Vergleich von Kontexten möglich, ohne dass Kontextinformationen offengelegt oder ihre Modalitäten im Voraus bekannt sein müssten. Das schützt die Privatsphäre der Nutzer und macht dAs Adressierungsschema für generelle kontextbasierte Vernetzung einsetzbar.

Im Rahmen dieser Arbeit sind zwei Erweiterungen von Bloom-Filtern entstanden, die im Kapitel 3 vorgestellt wurden. Die Gaußschen Bloom-Filter können Suchanfragen im Vergleich zum klassischen Bloom-Filter weiter optimieren, da sie noch weniger falschpositive Antworten liefern. Erreicht wird das, indem zu jeder Hash-Operation kodiert wird, welche Hash-Funktion verwendet wird. Beim Einfügen von Elementen in den Filter wird die verursachende Hash-Funktion jedes Bits in den Filter hineinkodiert. Bei Suchanfragen kann so entschieden werden, ob ein Bit im Filter von der aktuell verwendeten Hash-Funktion gesetzt worden ist.

Mit der zweiten Erweiterung des ursprünglichen Bloom-Filters konnte eine hierarchische Struktur unter den Elementen des Universums auf das eindimensionale Bit-Array des Bloom-Filters abgebildet werden. Die Hierarchie wurde von OVSF-Code-Bäumen repräsentiert, die normalerweise zum Spreizen von Funksignalen eingesetzt werden. Dank OVSF-Codes können in UMTS mehrere Benutzer gleichzeitig Daten über das Mobilfunknetz herunterladen. Ihre speziellen Eigenschaften hinsichtlich Generalisierbarkeit und gegenseitiger Orthogonalität wurden genutzt, um über die Auswahl von Hash-Funktionen zu entscheiden. Elemente können nun an Orten innerhalb der Hierarchie in einen Bloom-Filter eingefügt und dort auch gesucht werden, indem der entsprechende OVSF-Code eine eindeutige Kombination von Hash-Funktionen erzeugt.

Beide Erweiterungen sind auch für die Weiterentwicklung des kontextzentrischen Adressschemas interessant. Optimierte Suchanfragen mit Gaußschen Bloom-Filtern können eine bessere Vernetzung auf der Basis von Stichwortsuchen mit bestimmten Kontextinformationen ermöglichen – etwa, wenn nur

bestimmte Personen einer sozialen Gruppe adressiert werden. Die hierarchische Kodierung von Informationen könnte wiederum interessant sein, wenn bestimmte Kontextinformationen nur an bestimmten geografischen Orten gesucht werden.

Dass die Vernetzung auch mit traditionellen Bloom-Filtern gelingt, konnte in Kapitel 5 gezeigt werden. Dort wurden zwei Sorten von Datenerhebungen für Soziale Netze vorgestellt. Im Abschnitt 5.1 wurden graphtheoretische Methoden vorgestellt, die üblicherweise zur Analyse Sozialer Netze eingesetzt werden. Zum Einsatz kommen diese Verfahren in einem simulierten kontextzentrischen Sozialen Netz, das über den öffentlich zugänglichen Enron-Datensatz aufgebaut wurde. Im Vergleich zu dem im Datensatz enthaltenen tatsächlichen Sozialen Graphen zeigt sich, dass die kontextzentrische Vernetzung bei geeigneter Wahl einer Jaccard-Ähnlichkeit als unterer Grenze für den Kontextvergleich vergleichbare Graphstrukturen ausbildet. Nicht nur die Größe des resultierenden sozialen Graphen entsprach der tatsächlichen, es wurden auch dieselben Akteure an zentralen Stellen im Graphen gefunden. Die Ergebnisse eröffnen zudem spannende Perspektiven für weitere Untersuchungen zur Intensität sozialer Beziehungen; ein solches qualitatives Maß fehlt der Sozialen Netzwerkanalyse häufig.

Im Abschnitt 5.2 wurden Methoden zum *Data Mining* vorgestellt. Diese Verfahren eignen sich besonders für die Erhebung personenbezogener Daten, weil sie Garantien an die Privatsphäre der betroffenen Nutzer geben. Aus dem Ergebnis der Erhebungen sind keine Rückschlüsse auf einzelne Personen mehr möglich. Im Rahmen dieser Arbeit wurde ein Verfahren entwickelt, das diese privatsphäreschonende Datenerhebung in verteilten Systemen wie dem beschriebenen kontextzentrischen Sozialen Netz ermöglicht. Im Gegensatz zu bekannten Verfahren auf diesem Gebiet konnte ein verständlicheres Maß für den Grad der Privatsphäre der Nutzer vorgestellt werden. Für den Einsatz in kontextzentrischen Netzen ist das Verfahren interessant, weil damit personenbezogene Daten kontextabhängig und mit garantiertem Schutz der Privatsphäre der Nutzer erhoben werden können.

Wenn das Konzept der kontextzentrischen Vernetzung auch in zahlreichen Facetten beleuchtet wurde, ist es mitnichten erschöpfend untersucht. Die oben beschriebene Analyse am Beispiel des Enron-Datensatzes kommt einer Metaanalyse des sozialen Beziehungsgeflechts gleich. Damit eine Metaanalyse zur Bewertung der sozialen Bindungen sinnvolle Aussagen zulässt, ist es notwendig, dass zur Adressierung geeignete Kontextinformationen verwendet werden. Zur Auswahl dieser Kontextinformationen sind einige grundlegende Untersuchungen notwendig. Zum einen stellt sich die Frage, welche Kontextinformationen grundsätzlich geeignet sind. Im Bereich Sozialer Netze bieten sich zur Vernetzung beispielsweise semikonstante Informationen wie Lieblingsfilme oder -bücher, Interessen und Hobbys an. Die durch Facebook bekannten „likes“, mit denen die positive Einstellung gegenüber Themen, Links oder Statusnachrichten anderer Nutzer ausgedrückt wird, könnten ebenso als Eingabedaten zur

Adressierung dienen. Alternativ bieten sich sogenannte „Hashtags“ an. Das sind speziell gekennzeichnete Schlagwörter, die das Thema oder eine Bewertung einer Statusnachricht enthalten oder implizieren.

Grundsätzlich spannend ist auch die Frage, wie die Adressierung heterogene Kontextbeschreibungen unterstüzt – wenn also Profilinformationen und Kontextinformationen mit kürzerer Gültigkeit kombiniert werden. Diese Kontextinformationen können Sensormessungen sein oder sich auf Inhalte oder Funktionen innerhalb von Anwendungen beziehen: Anbieter personalisierter Inhalte können etwa diejenigen Nutzer miteinander vernetzen, die zu einem gegebenen Zeitpunkt dieselben individuellen Inhalte präsentiert bekommen oder einer ähnlichen Aktivität nachgehen.

Neben diesen Untersuchungen mit sozialem Bezug ergeben sich auch Fragestellungen im Bereich des *Internet of Things*. So ist es denkbar, dass Fahrzeuge untereinander Stauinformationen erheben oder Streaming-Inhalte mit lokaler Relevanz zwischenspeichern. Spannende Fragestellungen ergeben sich, wenn neben der vergangenen auch die zukünfigten Positionen in die Kontextbeschreibung mit einfließen. Diese stehen zum Beispiel über das Navigationssystem zur Verfügung.Dann ist in diesem Szenario eine Erhebung über das erwartete Fahrzeugaufkommen entlang der Strecke denkbar. Indem die Fahrzeuge sich darüber austauschen, wo sie wann vorbeifahren werden, können Staus womöglich vorzeitig verhindert werden.

Vor allem ist aber zu untersuchen, wie die Typisierung und Verfeinerung von Kontextinformationen über das vorgeschlagene Adressierungsschema abgebildet werden können. Eine Möglichkeit besteht darin, alle Typenbeschreibungen oder Kategorien einer Kontextinformation mit in die Kontextbeschreibung aufzunehmen. Alternativ ist es denkbar, ein spezielles Hash-Verfahren zu entwickeln, wonach typabhängig die Menge der eingesetzten Hash-Funktionen variiert wird. Ob dazu auf eine externe Modellierung der Hierarchie wie bei OVSF-kodierten Bloom-Filtern zurückgegriffen werden muss oder ob die relevanten Hash-Funktionen auch direkt aus der jeweiligen Typbeschreibung abgeleitet werden können, ist eine interessante Fragestellung.

Spannende Anküpfungspunkte bietet auch die Beweisbarkeit geteilter Kontexte. Können sich Nutzer zu einem späteren Zeitpunkt gegenseitig beweisen, dass sie bereits einen Kontext miteinander geteilt haben und welcher Kontext das war, eröffnen sich zahlreiche Anwendungsfälle im Bereich der *smart contracts*: Dann können über diese Architektur „Verträge“ geschlossen werden – etwa über den Verkauf digitaler Güter. Für den Kontextbeweis kann eine öffentlich einsehbare Blockchain eingesetzt werden, wie es bei Bitcoin ([235]) der Fall ist.

Auf der Ebene der Nutzer Sozialer Netze kann das hier vorgestellte Konzept qualitativ evaluiert werden. Schließlich ist zu klären, ob das Konzept geeignet ist, Selbstzensur zu verhindern und authentischere Kommunikation in Sozialen Netzen zu ermöglichen.

# Literaturverzeichnis


[1] J. Zirfas und B. Jörissen, „Phänomenologien der Identität: Human-, sozial- und kultur-wissenschaftliche Analysen“, in. Wiesbaden: VS Verlag für Sozialwissenschaften, 2007, Kap. Medialitäten und Technologien, S. 157–203, ISBN: 978-3-531-90676-8. DOI: 10.1007/978-3-531-90676-8_5.

[2] M. Rath, „Echtheit, Wahrheit, Ehrlichkeit“, in. Weinheim: Beltz Verlagsgruppe, 2013, Kap. Authentizität als Eigensein und Konstruktion - Überlegungen zur Wahrhaftigkeit in der computervermittelten Kommunikation, S. 16–27, ISBN: 9783779930013.

[3] H.-P. Frey und K. Hausser, *Identität : Entwicklungen psychologischer u. soziologischer Forschung*, Ser. Der Mensch als soziales und personales Wesen. Stuttgart: Enke, 1987, ISBN: 3-432-96401-3.

[4] H. Scherer und W. Wirth, „Ich chatte-wer bin ich? Identitat und Selbstdarstellung in virtuellen Kommunikationssituationen“, *Medien und Kommunikationswissenschaft*, Bd. 50, Nr. 3, S. 337–358, 2003.

