Werkzeugunterstützung für komplexe Webrecherchen

Von der Fakultät für Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Fabian Kaiser

aus Stuttgart-Bad Cannstatt

Hauptberichter: Prof. Dr.-Ing. habil. Bernhard Mitschang

Mitberichter: Prof. Hinrich Schütze, Ph. D.

Tag der mündlichen Prüfung: 11.01.2013

Institut für Parallele und Verteilte Systeme (IPVS)
der Universität Stuttgart

2013

Inhaltsverzeichnis

Vorwort

Diese Dissertation entstand im Rahmen meiner Tätigkeit als wissenschaftlicher Mitarbeiter der Abteilung *Anwendersoftware* des *Instituts für Parallele und Verteilte Systeme (IPVS)* an der *Universität Stuttgart*. Maßgeblich motiviert und inspiriert wurde sie durch das BMBF-Forschungsprojekt *nova-net* und die damit einhergehende interdisziplinäre Kooperation mit Partnern aus Forschung und Industrie zum Thema „Unterstützung von Innovationsprozessen durch Internettechnologien“.

Ohne das Mitwirken unzähliger Personen wäre diese Arbeit nicht in der vorliegenden Weise gelungen. Besonders bedanken möchte ich mich bei Herrn Prof. Bernhard Mitschang. Er hat die Arbeit stets wohlwollend begleitet, Anregungen geliefert und mir bei der Umsetzung meiner Ideen weitgehende Freiheiten gelassen. Bei Herrn Prof. Hinrich Schütze möchte ich mich für die fruchtbaren Diskussionen sowie für die Übernahme des Mitberichts bedanken.

Den Partnern im Rahmen des Projekts *nova-net*, insbesondere Daniel Heubach, Claus Lang-Kötz, Severin Beucker, Sven Schimpf vom Fraunhofer IAO, danke ich für die vielen Anregungen, die Bereitschaft ihr Wissen über betriebliche Abläufe und Methodiken einzubringen sowie für ihr Feedback zum Praxiseinsatz der im Rahmen dieser Arbeit entstandenen Software. Auch die Industrie hat in Interviews und Prototypen-Evaluationen viele wertvolle Anregungen und Feedback geliefert. An dieser Stelle sei insbesondere den Firmen Intracom, Invia und Ask Askubal gedankt.

Bei meinen Kolleginnen und Kollegen vom IPVS und IAAS möchte ich mich für die erfolgreiche Zusammenarbeit bedanken. Insbesondere danke ich Holger Schwarz und Mihály Jakob für die zahlreichen Diskussionen und Ratschläge, von denen diese Arbeit merklich profitiert hat. Die unterschiedlichen Schwerpunkte unserer jeweiligen Arbeiten wurden durch das gemeinsame Forschungsprojekt *nova-net* integriert und haben sich gegenseitig befruchtet. Für seine kritische Durchsicht dieser Arbeit und die daraus resultierenden Anmerkungen möchte ich Thorsten Scheibler danken.

Auch viele Studenten haben zum Gelingen dieser Dissertation beigetragen und wertvolle Eingaben im Rahmen von Studien- und Diplomarbeiten, Fachstudien und Studienprojekten geliefert. Stellvertretend für sie seien an dieser Stelle Ralph Mietzner, Florian Laws, Sebastian Wiedersheim und Benjamin Geißelmeier genannt. Natürlich gilt auch ihnen mein Dank.

Ohne die Unterstützung meiner Familie wäre diese Arbeit nicht möglich gewesen. Mein ganz besonderer Dank gilt daher meiner Frau Sandra für ihre oft strapazierte Geduld und ihr stetes Drängen, neben allen anderen wichtigen und vermeintlich wichtigen Themen und Aktivitäten die Fertigstellung dieser Arbeit nicht aus den Augen zu verlieren.

Stuttgart, im September 2011
Fabian Kaiser

Zusammenfassung

Wissen bzw. der Wissensvorsprung gegenüber Wettbewerbern stellt im unternehmerischen Umfeld ein zunehmend gefragtes Gut dar. Mehr und mehr Informationsquellen werden erschlossen, um diesen Vorteil der Informiertheit und die damit einhergehende Sicherheit zu erreichen und auszubauen. In den letzten Jahren bspw. wurde unter dem Stichwort „Data Mining“ hoher Aufwand in die Erschließung von unternehmensinternen Informationsquellen wie ERP- oder PPS-Systemen investiert, um die in den Datenbanken dieser Systeme verborgenen Informationen zu verarbeiten und entsprechendes Wissen daraus abzuleiten. Durch immer schnellere Innovationszyklen und Produktfolgen wächst jedoch der Druck, sich neuen Aufgaben und Herausforderungen zuzuwenden und sich ggf. auf technologisches oder methodisches Neuland zu begeben. Neue Wissensquellen müssen hierzu erschlossen werden, da innerhalb der jeweiligen Unternehmung, insbesondere bei kleineren Firmen, das benötigte Wissen oftmals nicht vorhanden ist.

Mit dem WWW hat sich eine Informationsbasis etabliert, die nahezu unerschöpflich ist. Informationen und Daten zu beliebigen Themen werden dort vorgehalten und harren ihrer Auswertung. Genau hierin besteht jedoch die Schwierigkeit. Existierende Werkzeuge, insbesondere herkömmliche Suchmaschinen, bieten dem Benutzer bei der Suche im Rahmen verhältnismäßig einfacher Fragestellungen, wie bspw. der Suche nach bekannten Fakten, eine weitgehende Unterstützung. Werden die Fragestellungen jedoch komplexer in dem Sinne, dass das Ergebnis der Suche kein einfaches Faktum sein kann,

sondern vielmehr das Resultat eines aufwändigen Analyseprozesses, so existiert hierfür kaum Werkzeugunterstützung. Die Durchführung einer solchen komplexen Recherche ist somit für den Benutzer zeitaufwändig und wenig effektiv, da ein hoher manueller Aufwand für die weiter gehende Analyse von ersten Suchresultaten erbracht werden muss.

Im Rahmen dieser Arbeit werden nun Konzepte vorgeschlagen, wie solche komplexen Recherchen besser durch Werkzeuge unterstützt werden können. Es wird dazu eine Software-Plattform als Basis für die Entwicklung von Suchwerkzeugen zum Einsatz in komplexen Webrecherchen diskutiert. Aufbauend auf dieser Plattform wurden verschiedene Suchwerkzeuge konzipiert, die im Anschluss präsentiert werden: eine Entwicklungs-Komponente zur webpräsenzspezifischen Analyse von Dokumenten, eine Komponente bzw. ein Maß zur Berechnung der inhaltlichen Ähnlichkeit von Texten, um Suchen schnell auf thematisch relevante Bereiche einzugrenzen, sowie eine Technik zur Informationsextraktion aus diesen Texten.

Diese Konzepte bilden die Grundlage für eine weiter gehende Unterstützung des Anwenders bei komplexen Webrecherchen, als herkömmliche Suchmaschinen sie zu leisten vermögen. Zudem werden Entwickler von Suchwerkzeugen in die Lage versetzt, schnell und effizient auf ein Suchproblem zugeschnittene Werkzeuge oder Werkzeugkomponenten zu entwickeln, und diese zu integrieren.

Die vorgeschlagenen Konzepte werden in Bezug auf ihre Leistungsfähigkeit evaluiert. Anhand eines übergreifenden Anwendungsbeispiels wird dann deren Zusammenspiel erläutert und der Mehrwert dieser Lösungen gegenüber herkömmlichen Werkzeugen dargestellt.

Abstract

Knowledge and the knowledge edge over competitors is gaining more and more importance in the business environment. New sources of information are developed in order to get more information and a more reliable basis for strategic decisions than any competitor and thus to gain a competitive edge. In recent years, for example, high effort was spent in developing company internal information sources such as ERP or PPS systems by means of „Data Mining“, in order to process the information that is hidden in the databases of these systems and to derive new valuable knowledge from them. However, increasingly rapid cycles of innovation lead to an increased pressure to face new challenges and to focus on technological and methodological new territory. Therefore, new sources of knowledge must be developed, as the necessary knowledge often does not exist within the company, especially within smaller companies.

The WWW forms an information basis that is virtually inexhaustible. It holds information and data on almost any topic and these information are awaiting their anaylsis and utilization. Yet, this is where difficulties arise. Existing tools such as conventional search engines provide valuable help to the user, especially for simple information needs such as searching for known issues. However, there is little tool supprort if complexity of such an information need increases in a way that it cannot be satisfied by answers consisting of simple facts but if it requires a complex analysis of many resources. Thus, manually processing such complex searches is highly time consuming and little effective, as lots of data have to be read and evaluated by the user.

The concepts proposed in this dissertation aim to enhance the tool support for complex web searches: A search framework builds the basis for developing new search tools and for integrating them. On top of this framework, several tools have been designed and implemented: a subframework for the website-specific analysis of resources, an approach to measure the semantic relatedness of two texts in order to quickly focus the search on areas of the web that are relevant for the current search, and finally a concept for extracting information from these resources.

These concepts form the basis for better supporting the user in complex search problems. Furthermore, developers of search tools are supported in efficiently creating tools and components that are specific to a given search problem and in integrating them.

The approaches proposed here are each evaluated in terms of their performance. A comprehensive business case then illustrates their interaction and points out the value of these solutions compared to conventional tools.

KAPITEL 1

EINLEITUNG

Im Zeitalter der Informationsgesellschaft sind Informationen und das daraus ableitbare Wissen ein entscheidender Faktor für den wirtschaftlichen Erfolg eines Unternehmens. Informationen und Wissen bilden die Grundlage für unternehmerische Entscheidungen und damit für jegliche Innovation und wirtschaftlichen Fortschritt. Sie bilden aktuelle Entwicklungen in Worte und Zahlen ab und stellen somit Vergleichbarkeit her bzw. machen wirtschaftliche und soziale Rahmenbedingungen messbar. Die Kenntnis dieser Rahmenbedingungen stellt die Grundlage für innovative Entwicklungen dar, aus denen in der Folge wirtschaftliche Erfolge generiert werden [FKH09]. Der Zugang zu den Quellen dieser Informations-Ressourcen bzw. die Möglichkeit, diese zu nutzen, ist daher ein mit Nachdruck verfolgtes Ziel modernen Unternehmensmanagements [Spr06].

Der Zugriff auf aktuelle Informationen spielt dabei in vielen Bereichen des unternehmerischen Handelns eine entscheidende Rolle. Er unterscheidet sich in Art, Umfang und Quelle der benötigten Informationen.

Abbildung 1.1 zeigt unterschiedliche Wissensquellen bzw. Arten von Wissen

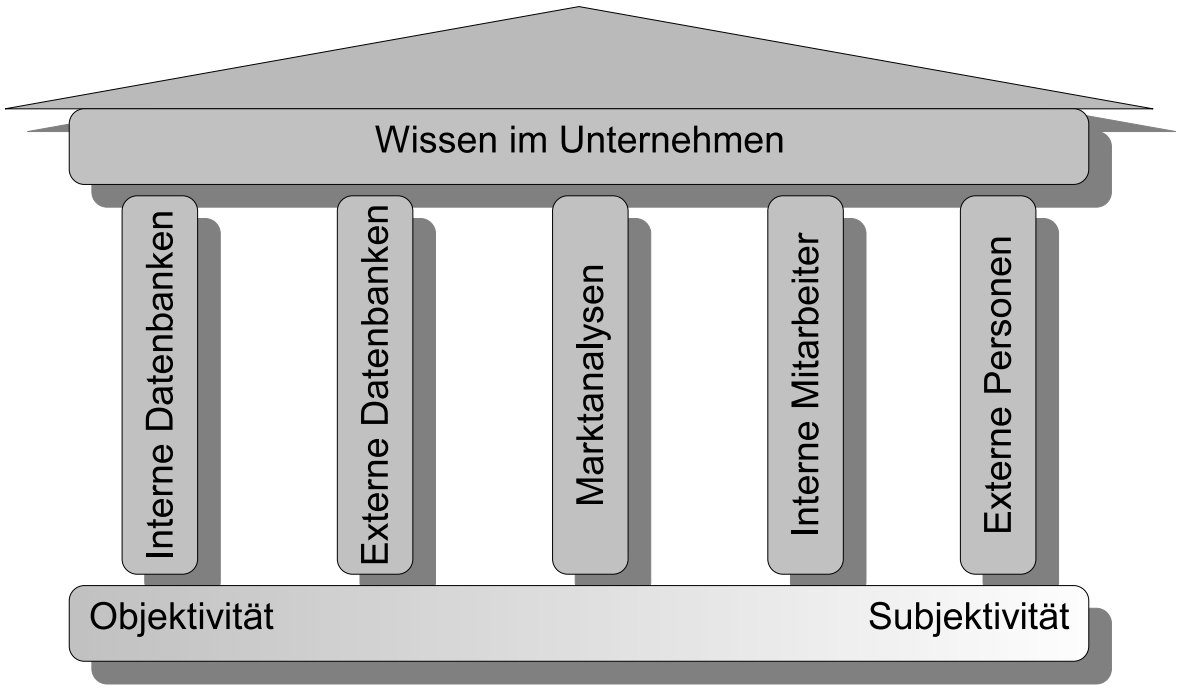

Abbildung 1.1: Quellen von Wissen im Unternehmen

im Unternehmen. Sie lassen sich in zwei Bereiche gliedern: Zum einen der Bereich der exakt quantifizierbaren objektiven Informationen wie Produktions- und Verkaufsdaten. Informationen in diesen Bereichen können aus anfallenden Prozessdaten wie bspw. „gefertigte Produkteinheiten pro Zeit“ oder „verkaufte Einheiten pro Monat“ abgeleitet werden und stellen ein wichtiges Instrument für die Erfassung der aktuellen Leistungsfähigkeit des Unternehmens dar. Dem gegenüber steht der Bereich, in dem Informationen nicht exakt quantifizierbar und vielmehr subjektiver Natur sind. Dort sind Einschätzungen und Bewertungen gefragt, die entweder von menschlichen Akteuren oder aber durch automatisierte Verfahren auf Basis von Heuristiken geliefert werden. Dies trifft bspw. auf Bereiche wie Kunden- und Verbraucherfeedback zu, wo subjektive, oftmals natürlichsprachig formulierte und daher schwer automatisiert verarbeitbare Markt- und Meinungsanalysen, Problembeschreibungen u. ä. die Informationsbasis bilden.

Um den aus dieser Unschärfe resultierenden Unsicherheiten zu begegnen, werden vielfach unternehmensinterne wie -externe Experten hinzugezogen, die ihr Wissen und ihre Erfahrung einbringen um die erhobenen Daten einzuord-

nen, zu berwerten und somit für eine gesichertere Entscheidungsgrundlage sorgen. Im Rahmen des Forschungsprojekts *nova-net – Innovation in der Internetökonomie* wurde hierzu die Methode des Delphigestützten Szenariomanagements entwickelt [JKS+07] sowie eine Softwarekomponente implementiert [JKS05][JSKM06], die diese Methode umsetzt und die Zusammenarbeit verteilter Experten sowie die zentrale Planung und Auswertung solcher Datenerhebungen unterstützt.

Bei der Bewertung und Analyse von unternehmensintern bereits verfügbaren Daten, bspw. aus CRM-, ERP- und PPS-Systemen, setzt die Industrie mehr und mehr auf Techniken wie das Data Mining. Damit wird versucht, Muster und Abhängigkeiten in strukturierten Daten und Prozessen zu entdecken und daraus entsprechende Schlussfolgerungen zu ziehen [JKS06].

Im Bereich der externen und meist weniger strukturierten Daten bekommen dagegen das Internet bzw. die derzeit bedeutendsten auf dem Internet aufbauenden Dienste *World Wide Web* und *E-Mail* einen immer höheren Stellenwert in der Unternehmens-IKT [Spr06][FKK10]. Dank der stetig wachsenden Speicherkapazitäten und Netzwerkbandbreiten wird insbesondere das World Wide Web dabei immer mehr zu einer umfangreichen Informationsbasis, die Unmengen an Wissen birgt. Die Schwierigkeit der Erschließung dieses Wissens besteht zunehmend in der Strukturierung der vorhandenen Informationen, in deren Bewertung und der Filterung im jeweiligen Kontext relevanter bzw. irrelevanter Informationen. Ein entscheidender Wettbewerbsvorteil kann daher die Kenntnis solcher Informationsquellen und die Fähigkeit zu deren Auswertung sein. Damit kann ein Wissensvorsprung gegenüber Wettbewerbern aufgebaut und im Idealfall dauerhaft gepflegt werden (vgl. [SBH+08]).

Im Kontext dieses Web-Szenarios ist jedoch offensichtlich, dass einfache Faktenrecherchen, wie sie den Großteil der heutigen Websuchen ausmachen, hier nicht zielführend sind bzw. den entscheidenden Wettbewerbsvorteil nicht nachhaltig werden liefern können: eine Faktenrecherche wird nie wirklich Neues zu Tage bringen sondern lediglich bereits existierendes Wissen zugreifbar machen. Zu einfach und naheliegend ist dieser Ansatz, als dass er nicht ohnehin

von allen Beteiligten bereits genutzt würde. Entscheidend kann jedoch sein, Zusammenhänge aufzudecken, Querbezüge zu identifizieren und Sammlungen von Fakten vor einem bestimmten Informationshintergrund zu interpretieren. Auf diese Art und Weise kann neues Wissen generiert werden und ein echter Wissensvorsprung und damit ein Wettbewerbsvorteil aufgebaut werden. Die Suche nach dieser Art Informationen ist jedoch ungleich komplexer als die reine Recherche von Fakten [FKK10]. Sie wird im Folgenden als „komplexe Informationsrecherche“ diskutiert.

1.1 Durchgängiges Beispiel und motivierendes Szenario

Die vorliegende Arbeit entstand im Rahmen des Forschungsprojekts *nova-net – Innovation in der Internetökonomie* [SMS+08] im Unterprojekt „Expertensuche“. Grundlage für die in diesem Unterprojekt durchgeführten Arbeiten war die Beobachtung, dass insbesondere bei kleinen und mittelständischen Unternehmen (KMU) oftmals unternehmensinterne Expertise nicht in dem Umfang vorhanden ist, wie sie benötigt wird, um sich auf neue Themen- und Technologiefelder zu wagen und dort erfolgreich zu bestehen [Sau99][KSS+07]. Dabei können Unsicherheiten unterschiedlicher Art den Erfolg unternehmerischer Aktivitäten gefährden: es handelt sich zumindest teilweise um technologisches Neuland, soziale Komponenten innerhalb des Akteursnetzwerkes sind dem Erfolg des Projektes nicht förderlich, die Kenntnis des Marktes ist nicht ausreichend etc. Oftmals erfolgt auch der Blick auf die Aktivität durch die Brille des Unternehmens bzw. der beteiligten Unternehmen und ist damit potentiell eingeschränkt oder beeinflusst durch wirtschaftliche und technische Rahmenbedingungen, was eine objektive Betrachtung der Probleme und Herausforderungen erschwert. Neben der Kenntnis konkreter Fakten wie bspw. wirtschaftlicher oder sozialer Rahmenbedingungen, sind vielfach auch implizite Wissensbestandteile [Pol85] gefragt: Wissen über die Anwendung und Interpretation von Informationen und über die Erschließung weiterer Informationsquellen.

Mit der Consulting-Branche baut ein kompletter Dienstleistungszweig auf eben

dieser Tatsache auf und bietet externes, hochspezialisiertes Wissen als Ergänzung zu den Kernkompetenzen des Beratung suchenden Unternehmens. Diese Experten zeichnen sich nicht nur durch explizites Wissen auf dem betrachteten Themenfeld aus, sondern auch durch ihre Fähigkeit, intuitiv bzw. auf Basis von implizitem Wissen die richtigen Interpretationen, Hinweise, Anhaltspunkte oder Handlungsanweisungen zu liefern. Grundlage hierfür sind die umfangreichen Kompetenzen eines Experten auf seinen jeweiligen Fachgebieten und der damit einhergehende Überblick sowie sein Verständnis für komplexe Zusammenhänge. Dadurch lassen sich hochspezialisierte Wissensquellen projektbezogen nutzen, das Risiko einer eingeschränkten Sichtweise auf die aktuelle Fragestellung wird reduziert und Handlungsalternativen können ggf. aufgezeigt werden [JKS05][JKS+07]. Neben den etablierten Consulting-Firmen spielen aber insbesondere bei KMU auch unabhängige Experten aus Industrie und Forschung eine zentrale Rolle bei der Einbringung externen Wissens. Sollen aber bei einer Produktentwicklung einzelne Experten themenbezogen hinzugezogen werden, so besteht eine entscheidende Herausforderung darin, die richtigen Experten ausfindig zu machen [SBH+08]. Verschiedene Quellen können hierzu genutzt werden: Bücher, Fachzeitschriften, Bibliothekskataloge, persönliche Kontakte und nicht zuletzt das World Wide Web mit seiner umfangreichen und stetig wachsenden Informationssammlung.