[5] G. H. Mead, *Geist, Identität und Gesellschaft - Aus der Sicht des Sozialbehaviorismus*. Frankfurt: Suhrkamp, 1973, ISBN: 978-3-518-27628-0.

[6] B. R. Schlenker, *Impression Management: The Self-concept, Social Identity, and Interpersonal Relations*. Brooks/Cole Publishing Company, 1980, ISBN: 9780818503986.

[7] J. Vitak, „The Impact of Context Collapse and Privacy on Social Network Site Disclosures“, *Journal of Broadcasting & Electronic Media*, Bd. 56, Nr. 4, S. 451–470, 2012. DOI: 10.1080/08838151.2012.732140. eprint: http://dx.doi.org/10.1080/08838151.2012.732140.

[8] E. Goffman, *The presentation of self in everyday life*. New York: Doubleday, 1959.

[9] B. R. Schlenker, „Identity and self-identification“, *The self and social life*, Bd. 65, S. 99, 1985.

[10] M. R. Leary, *Self-presentation: Impression management and interpersonal behavior*. Brown & Benchmark Publishers, 1995.

[11] N. Haferkamp, „StudiVZ: Diffusion, Nutzung und Wirkung eines sozialen Netzwerks im Internet“, in, C. Neuberger und V. Gehrau, Hrsg. Wiesbaden: VS Verlag für Sozialwissenschaften, 2011, Kap. Authentische Selbstbilder, geschönte Fremdbilder, S. 178–203, ISBN: 978-3-531-93096-1. DOI: 10.1007/978-3-531-93096-1_8.

[12] D. M. Boyd, *Taken out of context: American teen sociality in networked publics*. ProQuest, 2008.

[13] J. Binder, A. Howes und A. Sutcliffe, „The Problem of Conflicting Social Spheres: Effects of Network Structure on Experienced Tension in Social Network Sites“, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Ser. CHI '09, Boston, MA, USA: ACM, 2009, S. 965–974, ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518849.

[14] A. Lampinen, S. Tamminen und A. Oulasvirta, „All My People Right Here, Right Now: Management of Group Co-presence on a Social Networking Site“, in *Proceedings of the ACM 2009 International Conference on Supporting Group Work*, Ser. GROUP '09, Sanibel Island, Florida, USA: ACM, 2009, S. 281–290, ISBN: 978-1-60558-500-0. DOI: 10.1145/1531674.1531717.

[15] M. Sleeper, R. Balebako, S. Das, A. L. McConahy, J. Wiese und L. F. Cranor, „The Post That Wasn'T: Exploring Self-censorship on Facebook“, in *Proceedings of the 2013 Conference on Computer Supported Cooperative Work*, Ser. CSCW '13, San Antonio, Texas, USA: ACM, 2013, S. 793–802, ISBN: 978-1-4503-1331-5. DOI: 10.1145/2441776.2441865.

[16] J. Watson, A. Besmer und H. R. Lipford, „+Your Circles: Sharing Behavior on Google+“, in *Proceedings of the Eighth Symposium on Usable Privacy and Security*, Ser. SOUPS '12, Washington, D.C.: ACM, 2012, 12:1–12:9, ISBN: 978-1-4503-1532-6. DOI: 10.1145/2335356.2335373.

[17] J. Vitak, S. Blasiola, S. Patil und E. Litt, „Balancing Audience and Privacy Tensions on Social Network Sites: Strategies of Highly Engaged Users“, *International Journal of Communication*, Bd. 9, 2015.

[18] P. Karr-Wisniewski, D. Wilson und H. Richter-Lipford, „A new social order: Mechanisms for social network site boundary regulation“, in *Proceedings of the 17th Americas Conference on Information Systems,*, Detroit, Michigan, USA, 2011.

[19] J. Vitak und J. Kim, „"You Can'T Block People Offline": Examining How Facebook's Affordances Shape the Disclosure Process“, in *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, Ser. CSCW '14, Baltimore, Maryland, USA: ACM, 2014, S. 461–474, ISBN: 978-1-4503-2540-0. DOI: 10.1145/2531602.2531672.

[20] E. Litt und E. Hargittai, „“Just Cast the Net, and Hopefully the Right Fish Swim into It”: Audience Management on Social Network Sites“, in *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, Ser. CSCW '16, San Francisco,

California, USA: ACM, 2016, S. 1488–1500, ISBN: 978-1-4503-3592-8. DOI: 10.1145/2818048.2819933.

[21] S. Utz und N. Kramer, „The privacy paradox on social network sites revisited: The role of individual characteristics and group norms“, *Cyberpsychology: Journal of Psychosocial Research on Cyberspace*, Bd. 3, Nr. 2, S. 2, 2009.

[22] K. Strater und H. R. Lipford, „Strategies and Struggles with Privacy in an Online Social Networking Community“, in *Proceedings of the 22Nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction - Volume 1*, Ser. BCS-HCI '08, Liverpool, United Kingdom: British Computer Society, 2008, S. 111–119, ISBN: 978-1-906124-04-5.

[23] D. M. Boyd und N. B. Ellison, „Social Network Sites: Definition, History, and Scholarship“, *Journal of Computer-Mediated Communication*, Bd. 13, Nr. 1, S. 210–230, 2007, ISSN: 1083-6101. DOI: 10.1111/j.1083-6101.2007.00393.x.

[24] M. Durr, M. Werner und M. Maier, „Re-Socializing Online Social Networks“, in *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, Dez. 2010, S. 786–791. DOI: 10.1109/GreenCom-CPSCom.2010.18.

[25] W. N. Schilit, „A system architecture for context-aware mobile computing“, Diss., Columbia University, 1995.

[26] A. K. Dey, „Understanding and Using Context“, *Personal Ubiquitous Comput.*, Bd. 5, Nr. 1, S. 4–7, Jan. 2001, ISSN: 1617-4909. DOI: 10.1007/s007790170019.

[27] D. Franklin und J. Flaschbart, „All gadget and no representation makes jack a dull environment“, in *Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments*, 1998, S. 155–160.

[28] R. Hull, P. Neaves und J. Bedford-Roberts, „Towards situated computing“, in *Wearable Computers, 1997. Digest of Papers., First International Symposium on*, Okt. 1997, S. 146–153. DOI: 10.1109/ISWC.1997.629931.

[29] P. Brown, J. Bovey und X. Chen, „Context-aware applications: from the laboratory to the marketplace“, *Personal Communications, IEEE*, Bd. 4, Nr. 5, S. 58–64, Okt. 1997, ISSN: 1070-9916. DOI: 10.1109/98.626984.

[30] A. Jameson, „Modelling both the Context and the User“, *Personal and Ubiquitous Computing*, Bd. 5, Nr. 1, S. 29–33, ISSN: 1617-4909. DOI: 10.1007/s007790170025.

[31] J. Strassner und D. O'Sullivan, „Knowledge Management for Context-aware, Policy-based Ubiquitous Computing Systems", in *Proceedings of the 6th International Workshop on Managing Ubiquitous Communications and Services*, Ser. MUCS '09, Barcelona, Spain: ACM, 2009, S. 67–76, ISBN: 978-1-60558-579-6. DOI: 10.1145/1555321.1555335.

[32] R. Sterritt, M. Mulvenna und A. Lawrynowicz, „Autonomic Communication: First International IFIP Workshop, WAC 2004, Berlin, Germany, October 18-19, 2004, Revised Selected Papers", in, M. Smirnov, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, Kap. Dynamic and Contextualised Behavioural Knowledge in Autonomic Communications, S. 217–228, ISBN: 978-3-540-32009-8. DOI: 10.1007/11520184_17.

[33] J. Strassner, S. Meer, D. O'Sullivan und S. Dobson, „The Use of Context-Aware Policies and Ontologies to Facilitate Business-Aware Network Management", *Journal of Network and Systems Management*, Bd. 17, Nr. 3, S. 255–284, 2009, ISSN: 1573-7705. DOI: 10.1007/s10922-009-9126-4.

[34] K. Wrona und L. Gomez, „Context-aware security and secure context-awareness in ubiquitous computing environments", in *Polish Information Processing Society (PIPS), 21 Conference on*, 2005, S. 255–265.

[35] D. Schuster, A. Rosi, M. Mamei, T. Springer, M. Endler und F. Zambonelli, „Pervasive Social Context: Taxonomy and Survey", *ACM Trans. Intell. Syst. Technol.*, Bd. 4, Nr. 3, 46:1–46:22, Juli 2013, ISSN: 2157-6904. DOI: 10.1145/2483669.2483679.

[36] P. Bellavista, A. Corradi, M. Fanelli und L. Foschini, „A Survey of Context Data Distribution for Mobile Ubiquitous Systems", *ACM Comput. Surv.*, Bd. 44, Nr. 4, 24:1–24:45, Sep. 2012, ISSN: 0360-0300. DOI: 10.1145/2333112.2333119.

[37] B. Adams, D. Phung und S. Venkatesh, „Sensing and Using Social Context", *ACM Trans. Multimedia Comput. Commun. Appl.*, Bd. 5, Nr. 2, 11:1–11:27, Nov. 2008, ISSN: 1551-6857. DOI: 10.1145/1413862.1413864.

[38] M. Endler, A. Skyrme, D. Schuster und T. Springer, „Defining Situated Social Context for pervasive social computing", in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, März 2011, S. 519–524. DOI: 10.1109/PERCOMW.2011.5766945.

[39] S. Scellato, A. Noulas, R. Lambiotte und C. Mascolo, „Socio-Spatial Properties of Online Location-Based Social Networks", in *Proceedings of the 5th International AAAI Conference on Web and Social Media*, Barcelona, Spain, 2011, S. 329–336.

[40] J. Cranshaw, E. Toch, J. Hong, A. Kittur und N. Sadeh, „Bridging the Gap Between Physical Location and Online Social Networks“, in *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, Ser. UbiComp '10, Copenhagen, Denmark: ACM, 2010, S. 119–128, ISBN: 978-1-60558-843-8. DOI: 10.1145/1864349.1864380.

[41] A. Rosi, M. Mamei und F. Zambonelli, „Integrating social sensors and pervasive services: approaches and perspectives“, *International Journal of Pervasive Computing and Communications*, Bd. 9, Nr. 4, S. 294–310, 2013. DOI: 10.1108/IJPCC-09-2013-0022.