Die Suche nach geeigneten externen Experten gestaltet sich jedoch häufig schwierig und äußerst zeitintensiv. Vor diesem Hintergrund sollte im Rahmen von *nova-net* unter anderem untersucht werden, in wie weit das World Wide Web dabei als Informationsquelle für die Suche nach externen Experten herangezogen werden kann, welcher Grad an automatisierter Unterstützung der Suchende dabei erfahren kann und wie eine Vorgehensweise idealerweise auszusehen hat, um diese Suche effizient zu gestalten.

Diese Fragestellung lieferte die Motivation und den Anwendungsfall für die hier behandelten Konzepte zur informationstechnischen Unterstützung komplexer Webrecherchen. Da die Praxisrelevanz dieser Fragestellung im Rahmen des *nova-net*-Projektes in Zusammenarbeit mit Partnern aus Industrie und Forschung nachgewiesen wurde, dient dieses Szenario gleichzeitig zur Evaluierung

der Praxisrelevanz und -tauglichkeit der erarbeiteten Konzepte und Lösungen. Im Folgenden wird ein solches Szenario anhand eines beispielhaften Innovationsvorhabens beschrieben.

Ein mittelständischer Outdoor-Ausrüster entwickelt und produziert unter anderem Stirnlampen. Da die Kunden vermehrt nach helleren und gleichzeitig stromsparenden Modellen fragen, erfolgte vor einiger Zeit eine Orientierung hin zu LED-Technologie. Nach ersten Erfolgen mit derartigen Lampen wurde schnell klar, dass in dieser Technologie weiteres Potenzial steckt.
Neben der hohen Lichtausbeute ermöglichen LED-Lampen auch den Einsatz spezieller Optiken, die eine bessere Ausleuchtung des Blickfeldes im Vergleich zu herkömmlichen Optiken erreichen. Da in der Zielgruppe dieses Produktes typischerweise ein hohes Umweltbewusstsein herrscht, scheint eine Abkehr von Batterietechnologie und eine Orientierung hin zu alternativen Energiequellen sowohl technisch wie auch aus Marketingsicht ein sinnvoller Weg zur Festigung und zum Ausbau der Marktposition zu sein. Schnell erfolgte dabei eine Fokussierung auf die Brennstoffzellentechnologie.
Für den Bereich der Optiken konnte über persönliche Kontakte ein Technologiepartner gefunden werden, mit dessen Hilfe die Entwicklung einer optimierten Stirnlampe erfolgreich vorangetrieben wurde. Für den Bereich der alternativen Energiequelle hingegen waren die Vorstellungen noch nicht konkret und auch kompetente Partner konnten noch nicht identifiziert werden. Beides, die Konkretisierung der Vorstellungen im Sinne von „welche Technologien existieren, was ist möglich, was ist nicht möglich“ und die Identifizierung von Technologiepartnern sollte daher zeitnah vorangetrieben werden. Da bekannte Quellen keine weiteren Informationen mehr liefern konnten, sollte die Suche zunächst auf das World Wide Web ausgedehnt werden, um technische Hintergründe zu erkunden und Kontakte anzubahnen. Schnell wurde dabei deutlich, dass existierende Suchmaschinen diesen Prozess nur ungenügend unterstützen und mächtigere Suchwerkzeuge von Nöten war.

Es wurde daher ein Softwaresystem bzw. eine hinter diesem stehende Methodik entwickelt, die gemeinsam bei der Konkretisierung der Idee helfen sollten. Dazu sollte zunächst ein Überblick über mögliche Technologien generiert werden, über deren Vor- und Nachteile sowie ihre technische Reife und Einsetzbarkeit im mobilen Outdoor-Umfeld. Die Suche selbst sollte zum einen für das Themengebiet relevante Dokumente liefern und diese für weitere Analysen aufbereiten bzw. strukturieren. Zum anderen sollten Personen und Firmen identifiziert werden, die sich durch Erfahrungen oder spezielle Kenntnisse auf den betreffenden Gebieten auszeichnen und somit direkt als Technologiepartner oder aber indirekt als Wissenslieferant bzw. zur Herstellung von Kontakten zu potenziellen Technologiepartnern fungieren können.
Sowohl die identifizierten Webressourcen, als auch die Personen- und Firmenliste sollten einem nachvollziehbaren Ranking unterzogen werden und Zusatz- bzw. Hintergrundinformationen für eine manuelle Nachprüfung und ergänzende manuelle Recherchen vorgehalten werden.

In dieser Arbeit wird diese beispielhafte, in der Praxis als hochaktuell erachtete Problemstellung [KSS+07] aus unterschiedlichen Blickwinkeln betrachtet. Zur Verdeutlichung verschiedener Probleme wird dazu die Suche nach Experten im World Wide Web als Beispiel angeführt. Wie in den folgenden Kapiteln gezeigt wird, stellt diese Suche nach Experten dabei einen Spezialfall einer komplexen Informationsrecherche dar. Viele hier diskutierte Probleme und Lösungsansätze sind aber nicht auf die Expertensuche allein beschränkt, sondern gelten für viele Typen von Informationsrecherchen. Die Expertensuche an sich wird daher in dieser Arbeit als motivierendes Szenario verstanden, anhand dessen die Problemstellung und Lösungsansätze erarbeitet werden. Das Hauptaugenmerk dieser Arbeit liegt aber nicht in der Unterstützung der Expertensuche, sondern viel mehr auf generischen Konzepten zur Unterstützung allgemeiner komplexer, meist interaktiver oder „explorativer“ [Mar06][WR09] Informationsrecherchen. An verschiedenen Stellen wird die Expertensuche dabei als Beispielanwendung aufgegriffen um die entwickelten Konzepte zu verdeutlichen. Im Anschluss werden diese Ansätze anhand dieses Beispielszenarios bewertet.

1.2 Beiträge dieser Arbeit

Grundlage für sämtliche Aktivitäten im Rahmen dieser Arbeit war die Erkenntnis, dass zur Unterstützung komplexer Webrecherchen kaum Software-Werkzeuge zur Verfügung stehen. Wo diese existieren, da sind sie nur isoliert als Lösungen für bestimmte Aspekte der Suche verfügbar. Ihre Integration zur durchgehenden Nutzung dieser Lösungen über den gesamten Suchprozess hinweg wird jedoch nicht unterstützt. Um hier Abhilfe zu schaffen, und insbesondere um die Entwicklung solcher integrierter Suchanwendungen zu unterstützen, wurde eine Softwareplattform konzipiert, die eben diese Integration einzelner Such- und Analysekomponenten ermöglicht. Diese Suchplattform stellt damit ein Werkzeug für Entwickler von Suchanwendungen dar. Sie enthält zum einen eine Sammlung verschiedener Module zur Durchführung komplexer Webrecherchen. Zum anderen bildet sie einen Rahmen für die Erweiterung mit neuen Modulen, die auf spezielle Fragestellungen angepasst sind.

Neben der Plattform als Basis für die Entwicklung neuer Suchanwendungen wurden ferner drei Themen bearbeitet, die sich mit Kernfragen komplexer Webrecherchen befassen. Die hier erzielten Ergebnisse wurden als Bausteine in die Suchplattform integriert und bilden somit einen Teil ihrer Grundfunktionalität.

Das erste dieser drei Themen betrifft die Entwicklung von Suchanwendungen auf Basis der Plattform. Soll eine bestimmte Webpräsenz oder eine Gruppe von Webpräsenzen bei der Suche besondere Beachtung finden oder die Suche gar auf diese beschränkt sein, so kann sich die Entwicklung einer speziellen Zugriffskomponente anbieten, die Wissen über Inhalte und Strukturen der Ressourcen nutzt. Eine solche angepasste Komponente erlaubt es, das Maximum an verwertbaren Informationen aus der Quelle zu gewinnen. Im Rahmen der Suchplattform wurde dazu ein eigenständiges Konnektorframework entwickelt, das eine solche Implementierung unterstützt. Das Framework bietet Funktionalität für das Verbindungsmanagement, es implementiert verschiedene Authentifizierungsmechanismen und bildet den Rahmen für eine webpräsenz- bzw. ressourcenspezifische, gezielte Extraktion relevanter Informationen.

Das zweite Kernthema ist im Kontext der Bewertung von Suchergebnissen angesiedelt. Es wurde ein Maß für die Berechnung der inhaltlichen Ähnlichkeit von Dokumenten entwickelt, um im Unterschied zu herkömmlichen Suchmaschinen die Suche nicht nur anhand syntaktischer sondern auch inhaltlicher Kriterien durchführen zu können. Dabei werden die zu untersuchenden Texte mit Hilfe von Text- und Verknüpfungsinformationen der Online-Enzyklopädie *Wikipedia* auf inhaltliche Konzepte abstrahiert und auf Basis dieser Konzepte miteinander verglichen. Dieser Ansatz auf Basis von Konzepten hilft, schnell Dokumente zu einem bestimmten Thema zu finden, auch wenn teilweise andere Begrifflichkeiten in den Texten verwendet werden oder je nach Text andere Schwerpunkte gesetzt sind.

Das dritte Thema schließlich beschäftigt sich mit der Extraktion von Informationen aus für relevant befundenen Webressourcen. Dabei wird das Problem angegangen, dass existierende Systeme zur Erkennung von Entitäten wie Namen, Adressen, Telefonnummern etc. in natürlichsprachigen Texten meist dann gute Ergebnisse erzielen, wenn sie auf Texte angewandt werden, die den zu ihrem Training bzw. ihrer Entwicklung benutzten ähneln. Werden diese Systeme jedoch auf beliebige Texte, wie sie im WWW zu finden sind, angewandt, so sinkt die Erkennungsleistung rapide. Anhand zweier Ansätze zur Kombination mehrerer solcher Systeme und zur Nutzung von bereits erkannten Entitäten, konnte auch auf solchen Texten die Erkennungsleistung signifikant erhöht werden.

Diese vier Teilergebnisse – Suchplattform, Konnektorframework, Dokumentenähnlichkeit und Entitätenextraktion – bilden das Grundgerüst für die Suchmaschine *EXPOSE*. Sie dient der Identifizierung von Experten auf frei definierbaren Themenbereichen und behandelt damit ein klassisches Problem der komplexen Webrecherche. Diese Suchmaschine wurde im Rahmen des Forschungsprojekts *nova-net – Innovation in der Internetökonomie* entwickelt und sowohl im wissenschaftlichen als auch in der Praxis im industriellen Umfeld erfolgreich eingesetzt.