[42] S. Michalakos und I. T. Christou, „G2G: Location-aware Mobile Social Networking with Applications in Recommender Systems and Gaming“, in *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*, Ser. MoMM '08, Linz, Austria: ACM, 2008, S. 163–169, ISBN: 978-1-60558-269-6. DOI: 10.1145/1497185.1497221.

[43] A. Beach, M. Gartrell, S. Akkala, J. Elston, J. Kelley, K. Nishimoto, B. Ray, S. Razgulin, K. Sundaresan, B. Surendar, M. Terada und R. Han, „WhozThat? evolving an ecosystem for context-aware mobile social networks“, *Network, IEEE*, Bd. 22, Nr. 4, S. 50–55, Juli 2008, ISSN: 0890-8044. DOI: 10.1109/MNET.2008.4579771.

[44] A. Gupta, M. Miettinen, M. Nagy, N. Asokan und A. Wetzel, „PeerSense: Who is near you?“, *Pervasive Computing and Communications Workshops, IEEE International Conference on*, S. 516–518, 2012. DOI: http://doi.ieeecomputersociety.org/10.1109/PerComW.2012.6197553.

[45] A. Zunjarwad, H. Sundaram und L. Xie, „Contextual Wisdom: Social Relations and Correlations for Multimedia Event Annotation“, in *Proceedings of the 15th ACM International Conference on Multimedia*, Ser. MM '07, Augsburg, Germany: ACM, 2007, S. 615–624, ISBN: 978-1-59593-702-5. DOI: 10.1145/1291233.1291382.

[46] B. Shevade, H. Sundaram und L. Xie, „Modeling Personal and Social Network Context for Event Annotation in Images“, in *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries*, Ser. JCDL '07, Vancouver, BC, Canada: ACM, 2007, S. 127–134, ISBN: 978-1-59593-644-8. DOI: 10.1145/1255175.1255200.

[47] P. Shankar, Y.-W. Huang, P. Castro, B. Nath und L. Iftode, „Crowds replace experts: Building better location-based services using mobile social network interactions“, in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, März 2012, S. 20–29. DOI: 10.1109/PerCom.2012.6199845.

[48] X. Hu, X. Li, E.-H. Ngai, V. Leung und P. Kruchten, „Multidimensional context-aware social network architecture for mobile crowdsensing“, *Communications Magazine, IEEE*, Bd. 52, Nr. 6, S. 78–87, Juni 2014, ISSN: 0163-6804. DOI: 10.1109/MCOM.2014.6829948.

[49] M. Sarwat, J. Bao, A. Eldawy, J. J. Levandoski, A. Magdy und M. F. Mokbel, „Sindbad: A Location-based Social Networking System“, in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, Ser. SIGMOD '12, Scottsdale, Arizona, USA: ACM, 2012, S. 649–652, ISBN: 978-1-4503-1247-9. DOI: 10.1145/2213836.2213923.

[50] M. Demirbas, M. Bayir, C. Akcora, Y. Yilmaz und H. Ferhatosmanoglu, „Crowd-sourced sensing and collaboration using twitter“, in *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*, Juni 2010, S. 1–9. DOI: 10.1109/WOWMOM.2010.5534910.

[51] M. Baqer und A. Kamal, „S-Sensors: Integrating physical world inputs with social networks using wireless sensor networks“, in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on*, Dez. 2009, S. 213–218. DOI: 10.1109/ISSNIP.2009.5416815.

[52] M. Treiber, D. Schall, S. Dustdar und C. Scherling, „Tweetflows: Flexible Workflows with Twitter“, in *Proceedings of the 3rd International Workshop on Principles of Engineering Service-Oriented Systems*, Ser. PESOS '11, Waikiki, Honolulu, HI, USA: ACM, 2011, S. 1–7, ISBN: 978-1-4503-0591-4. DOI: 10.1145/1985394.1985395.

[53] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng und A. T. Campbell, „Sensing Meets Mobile Social Networks: The Design, Implementation and Evaluation of the CenceMe Application“, in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, Ser. SenSys '08, Raleigh, NC, USA: ACM, 2008, S. 337–350, ISBN: 978-1-59593-990-6. DOI: 10.1145/1460412.1460445.

[54] E. Miluzzo, N. D. Lane, S. B. Eisenman und A. T. Campbell, „Smart Sensing and Context: Second European Conference, EuroSSC 2007, Kendal, England, October 23-25, 2007. Proceedings“, in, G. Kortuem, J. Finney, R. Lea und V. Sundramoorthy, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, Kap. CenceMe – Injecting Sensing Presence into Social Networking Applications, S. 1–28, ISBN: 978-3-540-75696-5. DOI: 10.1007/978-3-540-75696-5_1.

[55] M. Pitkänen, T. Kärkkäinen, J. Ott, M. Conti, A. Passarella, S. Giordano, D. Puccinelli, F. Legendre, S. Trifunovic, K. Hummel, M. May, N. Hegde und T. Spyropoulos, „SCAMPI: Service Platform for Social Aware Mobile and Pervasive Computing“, *SIGCOMM Comput. Commun. Rev.*, Bd. 42, Nr. 4, S. 503–508, Sep. 2012, ISSN: 0146-4833. DOI: 10.1145/2377677.2377775.

[56] X. Chen, B. Proulx, X. Gong und J. Zhang, „Social Trust and Social Reciprocity Based Cooperative D2D Communications“, in *Proceedings of the Fourteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Ser. MobiHoc '13, Bangalore, India: ACM, 2013, S. 187–196, ISBN: 978-1-4503-2193-8. DOI: 10.1145/2491288.2491302.

[57] Y. Zhang, L. Song, W. Saad, Z. Dawy und Z. Han, „Exploring social ties for enhanced device-to-device communications in wireless networks“, in *Global Communications Conference (GLOBECOM), 2013 IEEE*, Dez. 2013, S. 4597–4602. DOI: 10.1109/GLOCOMW.2013.6855676.

[58] C. Boldrini, M. Conti, F. Delmastro und A. Passarella, „Context- and social-aware middleware for opportunistic networks“, *Journal of Network and Computer Applications*, Bd. 33, Nr. 5, S. 525–541, 2010, Middleware Trends for Network Applications, ISSN: 1084-8045. DOI: http://dx.doi.org/10.1016/j.jnca.2010.03.017.

[59] W. Gao und G. Cao, „User-centric data dissemination in disruption tolerant networks“, in *INFOCOM, 2011 Proceedings IEEE*, Apr. 2011, S. 3119–3127. DOI: 10.1109/INFCOM.2011.5935157.

[60] A. Mohaisen, E. Y. Vasserman, M. Schuchard, D. Foo Kune und Y. Kim, „Secure Encounter-based Social Networks: Requirements, Challenges, and Designs“, in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, Ser. CCS '10, Chicago, Illinois, USA: ACM, 2010, S. 717–719, ISBN: 978-1-4503-0245-6. DOI: 10.1145/1866307.1866408.

[61] A. Mohaien, D. Kune, E. Vasserman, M. Kim und Y. Kim, „Secure Encounter-Based Mobile Social Networks: Requirements, Designs, and Tradeoffs“, *Dependable and Secure Computing, IEEE Transactions on*, Bd. 10, Nr. 6, S. 380–393, Nov. 2013, ISSN: 1545-5971. DOI: 10.1109/TDSC.2013.19.

[62] Z. Yang, B. Zhang, J. Dai, A. Champion, D. Xuan und D. Li, „E-SmallTalker: A Distributed Mobile System for Social Networking in Physical Proximity“, in *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, Juni 2010, S. 468–477. DOI: 10.1109/ICDCS.2010.56.

[63] N. Eagle und A. Pentland, „Social serendipity: mobilizing social software“, *Pervasive Computing, IEEE*, Bd. 4, Nr. 2, S. 28–34, Jan. 2005, ISSN: 1536-1268. DOI: 10.1109/MPRV.2005.37.

[64] M. Durr, M. Maier und F. Dorfmeister, „Vegas – A Secure and Privacy-Preserving Peer-to-Peer Online Social Network“, in *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, Sep. 2012, S. 868–874. DOI: 10.1109/SocialCom-PASSAT.2012.42.

[65] N. Jabeur, S. Zeadally und B. Sayed, „Mobile Social Networking Applications“, *Commun. ACM*, Bd. 56, Nr. 3, S. 71–79, März 2013, ISSN: 0001-0782. DOI: 10.1145/2428556.2428573.

[66] J. Li, H. Wang und S. U. Khan, „A Semantics-based Approach to Large-Scale Mobile Social Networking“, *Mobile Networks and Applications*, Bd. 17, Nr. 2, S. 192–205, 2011, ISSN: 1572-8153. DOI: 10.1007/s11036-011-0330-6.

[67] J. Rana, J. Kristiansson, J. Hallberg und K. Synnes, „An Architecture for Mobile Social Networking Applications“, in *Computational Intelligence, Communication Systems and Networks, 2009. CICSYN '09. First International Conference on*, Juli 2009, S. 241–246. DOI: 10.1109/CICSYN.2009.73.

[68] A. Beach, M. Gartrell, X. Xing, R. Han, Q. Lv, S. Mishra und K. Seada, „Fusing Mobile, Sensor, and Social Data to Fully Enable Context-aware Computing“, in *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*, Ser. HotMobile '10, Annapolis, Maryland: ACM, 2010, S. 60–65, ISBN: 978-1-4503-0005-6. DOI: 10.1145/1734583.1734599.

[69] X. Hu, T. Chu, V. Leung, E.-H. Ngai, P. Kruchten und H. Chan, „A Survey on Mobile Social Networks: Applications, Platforms, System Architectures, and Future Research Directions“, *Communications Surveys Tutorials, IEEE*, Bd. 17, Nr. 3, S. 1557–1581, 2015, ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2371813.

[70] N. Kayastha, D. Niyato, P. Wang und E. Hossain, „Applications, Architectures, and Protocol Design Issues for Mobile Social Networks: A Survey“, *Proceedings of the IEEE*, Bd. 99, Nr. 12, S. 2130–2158, Dez. 2011, ISSN: 0018-9219. DOI: 10.1109/JPROC.2011.2169033.

[71] G. Chen und F. Rahman, „Analyzing Privacy Designs of Mobile Social Networking Applications“, in *Embedded and Ubiquitous Computing, 2008. EUC '08. IEEE/IFIP International Conference on*, Bd. 2, Dez. 2008, S. 83–88. DOI: 10.1109/EUC.2008.156.

[72] A. Teles, F. J. Silva und R. Batista, „Security and Privacy Preserving in Social Networks“, in, R. Chbeir und B. Al Bouna, Hrsg. Vienna: Springer Vienna, 2013, Kap. Security and Privacy Issues in Mobile Social Networks, S. 281–313, ISBN: 978-3-7091-0894-9. DOI: 10.1007/978-3-7091-0894-9_9.

[73] A. Beach, M. Gartrell und R. Han, „Solutions to Security and Privacy Issues in Mobile Social Networking“, in *Computational Science and Engineering, 2009. CSE '09. International Conference on*, Bd. 4, Aug. 2009, S. 1036–1042. DOI: 10.1109/CSE.2009.243.

[74] B. H. Bloom, „Space/time trade-offs in hash coding with allowable errors“, *Communications of the ACM*, Bd. 13, Nr. 7, S. 422–426, 1970.

[75] P. Bose, H. Guo, E. Kranakis, A. Maheshwari, P. Morin, J. Morrison, M. Smid und Y. Tang, „On the false-positive rate of Bloom filters“, *Information Processing Letters*, Bd. 108, Nr. 4, S. 210–213, 2008, ISSN: 0020-0190. DOI: http://dx.doi.org/10.1016/j.ipl.2008.05.018.

[76] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes und R. E. Gruber, „Bigtable: A Distributed Storage System for Structured Data“, *ACM Trans. Comput. Syst.*, Bd. 26, Nr. 2, 4:1–4:26, Juni 2008, ISSN: 0734-2071. DOI: 10.1145/1365815.1365816.

[77] K. Shvachko, H. Kuang, S. Radia und R. Chansler, „The Hadoop Distributed File System“, in *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, Mai 2010, S. 1–10. DOI: 10.1109/MSST.2010.5496972.

[78] A. Lakshman und P. Malik, „Cassandra: A Decentralized Structured Storage System“, *SIGOPS Oper. Syst. Rev.*, Bd. 44, Nr. 2, S. 35–40, Apr. 2010, ISSN: 0163-5980. DOI: 10.1145/1773912.1773922.

[79] L. Fan, P. Cao, J. Almeida und A. Z. Broder, „Summary Cache: A Scalable Wide-area Web Cache Sharing Protocol“, *IEEE/ACM Trans. Netw.*, Bd. 8, Nr. 3, S. 281–293, Juni 2000, ISSN: 1063-6692. DOI: 10.1109/90.851975.

[80] F. Bonomi, M. Mitzenmacher, R. Panigrahy, S. Singh und G. Varghese, „Algorithms – ESA 2006: 14th Annual European Symposium, Zurich, Switzerland, September 11-13, 2006. Proceedings“, in, Y. Azar und T. Erlebach, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, Kap. An Improved Construction for Counting Bloom Filters, S. 684–695, ISBN: 978-3-540-38876-0. DOI: 10.1007/11841036_61.

[81] S. Cohen und Y. Matias, „Spectral Bloom Filters“, in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, Ser. SIGMOD '03, San Diego, California: ACM, 2003, S. 241–252, ISBN: 1-58113-634-X. DOI: 10.1145/872757.872787.

[82] K. Cheng und L. Xiang, „Time-decaying Bloom Filters for data streams with skewed distributions“, in *Research Issues in Data Engineering: Stream Data Mining and Applications, 2005. RIDE-SDMA 2005. 15th International Workshop on*, Apr. 2005, S. 63–69. DOI: 10.1109/RIDE.2005.15.

[83] P. S. Almeida, C. Baquero, N. Preguiça und D. Hutchison, „Scalable Bloom Filters“, *Information Processing Letters*, Bd. 101, Nr. 6, S. 255–261, 2007, ISSN: 0020-0190. DOI: http://dx.doi.org/10.1016/j.ipl.2006.10.007.

[84] F. Hao, M. Kodialam und T. V. Lakshman, „Building High Accuracy Bloom Filters Using Partitioned Hashing“, in *Proceedings of the 2007 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, Ser. SIGMETRICS '07, San Diego, California, USA: ACM, 2007, S. 277–288, ISBN: 978-1-59593-639-4. DOI: 10.1145/1254882.1254916.

[85] S. Lumetta und M. Mitzenmacher, „Using the Power of Two Choices to Improve Bloom Filters“, *Internet Mathematics*, Bd. 4, Nr. 1, S. 17–33, 2007. DOI: 10.1080/15427951.2007.10129136.

[86] A. Kumar, J. ( Xu, L. Li und J. Wang, „Space-code Bloom Filter for Efficient Traffic Flow Measurement“, in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, Ser. IMC '03, Miami Beach, FL, USA: ACM, 2003, S. 167–172, ISBN: 1-58113-773-7. DOI: 10.1145/948205.948226.

[87] D. Eastlake und P. Jones, *US Secure Hash Algorithm 1 (SHA1)*, 2001.

[88] A. Appleby, *Murmurhash*, http://code.google.com/p/smhasher/, 2009.

[89] N. N. Schraudolph, „A fast, compact approximation of the exponential function“, *Neural Computation*, Bd. 11, Nr. 4, S. 853–862, 1999.

[90] J. Bruck, J. Gao und A. Jiang, „Weighted Bloom filter“, in *Information Theory, 2006 IEEE International Symposium on*, Juli 2006. DOI: 10.1109/ISIT.2006.261978.

[91] K. Shanmugasundaram, H. Brönnimann und N. Memon, „Payload Attribution via Hierarchical Bloom Filters“, in *Proceedings of the 11th ACM Conference on Computer and Communications Security*, Ser. CCS '04, Washington DC, USA: ACM, 2004, S. 31–41, ISBN: 1-58113-961-6. DOI: 10.1145/1030083.1030089.

[92] A. Kirsch und M. Mitzenmacher, „Distance-Sensitive Bloom Filters“, in *2006 Proceedings of the Eighth Workshop on Algorithm Engineering and Experiments (ALENEX)*, Kap. 3, S. 41–50. DOI: 10.1137/1.9781611972863.4. eprint: http://epubs.siam.org/doi/pdf/10.1137/1.9781611972863.4.

[93] A. Gionis, P. Indyk und R. Motwani, „Similarity Search in High Dimensions via Hashing“, in *Proceedings of the 25th International Conference on Very Large Data Bases*, Ser. VLDB '99, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, S. 518–529, ISBN: 1-55860-615-7.

[94] A. Andoni und P. Indyk, „Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions“, in *Foundations of Computer Science, 2006. FOCS '06. 47th Annual IEEE Symposium on*, Okt. 2006, S. 459–468. DOI: 10.1109/FOCS.2006.49.

[95] S. Tarkoma, C. Rothenberg und E. Lagerspetz, „Theory and Practice of Bloom Filters for Distributed Systems“, *Communications Surveys Tutorials, IEEE*, Bd. 14, Nr. 1, S. 131–155, Jan. 2012, ISSN: 1553-877X. DOI: 10.1109/SURV.2011.031611.00024.

[96] A. Broder und M. Mitzenmacher, „Network Applications of Bloom Filters: A Survey“, *Internet Mathematics*, Bd. 1, Nr. 4, S. 485–509, 2004. DOI: 10.1080/15427951.2004.10129096.

[97] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher und B. Ohlman, „A survey of information-centric networking“, *Communications Magazine, IEEE*, Bd. 50, Nr. 7, S. 26–36, Juli 2012, ISSN: 0163-6804. DOI: 10.1109/MCOM.2012.6231276.

[98] Z. Jerzak und C. Fetzer, „Bloom Filter Based Routing for Content-based Publish/Subscribe“, in *Proceedings of the Second International Conference on Distributed Event-based Systems*, Ser. DEBS '08, Rome, Italy: ACM, 2008, S. 71–81, ISBN: 978-1-60558-090-6. DOI: 10.1145/1385989.1385999.

[99] M. Mitzenmacher, „Compressed Bloom Filters“, *IEEE/ACM Trans. Netw.*, Bd. 10, Nr. 5, S. 604–612, Okt. 2002, ISSN: 1063-6692. DOI: 10.1109/TNET.2002.803864.

[100] J. Byers, J. Considine und M. Mitzenmacher, „Fast Approximate Reconciliation of Set Differences“, Boston University Computer Science Department, Techn. Ber., 2002.

[101] S. J. Swamidass und P. Baldi, „Mathematical Correction for Fingerprint Similarity Measures to Improve Chemical Retrieval“, *Journal of Chemical Information and Modeling*, Bd. 47, Nr. 3, S. 952–964, 2007. DOI: 10.1021/ci600526a.

[102] M. Werner, „BACR: Set Similarities with Lower Bounds and Application to Spatial Trajectories“, in *23rd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL 2015)*, 2015.

[103] J. Pan, S. Paul und R. Jain, „A survey of the research on future internet architectures“, *Communications Magazine, IEEE*, Bd. 49, Nr. 7, S. 26–36, Juli 2011, ISSN: 0163-6804. DOI: 10.1109/MCOM.2011.5936152.

[104] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker und I. Stoica, „A Data-oriented (and Beyond) Network Architecture“, in *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Ser. SIGCOMM '07, Kyoto, Japan: ACM, 2007, S. 181–192, ISBN: 978-1-59593-713-1. DOI: 10.1145/1282380.1282402.

[105] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs und R. L. Braynard, „Networking Named Content“, in *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*, Ser. CoNEXT '09, Rome, Italy: ACM, 2009, S. 1–12, ISBN: 978-1-60558-636-6. DOI: 10.1145/1658939.1658941.

[106] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren und H. Karl, „Network of Information (NetInf) – An information-centric networking architecture“, *Computer Communications*, Bd. 36, Nr. 7, S. 721–735, 2013, ISSN: 0140-3664. DOI: http://dx.doi.org/10.1016/j.comcom.2013.01.009.

[107] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan und J. Wilcox, „Information-centric Networking: Seeing the Forest for the Trees“, in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, Ser. HotNets-X, Cambridge, Massachusetts: ACM, 2011, S. 1–6, ISBN: 978-1-4503-1059-8. DOI: 10.1145/2070562.2070563.

[108] G. Carofiglio, G. Morabito, L. Muscariello, I. Solis und M. Varvello, „From content delivery today to information centric networking“, *Computer Networks*, Bd. 57, Nr. 16, S. 3116–3127, 2013, Information Centric Networking, ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/j.comnet.2013.07.002.