1.3 Gliederung

Im Folgenden wird ein Überblick über die verbleibenden Kapitel dieser Arbeit gegeben.

Kapitel 2: Definition und Analyse der Problemstellung
Es wird die dieser Arbeit zugrunde liegende Problemstellung aufgezeigt und die in Kapitel 4 vorgeschlagenen Lösungsansätze motiviert.

Kapitel 3: Grundlagen und verwandte Arbeiten
Ein Überblick über aktuelle Arbeiten im Bereich der explorativen und konventionellen Suche im World Wide Web wird gegeben und Grundlagentechnologien werden diskutiert. Ferner werden Ansätze zu weiteren Themen wie bspw. der Expertensuche vorgestellt, die diese Arbeit mittelbar oder unmittelbar betreffen.

Kapitel 4: Softwareplattform für Suchanwendungen
Ausgehend von der in Kapitel 2 vorgestellten Problemstellung werden konzeptionelle Lösungsansätze erarbeitet und eine Softwareplattform entwickelt, die sowohl Entwickler von Suchwerkzeugen, als auch deren Benutzer bei der Bearbeitung von komplexen Suchproblemen unterstützt.

Aufbauend auf dieser Plattform werden in den folgenden Kapiteln 5-7 konkrete Ansätze untersucht, die die Bearbeitung von komplexen Suchaufgaben effizienter gestalten. Die drei vorgestellten Ansätze zielen auf jeweils unterschiedliche Ebenen der Suchplattform, vom Zugriff auf Webressourcen über inhaltliche Analysen und Vergleiche bis hin zur Informationsextraktion.

Kapitel 5: Ein Konnektor- und Analyseframework
Zunächst wird in Kapitel 5 ein Framework präsentiert, das die Entwicklung von Zugriffs- und Analysekomponenten für spezielle Webpräsenzen erleichtert und damit Kontextwissen nutzbar macht, um qualitativ hochwertige Informationen zu extrahiern.

Kapitel 6: Berechnung von inhaltlicher Dokumentenähnlichkeit
Es wird eine Technik zur Bewertung von Dokumentenähnlichkeit diskutiert. Diese konzentriert sich, anders als die syntaxbasierten Maße herkömmlicher

Suchmaschinen, auf die inhaltliche Ähnlichkeit von Texten. Anwendung findet diese Technik bei der Suche nach Dokumenten zu einem vom Benutzer spezifizierten Themengebiet.

Kapitel 7: Erkennung von Namensnennungen in Texten
Dieses Kapitel behandelt Konzepte zur Erkennung von Namensnennungen in Texten. Der Fokus des hier präsentierten Ansatzes liegt in der Robustheit gegenüber Schwankungen von Textart und Genre der zu untersuchenden Texte.

Kapitel 8: Zusammenfassung & Ausblick
Die zentralen Fragestellungen und die dazu erarbeiteten Lösungskonzepte werden zusammengefasst und gegenüber gestellt. Es erfolgt eine abschließende Beurteilung der Möglichkeiten und Grenzen der vorgeschlagenen Ansätze und mögliche Anknüpfungspunkte zur Weiterführung der Arbeit werden aufgezeigt.

KAPITEL 2

DEFINITION UND ANALYSE DER PROBLEMSTELLUNG

In diesem Kapitel werden die Probleme aufgezeigt, die diese Arbeit motivieren und für die in den folgenden Kapiteln Lösungsvorschläge erarbeitet werden. Nach einer einführenden Begriffsklärung wird erläutert, welche Arten von Websuchen in dieser Arbeit betrachtet werden und worin die Schwierigkeiten bzw. Herausforderungen bei der Bearbeitung solcher Such-Fragestellungen liegen.

Das Thema dieser Arbeit ist die Informationsgewinnung im Rahmen von komplexen Suchen im WWW. Unter dem Begriff „komplexe Suche“ soll eine Art von Websuche verstanden werden, die sich in Ablauf und Erwartung an das Ergebnis signifikant von herkömmlichen Websuchen unterscheidet.

“Herkömmliche Websuchen” meint in diesem Zusammenhang die Art von ad-hoc-Suche, die ein durchschnittlicher Internetnutzer aus seiner regelmäßigen Anwendung von Websuchmaschinen kennt. Es handelt sich dabei klassisch um die Suche nach Fakten bzw. „known issues“ und weniger um eine analytische

Suche, die die Interpretation und eine weiter gehende Untersuchung der Suchergebnisse erfordern würde. Diese Art der Suche zeichnet sich dadurch aus, dass sie, abgesehen von der eingesetzten Suchmaschine, ohne nennenswerte Softwareunterstützung erfolgt, dass sie vielfach ungeplant und spontan erfolgt und in der Konsequenz, dass sie keiner fundierten Methodik folgt.

Insbesondere der letztgenannte Aspekt ist nicht als pauschale Kritik an dieser Art der Suche zu verstehen. Vielmehr bewährt sich diese Art der Suche täglich millionenfach und stellt ein hocheffizientes Mittel der Informationsgewinnung dar. Allerdings ist die damit lösbare Klasse der Probleme bzw. der damit befriedigbare Typ eines Informationsbedürfnisses stark eingeschränkt auf verhältnismäßig einfache Fragestellungen, wie die schlichte Recherche von Fakten im WWW. Die zugrunde liegende Fragestellung ist dabei einfacher Natur und meist ist eine hinreichend detaillierte Spezifikation des Informationsbedarfs anhand weniger Schlüsselworte möglich. Beispiele für eine solche Fragestellung sind „wann wurde die Brennstoffzelle erfunden“, „welche Firmen bewegen sich auf dem Feld der Nanotechnologie“, „welche Forschungseinrichtungen arbeiten auf dem Gebiet der Innovationsforschung“ etc. Die nötigen Anfragen an klassische Suchmaschinen lassen sich verhältnismäßig einfach aus den o. g. Fragestellungen ableiten, indem die relevanten Schlüsselbegriffe extrahiert werden. Bei der Auswertung der Suchmaschinenergebnisse jedoch, gibt es bereits gravierende Unterschiede: Die Frage nach der Erfindung/Entdeckung der Brennstoffzelle kann einfach anhand des nächstbesten Eintrages eines Online-Lexikons beantwortet werden. Hingegen erfordert die Beantwortung der Frage nach listenartigen Ergebnissen, wie in den beiden anderen Beispielfragestellung, eine Verarbeitung potenziell großer Dokumentenmengen, um die dort gespeicherten Informationen zu extrahieren und zusammen zu stellen [FKK10]. Dies lässt sich dahingehend zusammenfassen, dass eine Recherche genau dann einfach ist, wenn diese oder eine ähnliche Recherche zuvor bereits von anderen Personen durchgeführt wurde, und die Ergebnisse dieser Recherche („known issues“) im WWW publiziert wurden. D. h. die Antwort auf die Fragestellung muss direkt in Form von Texten, Zahlen, Verweisen o. ä. verfügbar sein und darf nicht das Ergebnis einer semantischen Analyse dieser sein. Zudem muss

sich die Fragestellung direkt aus einem Text beantworten lassen und darf nicht auf Bezüge zwischen mehreren Texten abzielen. Ist jedoch die Extraktion von Informationen aus verschiedenen Dokumenten für eine Beantwortung der Frage nötig oder müssen Querbezüge untersucht werden, dann erfordert dies einen Aufwand, der derzeit nicht automatisiert von Suchmaschinen durchgeführt wird, sondern manuell erfolgen muss.

Ist eine einfache Faktenrecherche nicht zielführend weil die benötigten Informationen nicht in aufbereiteter Form vorliegen, so erfolgt der Umweg über eine Dokumentenrecherche. Dabei werden Dokumente gesucht, die das betreffende Thema behandeln und diese Dokumente im Anschluss analysiert. Diese Art der Suche wird im Folgenden als „komplexe Suche“ oder „explorative Suche“ [Mar06][WR09] bezeichnet.

Kennzeichnend für sie ist der hohe manuelle Aufwand, den ein Suchender leisten muss, um an die gewünschten Informationen zu gelangen. Dieser hohe manuelle Aufwand kann aufgrund verschiedener Rahmenbedingungen entstehen: Im Unterschied zur o. g. herkömmlichen Suche handelt es sich bei den gesuchten Informationen nicht um reine Fakten, die explizit in textuellen Webressourcen gefunden werden können. Die Informationsgewinnung beinhaltet daher neben der reinen Suche nach relevanten Daten auch deren semantische Analyse und ggf. die Einbeziehung verschiedener solcher Ressourcen, um Informationen aus Querbezügen zwischen diesen ableiten zu können. Beide Komponenten, die semantische Analyse sowie die Analyse von Beziehungen zwischen Ressourcen, sind nur mit entsprechend hohem Aufwand realisierbar. Geschieht dies manuell, so erfordert bspw. die semantische Analyse ein detailliertes Lesen und Verstehen der Quellen, um zum einen die tatsächlich relevanten Dokumente zu filtern und zum anderen deren Informationsgehalt zu erfassen. Um Querbezüge zu analysieren, müssen offensichtlich große Mengen an Ressourcen analysiert werden, um zum einen die Existenz solcher Bezüge zu identifizieren und zum anderen, um diese zu klassifizieren und zu quantisieren. Auch hier ist ein hoher manueller Aufwand die Konsequenz.

Daneben ist diese Art der Suche durch einen hohen Grad an Unschärfe gekenn-

zeichnet, der sich durch Teile des Suchprozesses oder den gesamten Suchprozess hindurch zieht. Beispiele für diese Unschärfe sind i) Der Suchende ist sich nicht im Klaren darüber, was er überhaupt sucht, er hat lediglich eine mehr oder weniger ausgeprägte Idee anhand derer er die Suche durchführt. ii) Im Laufe einer – möglicherweise lang laufenden – Suche präzisiert oder ändert sich die Fragestellung aufgrund neu gewonnener Erkenntnisse und besseren Verständnisses des Suchenden für das betrachtete Themengebiet.

Solche „explorativen Suchen“ sind hochgradig interaktiv und verlangen vom Suchenden eine aktive Beteiligung über alle Phasen der Suche hinweg. Für eine Optimierung der Effizienz und der Effektivität solcher Suchen werden jedoch Softwaresysteme benötigt, die den Benutzer bspw. bei der Re-/Formulierung von Suchanfragen und bei der Analyse von Suchergebnissen weiter gehend unterstützen, als dies von herkömmlichen Suchmaschinen geleistet wird.