[109] M. Bari, S. Chowdhury, R. Ahmed, R. Boutaba und B. Mathieu, „A survey of naming and routing in information-centric networks“, *Communications Magazine, IEEE*, Bd. 50, Nr. 12, S. 44–53, Dez. 2012, ISSN: 0163-6804. DOI: 10.1109/MCOM.2012.6384450.

[110] M. Wählisch, T. C. Schmidt und M. Vahlenkamp, „Backscatter from the data plane - Threats to stability and security in information-centric network infrastructure“, *Computer Networks*, Bd. 57, Nr. 16, S. 3192–3206, 2013, Information Centric Networking, ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/j.comnet.2013.07.009.

[111] S. R. Chowdhury, A. R. Roy, M. Shaikh und K. Daudjee, „A taxonomy of decentralized online social networks“, *Peer-to-Peer Networking and Applications*, Bd. 8, Nr. 3, S. 367–383, 2014, ISSN: 1936-6450. DOI: 10.1007/s12083-014-0258-2.

[112] S. Buchegger und A. Datta, „A case for P2P infrastructure for social networks - opportunities & challenges“, in *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, Feb. 2009, S. 161–168. DOI: 10.1109/WONS.2009.4801862.

[113] R. Sharma und A. Datta, „SuperNova: Super-peers based architecture for decentralized online social networks“, in *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, Jan. 2012, S. 1–10. DOI: 10.1109/COMSNETS.2012.6151349.

[114] A. Shakimov, H. Lim, R. Caceres, L. Cox, K. Li, D. Liu und A. Varshavsky, „Vis- à-Vis: Privacy-preserving online social networking via Virtual Individual Servers“, in *Communication Systems and Networks (COMSNETS), 2011 Third International Conference on*, Jan. 2011, S. 1–10. DOI: 10.1109/COMSNETS.2011.5716497.

[115] S.-W. Seong, J. Seo, M. Nasielski, D. Sengupta, S. Hangal, S. K. Teh, R. Chu, B. Dodson und M. S. Lam, „PrPl: A Decentralized Social Networking Infrastructure“, in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*, Ser. MCS '10, San Francisco, California: ACM, 2010, 8:1–8:8, ISBN: 978-1-4503-0155-8. DOI: 10.1145/1810931.1810939.

[116] S. Nilizadeh, S. Jahid, P. Mittal, N. Borisov und A. Kapadia, „Cachet: A Decentralized Architecture for Privacy Preserving Social Networking with Caching“, in *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, Ser. CoNEXT '12, Nice, France: ACM, 2012, S. 337–348, ISBN: 978-1-4503-1775-7. DOI: 10.1145/2413176.2413215.

[117] L. Cutillo, R. Molva und T. Strufe, „Safebook: A privacy-preserving online social network leveraging on real-life trust“, *Communications Magazine, IEEE*, Bd. 47, Nr. 12, S. 94–101, Dez. 2009, ISSN: 0163-6804. DOI: 10.1109/MCOM.2009.5350374.

[118] S. Buchegger, D. Schiöberg, L.-H. Vu und A. Datta, „PeerSoN: P2P Social Networking: Early Experiences and Insights“, in *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, Ser. SNS '09, Nuremberg, Germany: ACM, 2009, S. 46–52, ISBN: 978-1-60558-463-8. DOI: 10.1145/1578002.1578010.

[119] M. Durr, M. Maier und K. Wiesner, „An Analysis of Query Forwarding Strategies for Secure and Privacy-Preserving Social Networks“, in *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, Aug. 2012, S. 535–542. DOI: 10.1109/ASONAM.2012.92.

[120] T. Xu, Y. Chen, J. Zhao und X. Fu, „Cuckoo: Towards Decentralized, Socio-aware Online Microblogging Services and Data Measurements“, in *Proceedings of the 2Nd ACM International Workshop on Hot Topics in Planet-scale Measurement*, Ser. HotPlanet '10, San Francisco, California: ACM, 2010, 4:1–4:6, ISBN: 978-1-4503-0177-0. DOI: 10.1145/1834616.1834622.

[121] D. R. Sandler und D. S. Wallach, „Birds of a FETHR: Open, Decentralized Micropublishing“, in *Proceedings of the 8th International Conference on Peer-to-peer Systems*, Ser. IPTPS'09, Boston, MA: USENIX Association, 2009, S. 1–1.

[122] P. Ruppel und A. Küpper, „Geocookie: A Space-Efficient Representation of Geographic Location Sets“, *Journal of Information Processing*, Bd. 22, Nr. 3, S. 418–424, 2014. DOI: 10.2197/ipsjjip.22.418.

[123] R. Ajami, N. Ramadan, N. Mohamed und J. Al-Jaroodi, „Security challenges and approaches in online social networks: A survey“, *International Journal of Computer Science and Network Security (IJCSNS)*, Bd. 11, Nr. 8, S. 1, 2011.

[124] M. Maier, C. Marouane, M. Klette, F. Dorfmeister, P. Marcus und C. Linnhoff-Popien, „SURFtogether: Towards Context Proximity Detection Using Visual Features“, in *Proceedings of the 3rd International Conference on Context-Aware Systems and Applications*, Ser. ICCASA '14, Dubai, United Arab Emirates: ICST (Institute for Computer Sciences, Social-Informatics und Telecommunications Engineering), 2014, S. 86–91, ISBN: 978-1-63190-005-1. Adresse: http://dl.acm.org/citation.cfm?id=2762722.2762738.

[125] S. Jana, S. N. Premnath, M. Clark, S. K. Kasera, N. Patwari und S. V. Krishnamurthy, „On the Effectiveness of Secret Key Extraction from Wireless Signal Strength in Real Environments“, in *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*, Ser. MobiCom '09, Beijing, China: ACM, 2009, S. 321–332, ISBN: 978-1-60558-702-8. DOI: 10.1145/1614320.1614356.

[126] S. Capkun, K. Bonne Rasmussen, M. Cagalj und M. Srivastava, „Secure Location Verification with Hidden and Mobile Base Stations“, *Mobile Computing, IEEE Transactions on*, Bd. 7, Nr. 4, S. 470–483, Apr. 2008, ISSN: 1536-1233. DOI: 10.1109/TMC.2007.70782.

[127] S. Saroiu und A. Wolman, „Enabling New Mobile Applications with Location Proofs“, in *Proceedings of the 10th Workshop on Mobile Computing Systems and Applications*, Ser. HotMobile '09, Santa Cruz, California: ACM, 2009, 3:1–3:6, ISBN: 978-1-60558-283-2. DOI: 10.1145/1514411.1514414.

[128] M. Maier, L. Schauer und F. Dorfmeister, „ProbeTags: Privacy-preserving proximity detection using Wi-Fi management frames“, in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2015 IEEE 11th International Conference on*, Okt. 2015, S. 756–763. DOI: 10.1109/WiMOB.2015.7348038.

[129] S. Wasserman und K. Faust, *Social network analysis*, Ser. Structural analysis in the social sciences. Cambridge [u.a.]: Cambridge Univ. Press, 2009, ISBN: 9780521382694 - 9780521387071.

[130] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy und B. Y. Zhao, „User Interactions in Social Networks and Their Implications“, in *Proceedings of the 4th ACM European Conference on Computer Systems*, Ser. EuroSys '09, Nuremberg, Germany: ACM, 2009, S. 205–218, ISBN: 978-1-60558-482-9. DOI: 10.1145/1519065.1519089.

[131] M. Trappmann, H. J. Hummell und W. Sodeur, *Strukturanalyse sozialer Netzwerke*. Springer, 2005.

[132] S. O. Krumke, H. Noltemeier und H.-C. Wirth, *Graphentheoretische Konzepte und Algorithmen*. Springer, 2009.

[133] L. C. Freeman, „A Set of Measures of Centrality Based on Betweenness“, *Sociometry*, Bd. 40, Nr. 1, S. 35–41, 1977, ISSN: 00380431.

[134] J. Diesner, T. L. Frantz und K. M. Carley, „Communication Networks from the Enron Email Corpus “It's Always About the People. Enron is no Different”“, *Computational & Mathematical Organization Theory*, Bd. 11, Nr. 3, S. 201–228, 2005, ISSN: 1572-9346. DOI: 10.1007/s10588-005-5377-0.

[135] J. Diesner und K. M. Carley, „Exploration of communication networks from the Enron email corpus“, in *In Proc. of Workshop on Link Analysis, Counterterrorism and Security, SIAM International Conference on Data Mining 2005*, 2005, S. 21–23.

[136] C. Perera, A. Zaslavsky, P. Christen und D. Georgakopoulos, „Context Aware Computing for The Internet of Things: A Survey“, *IEEE Communications Surveys Tutorials*, Bd. 16, Nr. 1, S. 414–454, Jan. 2014, ISSN: 1553-877X. DOI: 10.1109/SURV.2013.042313.00197.

[137] F. Garin und L. Schenato, „Networked Control Systems“, in, A. Bemporad, M. Heemels und M. Johansson, Hrsg. London: Springer London, 2010, Kap. A Survey on Distributed Estimation and Control Applications Using Linear Consensus Algorithms, S. 75–107, ISBN: 978-0-85729-033-5. DOI: 10.1007/978-0-85729-033-5_3.

[138] W. Dargie und C. Poellabauer, *Fundamentals of Wireless Sensor Networks: Theory and Practice*. John Wiley und Sons, Ltd., 2010, ISBN: 978-0-470-99765-9.

[139] Z. J. Haas, J. Y. Halpern und L. Li, „Gossip-based Ad Hoc Routing“, *IEEE/ACM Trans. Netw.*, Bd. 14, Nr. 3, S. 479–491, Juni 2006, ISSN: 1063-6692. DOI: 10.1109/TNET.2006.876186.

[140] M. Jelasity, A. Montresor und O. Babaoglu, „Gossip-based Aggregation in Large Dynamic Networks“, *ACM Trans. Comput. Syst.*, Bd. 23, Nr. 3, S. 219–252, Aug. 2005, ISSN: 0734-2071. DOI: 10.1145/1082469.1082470.

[141] M. Cao, D. Spielman und E. Yeh, „Accelerated Gossip Algorithms for Distributed Computation“, in *Proceedings of the 44th Annual Allerton Conference on Communication, Control, and Computation*. University of Groningen, Research Institute of Technology und Management, 2006, S. 952–959, Relation: http://www.rug.nl/ Rights: University of Groningen.