Offensichtlich sind die Grenzen zwischen der „herkömmlichen Suche“ und der „komplexen Suche“ in der Praxis nicht so starr wie hier präsentiert. Vielmehr sind sie als fließend zu bezeichnen und jede Art von Suche ist eher tendenziell in die eine oder andere Klasse einzuordnen. Dennoch soll diese binäre Klassifizierung als Diskussionsgrundlage für die folgenden Kapitel dienen, und darauf aufbauend Konzepte zur Unterstützung der „komplexen Suche“ erarbeitet werden. Da sich die meisten realen Suchen nicht 1:1 auf die beiden genannten Klassen abbilden lassen, werden auch die entwickelten Konzepte nicht zu 100% für alle diese Suchen passen, geschweige denn eine ultimative Lösung für deren zugrunde liegende Problemstellung bieten. Dennoch können sie für den Informationssuchenden eine entsprechende Erleichterungen bei der Suche darstellen und somit zu einer effizienteren und effektiveren Suche beitragen.

Komplexe Informationsrecherche war in den letzten Jahren Gegenstand vieler akademischer und industrieller Untersuchungen. Prominente Beispiele sind das deutsch-französische Forschungsvorhaben „Quaero“[1] bzw. das daraus hervorge-

[1] http://www.quaero.org

gangene Vorhaben „THESEUS“[1]. Auch die Industrie beteiligt sich dabei in signifikantem Umfang an der Erforschung der Herausforderungen komplexer Suche. Illustres Beispiel ist ein System der IBM zur automatisierten Beantwortung von Fragen, das unter dem Namen „Watson“ gegen menschliche Gegenspieler beim Wettbewerb „Jeopardy“ antritt[2]. Daneben existieren unzählige Einzelvorhaben, die sich mit den unterschiedlichen Aspekten der Fragestellungen zu komplexen Informationsrecherchen beschäftigen.

Wie in Kapitel 1 angeführt, wurde diese Arbeit durch das BMBF-Forschungsprojekt *nova-net – Innovation in der Internetökonomie* motiviert. Eine zentrale Fragestellung im Rahmen dieses Projektes war die Suche nach unternehmensexternen Experten zur Konsultation im Rahmen von Innovationsprozessen. In der Praxis eröffnen sich dem Suchenden hierzu verschiedene sich ergänzende Quellen, in denen nach den gewünschten Personen oder Institutionen gesucht werden kann. Zu diesen gehören zuallererst der Fundus an persönlichen Kontakten des Suchenden, bspw. Kollegen, Geschäftspartner oder anderweitig für ihre Kenntnisse auf dem jeweiligen Themengebiet bekannte Personen. Die Herausforderung hierbei besteht also in erster Linie darin, zu wissen, wer die benötigten Informationen und Wissen besitzt und darin, entsprechende Kontakte herzustellen.

Kann anhand dieser direkten Kontakte kein befriedigendes Ergebnis erzielt werden, so muss auf bisher unbekannte Personen zurückgegriffen werden. Dabei zeigt sich die klassische Gegenläufigkeit zweier Optimierungziele – „Precision“ und „Recall“: aufgrund persönlicher Bekanntschaften kann für einige wenige Personen noch mit hinreichender Gewissheit (=„Precision“) eine Einschätzung ihrer Expertise und Verlässlichkeit erreicht werden. Wird jedoch der Suchraum erweitert, d.h. der „Recall“ soll vergrößert werden, so ist dies nicht mehr in gleichem Maße möglich. Es fließen mehr und mehr Unsicherheiten in die jeweilige Einschätzung ein und reduzieren damit die Aussagekraft der Suchresultate.

Eine solche Erweiterung des Suchraumes und damit des „Recalls“ wäre bspw.

[1] http://www.theseus-programm.de
[2] http://www-03.ibm.com/innovation/us/watson

die Nutzung von Yellow Pages Systemen oder dedizierten Expertennetzwerken. Sie stellen Verzeichnisse bereit, in denen Themen und Fachgebiete einzelnen Personen oder Institutionen zugeordnet sind. Neben der reinen Themenzuordnung finden sich hier oft weitere Informationen über die Vernetzung der gelisteten Personen, deren Kontaktdaten etc. Hier allerdings aus der großen Masse diejenigen Treffer zu isolieren, die tatsächlich von Nutzen sind, stellt eine große Herausforderung für den Suchenden bzw. die ihn unterstützende Software dar.

Aus der vorangegangenen Schilderung der Herausforderungen der Expertensuche wird deutlich, dass es sich bei dieser Art von Suche um einen Vertreter der „komplexen Suche“ handelt. Es müssen Personen oder Institutionen identifiziert, deren Fähigkeiten und Erfahrungen quantifiziert werden und schließlich sind Beziehungen zwischen ihnen aufzudecken und zu analysieren, um evtl. vorhandene Expertennetze zu identifizieren. Die Expertensuche soll daher in dieser Arbeit als praktisches Beispiel einer „komplexen Suche“ dienen und die Anforderungsanalyse sowie die entwickelten Konzepte motivieren. Dennoch sei an dieser Stelle erneut darauf hingewiesen, dass die entwickelten Konzepte nicht ausschließlich für die Expertensuche dienlich sind, sondern den Anspruch haben, auch für andere Arten von Fragestellungen Lösungsansätze zu bieten.

Eine informationstechnische Unterstützung komplexer Suchen im Web ist mit verschiedenen Schwierigkeiten konfrontiert, die sich unter anderem aus den oben genannten Rahmenbedingungen ergeben. Die Anforderungen an eine technische Umsetzung lassen sich dabei auf die im Folgenden genannten technischen und methodischen Herausforderungen zurückführen.

2.1 Rolle des Suchenden

Während das WWW als Informationsquelle stetig wächst, wird die Suche darin zunehmend aufwändiger. Herkömmliche Suchmaschinen liefern jedoch lediglich Dokumentensammlungen als Trefferliste. Dabei sind zwei Ansätze zu unterscheiden: zum einen erfolgt bei Suchmaschinen wie Google, MSN oder

Yahoo eine Filterung von Informationen anhand syntaktischer Kriterien, die der Benutzer spezifiziert. Dies ermöglicht eine schnelle und automatisierte Filterung potenziell relevanter Informationen, allerdings auf Kosten fehlenden semantischen Verständnisses für das zugrunde liegende Suchproblem. Andere Ansätze, wie bspw. die ontologiebasierte Suche, versuchen diese Semantik in den Suchprozess zu integrieren, können aber im Gegenzug nicht die Datenmengen verarbeiten, wie die zuvor genannten Systeme.

Eine weitere Verarbeitung der Inhalte mit dem Ziel, die tatsächlich benötigten Informationen zu identifizieren, bleibt daher dem Suchenden überlassen. Insbesondere bei komplexen Informationsrecherchen wie bspw. der Suche nach Experten ist dieser der Dokumentensuche nachgelagerte Schritt äußerst aufwändig und komplex. Hier gilt es, den Benutzer entsprechend zu unterstützen, um zum einen die Ergebnismenge handhabbar zu halten und zum anderen deren Analyse, hier im Beispiel die Identifizierung von Experten und die Bewertung deren Expertise, zu ermöglichen. Die Präsentation von Suchergebnissen, deren Bewertungen und vor allem die zu dieser Bewertung führenden Kriterien spielt in diesem Zusammenhang eine große Rolle [WDM+07][GKC+10]. Grafische Repräsentationen die über herkömmliche Trefferlisten hinaus gehen, helfen dem Benutzer hier besonders, da Zusammenhänge so meist schneller und einfacher erfasst werden können.

Neben der Schwierigkeit der Informationsextraktion aus von Suchmaschinen zurück gemeldeten Dokumentensammlungen, stellt sich dem Benutzer jedoch bereits bei der Formulierung der Suchanfrage eine oftmals sogar noch größere Herausforderung. Auf einfache Suchanfragen werden von Suchmaschinen Unmengen an Treffer geliefert, womit aber nicht zwangsläufig eine hohe Qualität und damit eine hohe Zufriedenheit des Suchenden einhergehen. Der Grund hierfür liegt nicht zuletzt in der Spezifikation der Suchanfrage. Einfache Schlüsselwortanfragen, die zudem keine Verknüpfung mittels erweiterter Operatoren enthalten (abgesehen vom in der Regel impliziten „ODER“), besitzen eine entsprechend schwache Semantik. Es ist aber offensichtlich, dass die Ergebnisse einer Suchanfrage im Allgemeinen nicht besser sein können, als die Qualität bzw. die Präzision deren Spezifikation. Analysen von Suchmaschinen-Protokollen

haben aber gezeigt, dass genau diese Präzision von der Mehrzahl der Benutzer nicht erreicht wird. Silverstein et al. [SMHM99] legen bereits 1999 in Ihrer Studie dar, dass die Mehrzahl der Suchanfragen an eine Suchmaschine sich auf drei Begriffe beschränkt. Von den zurückgemeldeten Ergebnissen werden meist nur die ersten zehn betrachtet. Eine Verfeinerung der Suchanfrage findet nur in relativ wenigen Fällen statt. Auch neuere Studien, die das Nutzerverhalten und seine Änderung über die Zeit analysieren, kommen zu dem Ergebnis, dass der Anteil der komplexeren Suchanfragen (mehrere Suchbegriffe, Einsatz von Anfrageoperatoren etc.) auf konstant niedrigem Niveau verharrt [JSP05] [JS06].

Eine mögliche Deutung dieses Sachverhalts ist auch, dass keine komplexen Anfragen benötigt werden, um den jeweiligen Informationsbedarf zu befriedigen. Für einfache Fragestellungen, bspw. bei der Suche nach Faktenwissen, trifft diese Annahme auch oftmals zu. Hier kann mittels einfacher Anfragen wie z. B. „Erfinder der Brennstoffzelle“ schnell und effizient der Informationsbedarf befriedigt werden. Ist jedoch der Informationsbedarf von komplexer Natur, bspw. bei der Suche nach neuen Entwicklungen auf einem speziellen Technologiefeld, so ist diese Art der Suchspezifikation nicht zielführend, da, wie oben ausgeführt, eine derart simple Anfrage nicht die benötigte Komplexität wiedergeben kann.

Zwar „kennen“ Suchmaschinen einen großen Teil der im WWW verfügbaren Dokumente und könnten anhand dieses Wissens zumindest theoretisch in Bezug auf Quellenverfügbarkeit auch den komplexen Informationsbedarf eines Suchenden befriedigen. Dem steht aber die oftmals geringe Suchkompetenz des einzelnen Suchenden entgegen, der seinen Informationsbedarf nicht in einer hinreichend präzisen Anfrage spezifizieren kann.

Um dennoch den Benutzer bei der Suche nach hochwertigen Informationen zu unterstützen, müssen diese Defizite entsprechend durch Methoden- und Softwareunterstützung kompensiert werden. Insbesondere der Anfragespezifikation ist hierbei nach obigen Ausführungen besonderes Gewicht beizumessen. Zwar lassen sich allein anhand einer exakten Spezifikation des Informationsbedarfs noch keine qualitativ hochwertigen Informationen identifizieren. Dennoch baut

auf dieser Spezifikation der gesamte Suchprozess auf, weshalb sie eine wichtige Grundlage darstellt.

2.2 Struktur des WWW

Die heterogene Struktur der im WWW verfügbaren Informationen stellt Suchende und Suchmaschinen vor weitere Probleme. Während sich einfache Webressourcen noch mit verhältnismäßig einfachen Mitteln von Suchmaschinen indexieren lassen und damit dem Suchenden potenziell als Ergebnisse zurück gemeldet werden können, so stellt sich dies bei Ressourcen des sog. „Deep Web“ [Ber00] völlig anders dar. Unter dem Begriff „Deep Web“ werden diejenigen Ressourcen verstanden, die für Suchmaschinen nicht ohne weiteres erreichbar sind und deshalb von diesen nicht indexiert werden und in der Folge dem Benutzer auch nicht als Treffer für seine Suchanfrage präsentiert werden können. Ein Beispiel für derartige Ressourcen ist die Webschnittstelle zu einer Datenbank wie einem Bibliothekskatalog (Abbildung 2.1). Auf die in einer solchen Datenbank gespeicherten Informationen erhält der Benutzer lediglich durch das Ausfüllen von Formularen Zugriff. Die Inhalte selbst sind damit nur durch korrektes Ausfüllen der Formularfelder erreichbar. Eine Suchmaschine bzw. der von ihr betriebene Webcrawler (vgl. Kapitel 3.1.1) hat im Allgemeinen jedoch keine Kenntnis über die Semantik und damit über „korrekte“ oder zumindest „sinnvolle“ Eingaben für die Formularfelder. Demzufolge kann er auf die Inhalte der Datenbank nicht zugreifen und diese somit auch nicht für den Suchmaschinenbenutzer aufarbeiten.

Zwar gibt es Anstrengungen in Richtung der automatischen Erschließung solcher Inhalte, bspw. [WYDM04]. Dennoch sind aktuelle Suchmaschinen dieser Aufgabe bisher nicht gewachsen und werden es auch auf absehbare Zeit wohl nicht in signifikantem Umfang sein. Wie oben angeführt, sind dazu die Probleme in den Bereichen Semantik von Anfragefeldern und damit einhergehend der automatisierten Anfragegenerierung heute noch zu anspruchsvoll. Es ist also fraglich, ob solche datenbankartigen Informationsquellen überhaupt ohne

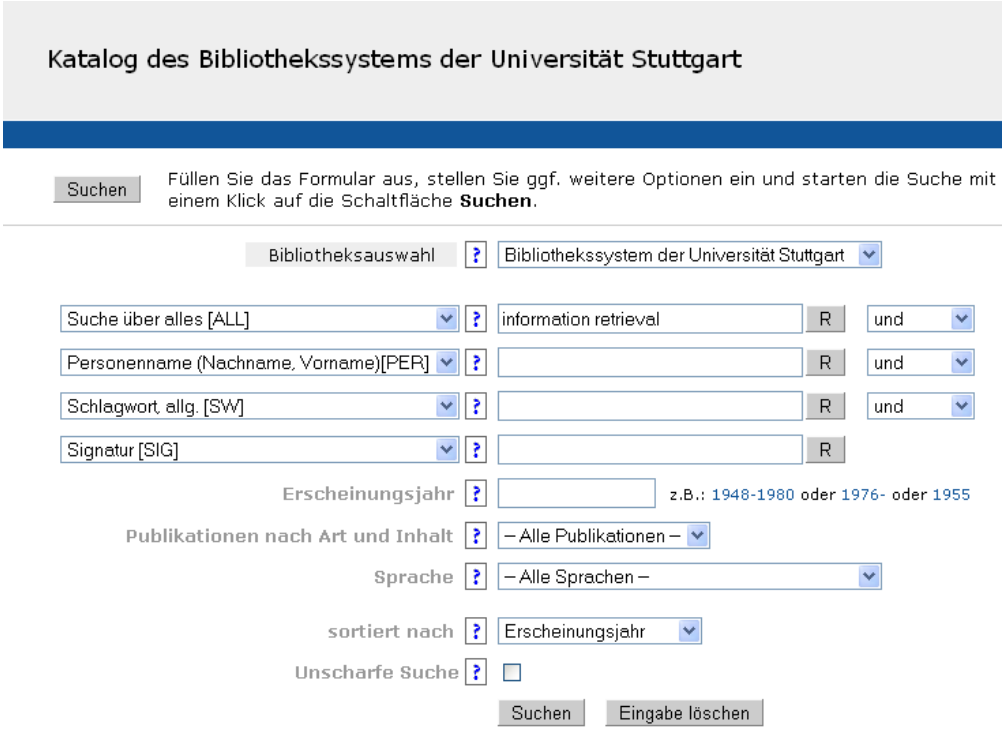

Abbildung 2.1: Schnittstelle zu Deep-Web Informationen am Beispiel der Suchmaske der Universitätsbibliothek Stuttgart

ein umfangreiches Domänenwissen mittels automatisch generierter Anfragen erschlossen werden können bzw. ob dieses Problem von einem generischen Suchsystem wie einer allgemeinen Websuchmaschine überhaupt bewältigt werden kann. Sollen dennoch die in solchen Ressourcen liegenden Informationsschätze gehoben werden, so werden Techniken und Werkzeuge benötigt, die den für die Erschließung dieser Ressourcen notwendigen manuellen Aufwand weitgehend reduzieren.

Neben den Datenbankschnittstellen existiert eine weitere Klasse an Deep-Web-Ressourcen: die der dynamisch generierten Webseiten, deren Inhalt u. a. vom jeweiligen Benutzer abhängt und damit personalisiert ist. Um auf diese Inhalte zugreifen zu können, ist im Vorfeld meist eine Authentifizierung bzw. Autorisierung am betreffenden System erforderlich. Dies stellt offensichtlich für generische Suchmaschinen ein großes Problem dar, da zum einen Zugangsdaten für diese Netze im Allgemeinen nicht bekannt sind, und zum anderen, sich die Inhalte von Nutzer zu Nutzer unterscheiden können. Eine Indexierung der für

Benutzer A sichtbaren Inhalte muss also nicht zwangsläufig auch für Benutzer B hilfreich sein. Ansätze, um diese Art von Quellen für die Suche zugänglich zu machen, finden sich bspw. in [RGM01] und werden auch in Kapitel 5 weiter ausgeführt.

2.3 Informationsextraktion aus unstrukturierten Texten

Eine grundlegende Schwierigkeit der komplexen Webrecherchen ist darin begründet, dass das Web zwar große Mengen an Informationen birgt, diese jedoch nicht in strukturierter Form vorliegen, sondern in der Regel unstrukturiert in natürlichsprachigen Dokumenten enthalten sind. Die Identifizierung von Informationen oder, weiter gehend, die Extraktion von Wissen aus unstrukturierten Daten wie natürlichsprachigen Texten ist jedoch ein hochkomplexes Problem. Softwaresysteme jedoch arbeiten nach festen Regeln. Wo diese Regeln nicht eindeutig sind oder zur korrekten Interpretation der untersuchten Daten weiteres Hintergrund- oder Kontextwissen benötigt wird, da stoßen diese Systeme an ihre Grenzen. Ferner ist auch aufgrund der Natur des WWW, Informationen zu beliebigen Themen vorzuhalten, die Struktur dieser Informationen beliebig heterogen. Es existiert eine Vielzahl unterschiedlicher Dienste und Plattformen, die alle ihren eigenen Regeln für die Verarbeitung und Präsentation von Inhalten folgen. Eine Orientierung hin zu maschinell verarbeitbaren Strukturen wird zwar im Rahmen verschiedener Semantic-Web-Aktivitäten [KM01] [DFJ+04] [VKV+06] angestrebt, konnte sich bisher aber nur in einzelnen speziellen Bereichen, wie bspw. Semantic Web Services [KTSW08] im industriellen Umfeld, durchsetzen. Wäre das WWW für die vollautomatisierte Interaktion von Softwaresystemen konzipiert, so würde sich die Darstellung der relevanten Informationen auf die reinen Inhalte und ggf. deren Metabeschreibung, also das Format dieser Inhalte beschränken. Da das WWW aber in erster Linie eine Plattform für menschliche Benutzer darstellt, finden sich neben den reinen informationsbeschreibenden Daten auch zusätzliche Daten – natürlichsprachige Texte zur Erläuterung der Daten, Diskussionen und weiterführende Verweise

zum Thema oder für den Suchenden zur Befriedigung seines Informationsbedarfs nicht direkt relevante Werbetexte und -grafiken.

Für den Suchenden stellt sich also das Problem, dass umfangreiche Datenmengen analysiert werden müssen, wozu jedoch nur in sehr eingeschränktem Umfang Unterstützung durch Software gegeben ist. Das bedeutet aber, dass mit allgemeinen Hilfsmitteln wie Suchmaschinen eine intelligente Suche nur schwer umsetzbar ist, da Suchmaschinen im Allgemeinen über zu wenig Wissen bezüglich der Inhalte der von ihnen indexierten Ressourcen verfügen. Sie können zwar helfen, den Suchraum auf diejenigen Ressourcen einzugrenzen, die einen thematischen Bezug zur Suchanfrage haben. Die Informationsextraktion bzw. in einem zweiten Schritt gar die Wissensextraktion kann von ihnen nach heutigem Stand jedoch nicht geleistet werden. Zwar sind Informationen und auch Wissen im WWW verfügbar, dieses aus den einzelnen Dokumenten zu extrahieren ist aber immer noch Aufgabe des Suchenden und herkömmliche Suchmaschinen bieten wenig bis keine Unterstützung bei diesem Analyse- oder Extraktionsprozess.