[142] S. Boyd, A. Ghosh, B. Prabhakar und D. Shah, „Randomized Gossip Algorithms“, *IEEE/ACM Trans. Netw.*, Bd. 14, Nr. SI, S. 2508–2530, Juni 2006, ISSN: 1063-6692. DOI: 10.1109/TIT.2006.874516.

[143] S. Boyd, A. Ghosh, B. Prabhakar und D. Shah, „Gossip algorithms: design, analysis and applications“, in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, Bd. 3, März 2005, 1653–1664 vol. 3. DOI: 10.1109/INFCOM.2005.1498447.

[144] M. Franceschelli, A. Giua und C. Seatzu, „Distributed Averaging in Sensor Networks Based on Broadcast Gossip Algorithms“, *Sensors Journal, IEEE*, Bd. 11, Nr. 3, S. 808–817, März 2011, ISSN: 1530-437X. DOI: 10.1109/JSEN.2010.2064295.

[145] T. Aysal, M. Yildiz, A. Sarwate und A. Scaglione, „Broadcast Gossip Algorithms for Consensus“, *Signal Processing, IEEE Transactions on*, Bd. 57, Nr. 7, S. 2748–2761, Juli 2009, ISSN: 1053-587X. DOI: 10.1109/TSP.2009.2016247.

[146] A. Speranzon, C. Fischione und K. H. Johansson, „Distributed and Collaborative Estimation over Wireless Sensor Networks“, in *Proceedings of the 45th IEEE Conference on Decision and Control*, Dez. 2006, S. 1025–1030. DOI: 10.1109/CDC.2006.377101.

[147] S. Kar und J. Moura, „Gossip and Distributed Kalman Filtering: Weak Consensus Under Weak Detectability“, *Signal Processing, IEEE Transactions on*, Bd. 59, Nr. 4, S. 1766–1784, Apr. 2011, ISSN: 1053-587X. DOI: 10.1109/TSP.2010.2100385.

[148] A. Chiuso, F. Fagnani, L. Schenato und S. Zampieri, „Gossip Algorithms for Simultaneous Distributed Estimation and Classification in Sensor Networks“, *Selected Topics in Signal Processing, IEEE Journal of*, Bd. 5, Nr. 4, S. 691–706, Aug. 2011, ISSN: 1932-4553. DOI: 10.1109/JSTSP.2011.2123079.

[149] R. Carli, A. Chiuso, L. Schenato und S. Zampieri, „Distributed Kalman filtering based on consensus strategies“, *Selected Areas in Communications, IEEE Journal on*, Bd. 26, Nr. 4, S. 622–633, Mai 2008, ISSN: 0733-8716. DOI: 10.1109/JSAC.2008.080505.

[150] Y. Lindell und B. Pinkas, „Secure multiparty computation for privacy-preserving data mining“, *Journal of Privacy and Confidentiality*, Bd. 1, Nr. 1, S. 5, 2009.

[151] M. Werner, „Indoor location-based services“, *Prerequisites and foundations. Cham: Springer*, 2014.

[152] C. C. Aggarwal und P. S. Yu, „A General Survey of Privacy-Preserving Data Mining Models and Algorithms“, in *Privacy-Preserving Data Mining*, 2008, S. 11–52. DOI: 10.1007/978-0-387-70992-5_2.

[153] P. Samarati, „Protecting respondents identities in microdata release“, *IEEE Transactions on Knowledge and Data Engineering*, Bd. 13, Nr. 6, S. 1010–1027, Nov. 2001, ISSN: 1041-4347. DOI: 10.1109/69.971193.

[154] A. Machanavajjhala, D. Kifer, J. Gehrke und M. Venkitasubramaniam, „L-diversity: Privacy Beyond K-anonymity“, *ACM Trans. Knowl. Discov. Data*, Bd. 1, Nr. 1, März 2007, ISSN: 1556-4681. DOI: 10.1145/1217299.1217302.

[155] N. Li, T. Li und S. Venkatasubramanian, „t-Closeness: Privacy Beyond k-Anonymity and l-Diversity“, in *2007 IEEE 23rd International Conference on Data Engineering*, Apr. 2007, S. 106–115. DOI: 10.1109/ICDE.2007.367856.

[156] C. Dwork, F. McSherry, K. Nissim und A. Smith, „Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings“, in, S. Halevi und T. Rabin, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, Kap. Calibrating Noise to Sensitivity in Private Data Analysis, S. 265–284, ISBN: 978-3-540-32732-5. DOI: 10.1007/11681878_14.

[157] C. Dwork, „Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II“, in, M. Bugliesi, B. Preneel, V. Sassone und I. Wegener, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, Kap. Differential Privacy, S. 1–12, ISBN: 978-3-540-35908-1. DOI: 10.1007/11787006_1.

[158] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov und M. Naor, „Advances in Cryptology - EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006. Proceedings“, in, S. Vaudenay, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, Kap. Our Data, Ourselves: Privacy Via Distributed Noise Generation, S. 486–503, ISBN: 978-3-540-34547-3. DOI: 10.1007/11761679_29.

[159] Y. Duan, J. Canny und J. Zhan, „P4P: Practical Large-scale Privacy-preserving Distributed Computation Robust Against Malicious Users“, in *Proceedings of the 19th USENIX Conference on Security*, Ser. USENIX Security'10, Washington, DC: USENIX Association, 2010, S. 14–14, ISBN: 888-7-6666-5555-4.

[160] J. Shi, R. Zhang, Y. Liu und Y. Zhang, „PriSense: Privacy-Preserving Data Aggregation in People-Centric Urban Sensing Systems“, in *INFOCOM, 2010 Proceedings IEEE*, März 2010, S. 1–9. DOI: 10.1109/INFCOM.2010.5462147.

[161] W. He, X. Liu, H. Nguyen, K. Nahrstedt und T. Abdelzaher, „PDA: Privacy-Preserving Data Aggregation in Wireless Sensor Networks“, in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, Mai 2007, S. 2045–2053. DOI: 10.1109/INFCOM.2007.237.

[162] Z. Huang, S. Mitra und G. Dullerud, „Differentially Private Iterative Synchronous Consensus“, in *Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society*, Ser. WPES '12, Raleigh, North Carolina, USA: ACM, 2012, S. 81–90, ISBN: 978-1-4503-1663-7. DOI: 10.1145/2381966.2381978.

[163] Y. Xiao, L. Xiong und C. Yuan, „Secure Data Management: 7th VLDB Workshop, SDM 2010, Singapore, September 17, 2010. Proceedings“, in, W. Jonker und M. Petković, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, Kap. Differentially Private Data Release through Multidimensional Partitioning, S. 150–168, ISBN: 978-3-642-15546-8. DOI: 10.1007/978-3-642-15546-8_11.

[164] X. Xiao, G. Bender, M. Hay und J. Gehrke, „iReduct: Differential Privacy with Reduced Relative Errors“, in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, Ser. SIGMOD '11, Athens, Greece: ACM, 2011, S. 229–240, ISBN: 978-1-4503-0661-4. DOI: 10.1145/1989323.1989348.

[165] L. Fan und L. Xiong, „An Adaptive Approach to Real-Time Aggregate Monitoring With Differential Privacy“, *IEEE Transactions on Knowledge and Data Engineering*, Bd. 26, Nr. 9, S. 2094–2106, Sep. 2014, ISSN: 1041-4347. DOI: 10.1109/TKDE.2013.96.

[166] T. F. Chan, G. H. Golub und R. J. Leveque, „Algorithms for Computing the Sample Variance: Analysis and Recommendations“, *The American Statistician*, Bd. 37, Nr. 3, S. 242–247, 1983. DOI: 10.1080/00031305.1983.10483115.

[167] B. Awerbuch, A. Bar-Noy, N. Linial und D. Peleg, „Compact Distributed Data Structures for Adaptive Routing“, in *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, Ser. STOC '89, Seattle, Washington, USA: ACM, 1989, S. 479–489, ISBN: 0-89791-307-8. DOI: 10.1145/73007.73053.

[168] M. Werner, „Privacy-protected communication for location-based services“, *Security and Communication Networks*, Bd. 9, Nr. 2, S. 130–138, 2016, ISSN: 1939-0122. DOI: 10.1002/sec.330.

[169] D. Agrawal und C. C. Aggarwal, „On the Design and Quantification of Privacy Preserving Data Mining Algorithms“, in *Proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Ser. PODS '01, Santa Barbara, California, USA: ACM, 2001, S. 247–255, ISBN: 1-58113-361-8. DOI: 10.1145/375551.375602.

[170] R. Wolff, „Local Thresholding in General Network Graphs“, *Knowledge and Data Engineering, IEEE Transactions on*, Bd. 26, Nr. 4, S. 916–928, Apr. 2014, ISSN: 1041-4347. DOI: 10.1109/TKDE.2013.59.

[171] A. Acquisti und R. Gross, „Privacy Enhancing Technologies: 6th International Workshop, PET 2006, Cambridge, UK, June 28-30, 2006, Revised Selected Papers“, in, G. Danezis und P. Golle, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, Kap. Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook, S. 36–58, ISBN: 978-3-540-68793-1. DOI: 10.1007/11957454_3.

[172] R. J. Höflich, „Der Mensch und seine Medien: Mediatisierte interpersonale Kommunikation. Eine Einführung“, in. Wiesbaden: Springer Fachmedien Wiesbaden, 2016, Kap. Was bedeutet es, wenn Menschen ein Medium verwenden?, S. 39–65, ISBN: 978-3-531-18683-2. DOI: 10.1007/978-3-531-18683-2_3.

[173] E. Lippmann, *Identität im Zeitalter des Chamäleons: Flexibel sein und Farbe bekennen*. Vandenhoeck & Ruprecht, 2013.

[174] D. Balfanz, „Identität in der Virtualität - Einblicke in neue Arbeitswelten und Industrie 4.0“, in. Mössingen-Talheim: Talheimer Verlag, 2014, Kap. Die digital persona, ISBN: 978-3-89376-155-5.

[175] A. Beimel, K. Nissim und E. Omri, „Advances in Cryptology – CRYPTO 2008: 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings“, in, D. Wagner, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, Kap. Distributed Private Data Analysis: Simultaneously Solving How and What, S. 451–468, ISBN: 978-3-540-85174-5. DOI: 10.1007/978-3-540-85174-5_25.

[176] A. B. Zaslavsky, C. Perera und D. Georgakopoulos, „Sensing as a Service and Big Data“, *Proceedings of the International Conference on Advances in Cloud Computing (ACC)*, Jan. 2013, ISSN: 9788173717789.