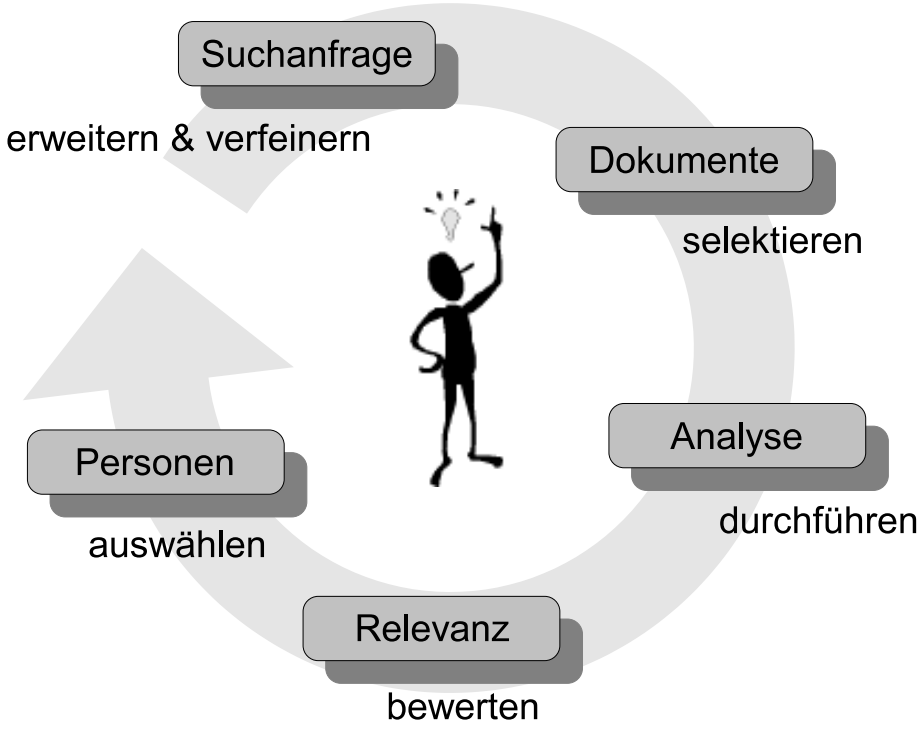

Abbildung 2.2: Beispielhafter Suchprozess zur Identifizierung von Experten

Abbildung 2.2 verdeutlicht das Problem der komplexen Suche anhand des Beispiels der Expertensuche. Auf eine Suchanfrage hin werden mittels herkömmlicher Suchmaschinen passende Dokumente gefunden. Zunächst werden nun die gefundenen Dokumente auf ihre Relevanz hin geprüft, um im Anschluss die auf den als relevant identifizierten Dokumenten genannten Personen zu identifizieren, die zudem den gewünschten Bezug zum Thema vermuten lassen. Die darauf folgende Arbeit ist ohne spezielle Softwareunterstützung monoton und äußerst aufwändig, da sie sich jedes Mal wiederholt: Es müssen weitere Quellen gesucht werden, die mit den identifizierten Person in Verbindung gebracht werden können, es muss das Verhältnis dieser Person zur entsprechenden Quelle ermittelt und schließlich bestimmt werden, ob diese Quelle relevant für das Thema ist. Finden sich im Rahmen dieses Prozesses weitere potentielle Experten, so lässt sich der Prozess rekursiv beliebig fortsetzen.

Bei aller Forderung nach softwareseitiger Unterstützung eines solchen Suchprozesses, darf jedoch nicht über das Ziel hinausgeschossen werden. Eine vollständige Automatisierung der Suche scheint nicht erstrebenswert zu sein, da der Prozess von zu vielen Unsicherheiten begleitet wird. Bspw. gelingt es auch versierten Benutzern nicht immer, Anfragen mit der nötigen Präzision zu spezifizieren. Das liegt unter anderem an den Unterschieden zwischen dem Vokabular des Suchenden und dem der betreffenden Ressourcen. Dieses Vokabular lässt sich nicht immer durch den Einsatz von Synonymen (automatisiert) aufeinander abbilden. Auch sind aktuelle Softwaresysteme noch nicht in dem Maße fähig, die Semantik eines natürlichsprachigen Textes zu verstehen, wie es zur exakten Analyse des Inhalts nötig wäre. Wie in Kapitel 2.1 dargestellt, kann aber das Ergebnis einer Suche im Allgemeinen nicht besser sein als deren Spezifikation. Im Falle einer unvorteilhaften Anfragespezifikation werden dann aber entsprechend schlechte (Zwischen-) Ergebnisse generiert.

Das macht deutlich, dass eine wesentliche Anforderung an ein solches Suchsystem ist, dem Benutzer Routinearbeiten abzunehmen, Texte möglichst exakt zu klassifizieren und gegeneinander abzugrenzen, Unterstützung bei der Identifizierung von Entitäten wie Personen-, Firmen- oder Ortsnamen zu bieten und Zusammenhänge zwischen diesen Entitäten und gefundenen Dokumenten

herzustellen bzw. diese zu visualisieren. Dabei muss der Benutzer stets die Möglichkeit haben, lenkend oder korrigierend in den Suchverlauf einzugreifen, um dem System aus wenig Erfolg versprechenden Suchräumen zu verhelfen und es entsprechend in eine andere Richtung zu leiten.

In der Praxis sind die oben aufgeführten Probleme die Hauptgründe für ineffiziente oder gar wenig effektive komplexe Webrecherchen. Zusammenfassend kann festgehalten werden, dass nicht speziell versierte Benutzer ohne entsprechende Unterstützung durch Softwarewerkzeuge die stetig wachsende Flut an unstrukturierten Daten im WWW kaum effizient nach den gewünschten Informationen durchsuchen können. Im Kontext des Forschungsprojekts *nova-net – Innovation in der Internetökonomie* wurde in Zusammenarbeit mit Forschungs- und Industriepartnern ein großer Bedarf nach einer solchen Unterstützung in der Praxis identifiziert [KSS+07], der in der vorliegenden Arbeit angegangen wird. In den folgenden Kapiteln werden einzelne Konzepte zur Lösung der o. g. Probleme diskutiert und deren Integration in einem flexiblen Suchframework präsentiert.

2.4 Zusammenfassung

Im vorliegenden Kapitel wurde die dieser Arbeit zugrunde liegende Problemstellung der Suche nach Informationen im WWW skizziert. Zunächst wurde dazu das Teilgebiet der „Komplexen Websuche“ beschrieben und gegen einfach strukturierte Websuchen abgegrenzt. Am Beispiel der Suche nach Experten im WWW wurde das Szenario der „Komplexen Websuche“ veranschaulicht.

Im Anschluss daran wurden die Schwierigkeiten herausgearbeitet, vor die Nutzer des WWW bei dieser Art von Suche gestellt sind. Diese Schwierigkeiten bilden die Motivation für die vorliegende Arbeit und werden in den folgenden Kapiteln zusammen mit Vorschlägen zu deren Lösung diskutiert.

KAPITEL 3

GRUNDLAGEN UND VERWANDTE ARBEITEN

In diesem Kapitel werden grundlegende Techniken und Ansätze präsentiert, die für die automatisierte oder teilautomatisierte Bearbeitung komplexer Websuchen von Bedeutung sind. Neben der Vorstellung der jeweiligen Techniken und Ansätze erfolgt eine Diskussion über deren Einfluss auf diese Arbeit bzw. die Abgrenzung gegenüber dieser.

3.1 Web-Suchmaschinen

Das WWW hat seit seiner Erfindung im Jahre 1989 [BL89] ein rasantes Wachstum erfahren. Von einem wissenschaftlichen Forschungsprojekt am schweizer CERN (European Organization for Nuclear Research) mit überschaubarem Inhalt und eingeschränktem Nutzerkreis hat es sich zu einer hochgradig heterogenen Sammlung von Ressourcen aller Art entwickelt, die von millionen Menschen weltweit genutzt werden. Jeder dieser Nutzer kann im Umfeld des

WWW mit geringem Aufwand beliebige Informationen publizieren. Die Art der publizierten Informationen ist dabei prinzipiell unerheblich und auch eine Prüfung des Inhalts auf Korrektheit findet nicht zwangsläufig statt. Aufgrund verschiedener Aspekte hat sich das Internet und insbesondere das WWW in den letzten Jahren als eine zentrale Komponente des geschäftlichen wie privaten Lebens weiter Bevölkerungsschichten etabliert. Internet-Dienste wie E-Mail, oder WWW-Plattformen wie Online-Versandhäuser, Online-Gemeinschaften, Soziale Netzwerke u. ä. haben sich in diesem Kontext entwickelt und machen das Internet bzw. das WWW sowohl im persönlichen als auch im beruflichen Alltag zu einem ständigen Begleiter vieler Menschen. Im Zuge dessen hat der Umfang des WWW, d. h. die Menge an publizierten Informationen, gemessen in der Anzahl verfügbarer Dokumente, in den letzten Jahren ebenfalls rasant zugenommen.

Um in dieser wachsenden Informationsmenge den Überblick zu behalten und gezielt auf benötigte Informationen zugreifen zu können, wurde früh entdeckt, dass der Nutzer dieser Dienste hierbei Unterstützung benötigt. Andernfalls wäre die wachsende Informationsflut kaum zu bewältigen. Diese Erkenntnis hat zwei grundlegend unterschiedliche Konzepte hervorgebracht: Zum einen haben sich manuell erstellte Verzeichnisse entwickelt, in denen Inhalte nach bestimmten Strukturen geordnet und damit suchbar gemacht werden. Die Suche erfolgt dabei entweder entlang der in der Struktur definierten Relationen, oder aber mittels Volltextsuche (vgl. Kapitel 3.1.2). Der andere Ansatz entwickelte sich aus der Erkenntnis heraus, dass eine manuelle Strukturierung auf verhältnismäßig kleine Bereiche des WWW beschränkt sein muss, da eine umfassende Verarbeitung aufgrund des hohen Aufwandes nicht realisierbar ist. Es wurden daher Verfahren entwickelt, die automatisch weite Teile des WWW absuchen, die gefundenen Inhalte analysieren, indexieren und dem Benutzer über einfache schlüsselwortbasierte Suchschnittstellen zugänglich machen.

Während der erstgenannte manuelle Ansatz auch heute noch im Bereich von Spezialanwendungen oder überschaubaren Themenbereichen angewandt wird, so hat sich im Großen doch das automatisierte Verfahren durchgesetzt und kommt in Form verschiedener Web-Suchmaschinen zum Einsatz.