[177] Y. Duan und J. Canny, „How to deal with malicious users in privacy-preserving distributed data mining“, *Statistical Analysis and Data Mining*, Bd. 2, Nr. 1, S. 18–33, 2009, ISSN: 1932-1872. DOI: 10.1002/sam.10029.

[178] R. Agrawal und R. Srikant, „Privacy-preserving Data Mining“, in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, Ser. SIGMOD '00, Dallas, Texas, USA: ACM, 2000, S. 439–450, ISBN: 1-58113-217-4. DOI: 10.1145/342009.335438.

[179] W. Kowalczyk, M. Jelasity und A. Eiben, „Towards Data Mining in Large and Fully Distributed Peer-to-Peer Overlay Networks“, in *BNAIC '03. 15th Belgian-Dutch Conference on Artificial Intelligence, Proceedings of the*, 2003, S. 203–210.

[180] E. S. S. Chatterjee, „Towards Consensus: Some Convergence Theorems on Repeated Averaging“, *Journal of Applied Probability*, Bd. 14, Nr. 1, S. 89–97, 1977, ISSN: 00219002.

[181] S. Datta, K. Bhaduri, C. Giannella, R. Wolff und H. Kargupta, „Distributed Data Mining in Peer-to-Peer Networks“, *Internet Computing, IEEE*, Bd. 10, Nr. 4, S. 18–26, Juli 2006, ISSN: 1089-7801. DOI: 10.1109/MIC.2006.74.

[182] S. Kar und J. Moura, „Sensor Networks With Random Links: Topology Design for Distributed Consensus“, *Signal Processing, IEEE Transactions on*, Bd. 56, Nr. 7, S. 3315–3326, Juli 2008, ISSN: 1053-587X. DOI: 10.1109/TSP.2008.920143.

[183] R. Wolff, K. Bhaduri und H. Kargupta, „Local L2-Thresholding Based Data Mining in Peer-to-Peer Systems“, in *Proceedings of the 2006 SIAM International Conference on Data Mining*, Kap. 38, S. 430–441. DOI: 10.1137/1.9781611972764.38.

[184] S. Ruohomaa, L. Kutvonen und E. Koutrouli, „Reputation Management Survey“, in *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on*, Apr. 2007, S. 103–111. DOI: 10.1109/ARES.2007.123.

[185] K. Hoffman, D. Zage und C. Nita-Rotaru, „A Survey of Attack and Defense Techniques for Reputation Systems“, *ACM Comput. Surv.*, Bd. 42, Nr. 1, 1:1–1:31, Dez. 2009, ISSN: 0360-0300. DOI: 10.1145/1592451.1592452.

[186] S. D. Kamvar, M. T. Schlosser und H. Garcia-Molina, „The Eigentrust Algorithm for Reputation Management in P2P Networks“, in *Proceedings of the 12th International Conference on World Wide Web*, Ser. WWW '03, Budapest, Hungary: ACM, 2003, S. 640–651, ISBN: 1-58113-680-3. DOI: 10.1145/775152.775242.

[187] P. Resnick, K. Kuwabara, R. Zeckhauser und E. Friedman, „Reputation Systems“, *Commun. ACM*, Bd. 43, Nr. 12, S. 45–48, Dez. 2000, ISSN: 0001-0782. DOI: 10.1145/355112.355122.

[188] L. Xiong und L. Liu, „PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities“, *Knowledge and Data Engineering, IEEE Transactions on*, Bd. 16, Nr. 7, S. 843–857, Juli 2004, ISSN: 1041-4347. DOI: 10.1109/TKDE.2004.1318566.

[189] Y. Kim und I. Yeom, „Performance analysis of in-network caching for content-centric networking“, *Computer Networks*, Bd. 57, Nr. 13, S. 2465–2482, 2013, ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/j.comnet.2012.11.026.

[190] G. Zhang, Y. Li und T. Lin, „Caching in information centric networking: A survey“, *Computer Networks*, Bd. 57, Nr. 16, S. 3128–3141, 2013, Information Centric Networking, ISSN: 1389-1286. DOI: http://dx.doi.org/10.1016/j.comnet.2013.07.007.

[191] S. M. Bellovin und W. R. Cheswick, „Privacy-enhanced searches using encrypted bloom filters“, Techn. Ber., 2007.

[192] R. Nojima und Y. Kadobayashi, „Cryptographically Secure Bloom-Filters“, *Transactions on Data Privacy*, Bd. 2, Nr. 2, S. 131–139, Aug. 2009, ISSN: 1888-5063.

[193] B. Chazelle, J. Kilian, R. Rubinfeld und A. Tal, „The Bloomier Filter: An Efficient Data Structure for Static Support Lookup Tables“, in *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, Ser. SODA '04, New Orleans, Louisiana: Society for Industrial und Applied Mathematics, 2004, S. 30–39, ISBN: 0-89871-558-X.

[194] Y. Katz und J. Golbec, „Using social network-based trust for default reasoning on the web“, *Journal of Web Semantics*, 2007.

[195] T. Hogg, „Security Challenges for Reputation Mechanisms Using Online Social Networks“, in *Proceedings of the 2Nd ACM Workshop on Security and Artificial Intelligence*, Ser. AISec '09, Chicago, Illinois, USA: ACM, 2009, S. 31–34, ISBN: 978-1-60558-781-3. DOI: 10.1145/1654988.1654998.

[196] R. Cáceres, L. Cox, H. Lim, A. Shakimov und A. Varshavsky, „Virtual Individual Servers As Privacy-preserving Proxies for Mobile Devices“, in *Proceedings of the 1st ACM Workshop on Networking, Systems, and Applications for Mobile Handhelds*, Ser. MobiHeld '09, Barcelona, Spain: ACM, 2009, S. 37–42, ISBN: 978-1-60558-444-7. DOI: 10.1145/1592606.1592616.

[197] J. Liu und V. Issarny, „Trust Management: Second International Conference, iTrust 2004, Oxford, UK, March 29 - April 1, 2004. Proceedings“, in, C. Jensen, S. Poslad und T. Dimitrakos, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, Kap. Enhanced Reputation Mechanism for Mobile Ad Hoc Networks, S. 48–62, ISBN: 978-3-540-24747-0. DOI: 10.1007/978-3-540-24747-0_5.

[198] L. Lamport, „The Part-time Parliament“, *ACM Trans. Comput. Syst.*, Bd. 16, Nr. 2, S. 133–169, Mai 1998, ISSN: 0734-2071. DOI: 10.1145/279227.279229.

[199] E. Jaho und I. Stavrakakis, „Joint interest- and locality-aware content dissemination in social networks“, in *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, Feb. 2009, S. 173–180. DOI: 10.1109/WONS.2009.4801864.

[200] K. Govindan und P. Mohapatra, „Trust Computations and Trust Dynamics in Mobile Adhoc Networks: A Survey“, *Communications Surveys Tutorials, IEEE*, Bd. 14, Nr. 2, S. 279–298, Feb. 2012, ISSN: 1553-877X. DOI: 10.1109/SURV.2011.042711.00083.

[201] K. P. N. Puttaswamy und B. Y. Zhao, „Preserving Privacy in Location-based Mobile Social Applications“, in *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*, Ser. HotMobile '10, Annapolis, Maryland: ACM, 2010, S. 1–6, ISBN: 978-1-4503-0005-6. DOI: 10.1145/1734583.1734585.

[202] E. Yoneki, P. Hui, S. Chan und J. Crowcroft, „A Socio-aware Overlay for Publish/Subscribe Communication in Delay Tolerant Networks“, in *Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, Ser. MSWiM '07, Chania, Crete Island, Greece: ACM, 2007, S. 225–234, ISBN: 978-1-59593-851-0. DOI: 10.1145/1298126.1298166.

[203] X. Zhang, H. Liu und A. Abraham, „A Novel Process Network Model for Interacting Context-Aware Web Services“, *Services Computing, IEEE Transactions on*, Bd. 6, Nr. 3, S. 344–357, Juli 2013, ISSN: 1939-1374. DOI: 10.1109/TSC.2012.6.

[204] J. Manweiler, R. Scudellari und L. P. Cox, „SMILE: Encounter-based Trust for Mobile Social Services“, in *Proceedings of the 16th ACM Conference on Computer and Communications Security*, Ser. CCS '09, Chicago, Illinois, USA: ACM, 2009, S. 246–255, ISBN: 978-1-60558-894-0. DOI: 10.1145/1653662.1653692.

[205] C.-H. O. Chen, C.-W. Chen, C. Kuo, Y.-H. Lai, J. M. McCune, A. Studer, A. Perrig, B.-Y. Yang und T.-C. Wu, „GAnGS: Gather, Authenticate 'N Group Securely“, in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, Ser. MobiCom '08,

San Francisco, California, USA: ACM, 2008, S. 92–103, ISBN: 978-1-60558-096-8. DOI: 10.1145/1409944.1409957.

[206] I. Fez, M. Gil, J. Fons, J. C. Guerri und V. Pelechano, „A personalized system for scalable distribution of multimedia content in multicast wireless networks“, *Multimedia Tools and Applications*, Bd. 74, Nr. 21, S. 9595–9621, 2014, ISSN: 1573-7721. DOI: 10.1007/s11042-014-2139-3.

[207] M. Lund, B. Solhaug und K. Stlen, „Evolution in Relation to Risk and Trust Management“, *Computer*, Bd. 43, Nr. 5, S. 49–55, Mai 2010, ISSN: 0018-9162. DOI: 10.1109/MC.2010.134.

[208] E. I. Tatli, „Security in Context-aware Mobile Business Applications“, ARRAY(0x7fc146a26400), Diss., März 2008.

[209] M. Tschersich, C. Kahl, S. Heim, S. Crane, K. Böttcher, I. Krontiris und K. Rannenberg, „Towards privacy-enhanced mobile communities – Architecture, concepts and user trials“, *Journal of Systems and Software*, Bd. 84, Nr. 11, S. 1947–1960, 2011, Mobile Applications: Status and Trends, ISSN: 0164-1212. DOI: http://dx.doi.org/10.1016/j.jss.2011.06.048.

[210] J.-H. Cho, A. Swami und I.-R. Chen, „A Survey on Trust Management for Mobile Ad Hoc Networks“, *Communications Surveys Tutorials, IEEE*, Bd. 13, Nr. 4, S. 562–583, Apr. 2011, ISSN: 1553-877X. DOI: 10.1109/SURV.2011.092110.00088.

[211] D. Balakrishnan, M. Barachi, A. Karmouch und R. Glitho, „Mobility Aware Technologies and Applications: Second International Workshop, MATA 2005, Montreal, Canada, October 17-19, 2005. Proceedings“, in, T. Magedanz, A. Karmouch, S. Pierre und I. Venieris, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, Kap. Challenges in Modeling and Disseminating Context Information in Ambient Networks, S. 32–42, ISBN: 978-3-540-32031-9. DOI: 10.1007/11569510_4.

[212] X. Hu, Y. Ding, N. Paspallis, P. Bratskas, G. A. Papadopoulos, Y. Vanrompay, M. K. Pinheiro und Y. Berbers, „Information Systems Development: Towards a Service Provision Society“, in, A. G. Papadopoulos, W. Wojtkowski, G. Wojtkowski, S. Wrycza und J. Zupancic, Hrsg. Boston, MA: Springer US, 2010, Kap. A Hybrid Peer-to-Peer Solution for Context Distribution in Mobile and Ubiquitous Environments, S. 501–510, ISBN: 978-0-387-84810-5. DOI: 10.1007/b137171_52.

[213] S. Kiani, M. Knappmeyery, N. Baker und B. Moltchanov, „A Federated Broker Architecture for Large Scale Context Dissemination“, in *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, Juni 2010, S. 2964–2969. DOI: 10.1109/CIT.2010.495.

[214] J. Simoes und T. Magedanz, „Contextualized User-Centric Multimedia Delivery System for Next Generation Networks“, *Telecommunication Systems*, Bd. 48, Nr. 3, S. 301–316, 2010, ISSN: 1572-9451. DOI: 10.1007/s11235-010-9345-8.

[215] K. Paridel, T. Mantadelis, A.-U.-H. Yasar, D. Preuveneers, G. Janssens, Y. Vanrompay und Y. Berbers, „Analyzing the efficiency of context-based grouping on collaboration in VANETs with large-scale simulation“, *Journal of Ambient Intelligence and Humanized Computing*, Bd. 5, Nr. 4, S. 475–490, 2012, ISSN: 1868-5145. DOI: 10.1007/s12652-012-0115-1.

[216] A. Yasar, Y. Vanrompay, D. Preuveneers und Y. Berbers, „Optimizing information dissemination in large scale mobile peer-to-peer networks using context-based grouping“, in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, Sep. 2010, S. 1065–1071. DOI: 10.1109/ITSC.2010.5625104.

[217] L. Jin, Y. Chen, T. Wang, P. Hui und A. Vasilakos, „Understanding user behavior in online social networks: a survey“, *Communications Magazine, IEEE*, Bd. 51, Nr. 9, S. 144–150, Sep. 2013, ISSN: 0163-6804. DOI: 10.1109/MCOM.2013.6588663.

[218] N. Agoulmine, S. Balasubramaniam, D. Botvich, J. Strassner, E. Lehtihet und W. Donnelly, „Challenges for autonomic network management“, 2006.

[219] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt und F. Zambonelli, „A Survey of Autonomic Communications“, *ACM Trans. Auton. Adapt. Syst.*, Bd. 1, Nr. 2, S. 223–259, Dez. 2006, ISSN: 1556-4665. DOI: 10.1145/1186778.1186782.

[220] N. Baker, M. Zafar, B. Moltchanov und M. Knappmeyer, „Context-Aware Systems and Implications for Future Internet.“, in *Future Internet Assembly*, 2009, S. 335–344.

[221] J. Bao, Y. Zheng, D. Wilkie und M. Mokbel, „Recommendations in location-based social networks: a survey“, *GeoInformatica*, Bd. 19, Nr. 3, S. 525–565, 2015, ISSN: 1573-7624. DOI: 10.1007/s10707-014-0220-8.

[222] W. Sherchan, S. Nepal und C. Paris, „A Survey of Trust in Social Networks“, *ACM Comput. Surv.*, Bd. 45, Nr. 4, 47:1–47:33, Aug. 2013, ISSN: 0360-0300. DOI: 10.1145/2501654.2501661.

[223] P. Makris, D. Skoutas und C. Skianis, „A Survey on Context-Aware Mobile and Wireless Networking: On Networking and Computing Environments' Integration“, *Communications Surveys Tutorials, IEEE*, Bd. 15, Nr. 1, S. 362–386, Jan. 2013, ISSN: 1553-877X. DOI: 10.1109/SURV.2012.040912.00180.

[224] N. Kourtellis, J. Finnis, P. Anderson, J. Blackburn, C. Borcea und A. Iamnitchi, „Prometheus: User-controlled P2P Social Data Management for Socially-aware Applications“, in *Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware*, Ser. Middleware '10, Bangalore, India: Springer-Verlag, 2010, S. 212–231, ISBN: 978-3-642-16954-0.

[225] A. Abidi und S. Gammar, „Towards New Caching Strategy for Information-centric Networking Based on Data Proximity Control“, in *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*, Okt. 2015, S. 540–547. DOI: 10.1109/CIT/IUCC/DASC/PICOM.2015.77.

## Internetquellen

[226] LinkedIn. (2016), Adresse: http://www.linkedin.com (besucht am 19.09.2016).

[227] Facebook. (2016), Adresse: http://www.facebook.com (besucht am 19.09.2016).

[228] Twitter. (2016), Adresse: http://www.twitter.com (besucht am 19.09.2016).

[229] Yelp. (2016), Adresse: http://www.yelp.com (besucht am 19.09.2016).

[230] A. Cassandra. (2016). Use-Cases der Datenbank Apache Cassandra, Adresse: http://www.planetcassandra.org/apache-cassandra-use-cases/ (besucht am 19.09.2016).

[231] G. Niemeyer. (2003). Öffentliche Ankündigung von GeoHashes, Adresse: http://forums.groundspeak.com/GC/index.php?showtopic=186412 (besucht am 19.09.2016).

[232] YouTube. (2016), Adresse: http://www.youtube.com (besucht am 19.09.2016).

[233] BitTorrent. (2016), Adresse: http://www.bittorrent.com (besucht am 19.09.2016).

[234] M. Honan. (2012). How Apple and Amazon Security Flaws Led to My Epic Hacking, Adresse: http://www.wired.com/2012/08/apple-amazon-mat-honan-hacking/ (besucht am 16.06.2016).

[235] BitCoin. (2016), Adresse: http://bitcoin.org (besucht am 19.09.2016).

[236] Amazon. (2016), Adresse: http://www.amazon.de (besucht am 19.09.2016).

# Eigene Publikationen


[237] S. Feld, M. Schönfeld und M. Werner, „Traversing Bitcoin's P2P network: Insights into the structure of a decentralized currency“, *International Journal of Computational Science and Engineering, Special Issue on Security and Trust Issues in Peer-to-Peer Networks*, Bd. 13, Nr. 2, S. 122–131, 2016.

[238] S. Feld, M. Werner, M. Schönfeld und S. Hasler, „Archetypes of Alternative Routes in Buildings“, in *Proceedings of the 6th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2015.

[239] M. Schönfeld, „HbbRadio oder der personalisierte Rundfunk“, German, in *Marktplätze im Umbruch*, Ser. Xpert.press, C. Linnhoff-Popien, M. Zaddach und A. Grahl, Hrsg., Springer Berlin Heidelberg, 2015, S. 203–211, ISBN: 978-3-662-43781-0. DOI: 10.1007/978-3-662-43782-7_23.

[240] M. Schönfeld, M. Werner, C. Linnhoff-Popien und A. Erk, „Towards a Privacy-Preserving Hybrid Radio Network: Design and Open Challenges“, in *I4CS 2015, Proceedings of the 15th International Conference on Innovations for Community Services*, 2015.

[241] M. Werner, F. Dorfmeister und M. Schönfeld, „AMBIENCE: A Context-Centric Online Social Network“, in *WPNC 2015, 12th Workshop on Positioning, Navigation and Communications*, 2015.

[242] M. Werner und M. Schönfeld, „The Gaussian Bloom Filter“, in *DASFAA 2015, Proceedings of the 20th International Conference on Database Systems for Advanced Applications*, 2015.

[243] S. Feld, M. Schönfeld und M. Werner, „Analyzing the deployment of Bitcoin's P2P network under an AS-level perspective“, in *SPINS 2014, International Workshop on Secure Peer-to-Peer Intelligent Networks & Systems*, 2014, S. 1121–1126.

[244] M. Schönfeld und M. Werner, „Distributed Privacy-Preserving Mean Estimation“, in *PRISMS 2014, Proceedings of the 2nd International Conference on Privacy and Security in Mobile Systems*, 2014.

[245] M. Schönfeld und M. Werner, „Node Wake-Up via OVSF-Coded Bloom Filters in Wireless Sensor Networks“, in *ADHOCNETS 2013, Proceedings of the 5th International Conference on Ad Hoc Networks*, 2013.

[246] F. Dorfmeister, M. Maier, M. Schönfeld und S. A. W. Verclas, „SmartBEEs: Enabling Smart Business Environments Based on Location Information and Sensor Networks“, in *9. GI/ITG KuVS Fachgespräch Ortsbezogene Anwendungen und Dienste*, 2012, S. 43–56.

[247] M. Kessel, M. Maier, M. Schönfeld und F. Dorfmeister, „Testing Sensor Fusion Algorithms in Indoor Positioning Scenarios“, in *9. GI/ITG KuVS Fachgespräch Ortsbezogene Anwendungen und Dienste*, 2012, S. 133–142.

[248] M. Maier, F. Dorfmeister, M. Schönfeld und M. Kessel, „A Tool for Visualizing and Editing Multiple Parallel Tracks of Time Series Data from Sensor Logs“, in *9. GI/ITG KuVS Fachgespräch Ortsbezogene Anwendungen und Dienste*, 2012, S. 99–108.

[249] M. Schönfeld, M. Werner und F. Dorfmeister, „Location-based Access Control Providing One-Time Passwords Through 2D Barcodes“, in *9. GI/ITG KuVS Fachgespräch Ortsbezogene Anwendungen und Dienste*, 2012, S. 165–168.

[250] M. Werner und M. Schönfeld, „DualCodes: Backward-Compatible Multi-Layer 2D-Barcodes“, in *MOBIQUITOUS 2012, LNICST 120*, 2012, S. 25–36.