Im Folgenden soll ein Überblick über Technologien gegeben werden, die in diesem Kontext verwendet oder erforscht werden. Zudem wird deren Bedeutung für die vorliegende Arbeit diskutiert.

3.1.1 Crawling

Aus Sicht des Benutzers aktueller Suchmaschinen besteht deren Sinn und Zweck darin, ihm Webressourcen zu präsentieren, die möglichst gut zu seiner Suchanfrage passen. Während Kapitel 3.1.3 das Problem behandelt, was genau die Bedeutung von „möglichst gut passen“ ist, soll hier zunächst betrachtet werden, wie Suchmaschinen ihr Wissen über verfügbare Webressourcen erlangen.

Um Aussagen über den Inhalt einer Webressource machen zu können, muss eine Suchmaschine diese zuvor gefunden und analysiert haben, wozu die betreffende Ressource durch die Suchmaschine vom Webserver des Anbieters angefordert und heruntergeladen werden muss. Prinzipiell unterscheiden sich hierbei zwei Ansätze, wann dieses Herunterladen erfolgt.

Ansatz 1: Die meisten gängigen Suchmaschinen folgen dem ersten Ansatz und stellen diesen Schritt an den – aus Sicht der Suchmaschine – Anfang des Suchprozesses. D. h., zuerst werden Webressourcen heruntergeladen, dann analysiert (u. a. indexiert, vgl. Kapitel 3.1.2) und schließlich dem Benutzer über den Index zugreifbar gemacht. Vereinfacht dargestellt folgt die Suchmaschine beim Herunterladen der Ressourcen, ausgehend von einigen Start-Ressourcen, rekursiv allen in diesen Ressourcen gefundenen Verweisen. Bereits besuchte Ressourcen werden bei erneutem Auftreten ignoriert und lediglich den Verweisen zu bislang unbekannten Ressourcen wird gefolgt. Dieser Vorgang wird allgemein als „Crawling“ bezeichnet.

Die großen Suchmaschinen bieten dem Benutzer allerdings nur einen eingeschränkten Zugriff auf ihren Index, nämlich in Form von Schlüsselwortanfragen, die ggf. mit verschiedenen Operatoren erweitert werden können. Sollen aber Analysen durchgeführt werden, die über diesen eingeschränkten Indexzugriff nicht effizient implementiert werden können, so ist die Nutzung alleine dieser

Suchmaschinenfunktionalität nicht zielführend und der Aufbau einer eigenen, erweiterten Datenbasis wird nötig. Typischerweise erfolgt dies wieder anhand eines Crawling-Mechanismus, der dem o. g. Prinzip folgt und ggf. an die speziellen Bedürfnisse des Suchenden bzw. der Fragestellung angepasst wird.

Um jedoch eine signifikante Teilmenge des WWW zu indexieren und damit für den Benutzer suchbar zu machen, muss ein solcher Crawler große Mengen an Webressourcen herunterladen und der Indexierung zuführen. Über die aktuelle Größe des WWW kann nur spekuliert werden und auch die großen Suchmaschinen sind mittlerweile davon abgekommen, die genaue Anzahl der von ihnen indexierten Webressourcen zu publizieren. Die letzten bspw. von Google veröffentlichten Zahlen stammen aus dem Jahr 2005 und weisen eine Größenordnung von ca. 8 Mrd. indexierten Ressourcen aus, wobei dies nicht die Anzahl aller im Web verfügbaren Ressourcen, sondern lediglich die Anzahl der von dieser Suchmaschine indexierten Ressourcen beschreibt.

Das in Kapitel 1 beschriebene Szenario, das diese Arbeit motiviert, ist ein Szenario, das im Wesentlichen die Situation kleiner und mittelständischer Unternehmen abbildet und das widerspiegelt, wie diese das Web nutzen, um an benötigte Informationen zu gelangen. Offensichtlich kann aber in diesem Umfeld der Aufwand nicht erbracht werden, eine komplette Suchmaschine zu implementieren bzw. zu unterhalten. Zum einen haben die betreffenden Firmen im Allgemeinen nicht das Know-How, eine solche Suchmaschine zu betreiben. Zum anderen sind die Kosten für die technische Ausrüstung und den Betrieb einer solchen Suchmaschine derart hoch, dass der mögliche Nutzen in keinem Verhältnis zu diesen Kosten stehen würde. Eine Informationssuche in der Art, wie der Benutzer es von herkömmlichen Suchmaschinen gewohnt ist, jedoch mit umfangreicheren Analysemöglichkeiten, scheint also zunächst nicht praktikabel.

Ansatz 2: Ein alternativer Ansatz zur oben angeführten Indexierung vor der Suche besteht nun darin, zwar mit einem eigenen Crawler das WWW zu durchsuchen, sich dabei jedoch lediglich auf diejenigen Bereiche des WWW zu konzentrieren, die sich mit den in der Suchanfrage spezifizierten Themen beschäftigen.

In [CvdBD99] wurde hierzu das Konzept des *Focused Crawlers* entwickelt, der genau dies ermöglicht. Ein solcher Crawler folgt einem modifizierten Konzept: herkömmliche Suchmaschinen durchsuchen zunächst das Web nach beliebigen Dokumenten, indexieren diese und stellen sie dem Benutzer dann über Suchanfragen auf dem Index zur Verfügung. Das Crawling durch die Suchmaschine ist dabei der Suche des Benutzers zeitlich vorgelagert.

Bei einem Focused Crawler hingegen erfolgen beide Schritte quasi gleichzeitig. Ausgangspunkt ist die Spezifikation des Informationsbedarfs durch den Benutzer, der damit die Suche auf ein bestimmtes Themengebiet „fokussiert“. Als zusätzliche Eingabe liefert er dem System einen Ausgangspunkt in Form von URLs bei denen der Crawler die Suche beginnen soll. Von diesen Start-URLs ausgehend, durchsucht nun der Crawler das Web und folgt den dort verlinkten Dokumenten rekursiv. Zentral für die Leistungsfähigkeit dieses Konzepts sind die Algorithmen des Crawlers, die die Reihenfolge festlegen, in welcher den gefundenen Verweisen gefolgt werden soll. Dies ist essentiell, da das Laden und Analysieren von Dokumenten teuer ist und daher in der Hauptsache potenziell relevanten Verweisen gefolgt werden soll. Hierzu kommen verschiedene Heuristiken zum Einsatz, wobei der zentrale Ansatz der ist, den Verweisen zu folgen, die auf den bisher am besten bewerteten Dokumenten gefunden wurden. Die Idee hinter diesem Vorgehen ist, dass thematisch zusammenhängende Dokumente oft auch über Hyperlinks miteinander verbunden sind und somit schnell eine große, qualitativ hochwertige Treffermenge generiert werden kann. Entscheidend hierbei ist, dass die Bewertung einer Ressource nicht durch die allgemeingültigen Algorithmen der Standardsuchmaschinen erfolgen muss, sondern eigene Bewertungsschemata zum Einsatz kommen können. Damit kann eine deutlich bessere Adaption an das Suchszenario realisiert werden, was letztlich bessere Ergebnisse erwarten lässt.

Die Funktionsweise eines Focused Crawlers unterscheidet sich dabei signifikant von der herkömmlicher Suchmaschinen. Während letztere die Antwort auf eine Suchanfrage innerhalb von (Milli-)Sekunden liefern, kann ein Focused Crawler durchaus mehrere Minuten oder gar Stunden aktiv sein. In Bezug auf schnelle Ergebnisrückmeldung ist dies ein klarer Nachteil für den Focused Crawler.

Ein weiterer Nachteil ist die hohe Lokalität des Focused Crawlers gegenüber der globalen Sicht herkömmlicher Suchmaschinen. Ein Focused Crawler kann zur Beantwortung der Suchanfrage nur auf einen äußerst kleinen Prozentsatz aller existierender Webressourcen zurückgreifen, da er aufgrund limitierter Ressourcen nicht über die „Reichweite“ einer großen Suchmaschine verfügt. Ein hoher Recall ist also nur dann zu erwarten, wenn die gesuchten Dokumente tatsächlich mehr oder weniger stark miteinander verlinkt sind und der Zugang zu einer solchen Dokumentensammlung vom Focused Crawler schnell gefunden werden kann.

Der klare Vorteil dieses Ansatzes liegt jedoch darin, dass eine ungleich präzisere Antwort auf die Suchanfrage möglich ist. Durch die Möglichkeit, eigene kontextabhängige Komponenten zur Analyse der besuchten Webressourcen einzusetzen, lässt sich vorhandenes Wissen über das Themengebiet, Dokumentenstrukturen, die verwendete Sprache o. ä. zur Steigerung der Ergebnisqualität nutzen. Zudem kann dem Benutzer die Möglichkeit gegeben werden, auf den Verlauf der Suche einzuwirken und ggf. korrigierend einzugreifen, indem die Anfrage modifiziert wird oder offensichtlich irrelevante Suchzweige abgeschnitten werden. Der Nachteil langlaufender Suchen kann somit also zumindest teilweise entkräftet werden.

Die beiden Aspekte, hoher Recall und ein durch detaillierte Analyse bedingte qualitativ hochwertige Ergebnismenge, stellen gegenläufige Optimierungsziele dar. Während herkömmliche Suchmaschinen wie oben erläutert ihre Stärke im Bereich des hohen Recalls haben, bedingt durch ihr umfassendes Wissen über große Teile des WWW, so haben Focused Crawler ein entsprechend hohes Potenzial im Bereich der detaillierten Analyse.

Eine Kompromisslösung zur Maximierung beider Optimierungsziele stellt die Kombination der beiden Ansätze dar. In Kapitel 4.2 wird dazu ein System vorgestellt, das primär auf einen Focused Crawler baut. Um die Schwächen dieses Focused Crawlers zu kompensieren, insbesondere die der hohen Lokalität, werden herkömmliche Suchmaschinen integriert und somit deren Wissen über Struktur und Inhalt eines weitaus größeren Teils des WWW nutzbar gemacht.

3.1.2 Indexierung

Um Suchanfragen schnell und effizient beantworten zu können, bauen Suchmaschinen umfangreiche Indexstrukturen auf, mittels derer eine Abbildung von Suchanfragen auf Ergebnisdokumente erfolgt. Das grundlegende Prinzip, nach dem solche Indexstrukturen aufgebaut sind, ist meist das der „Invertierten Liste“ bzw. des „Invertierten Index“ (engl. „Inverted Index“) [BYRN99].

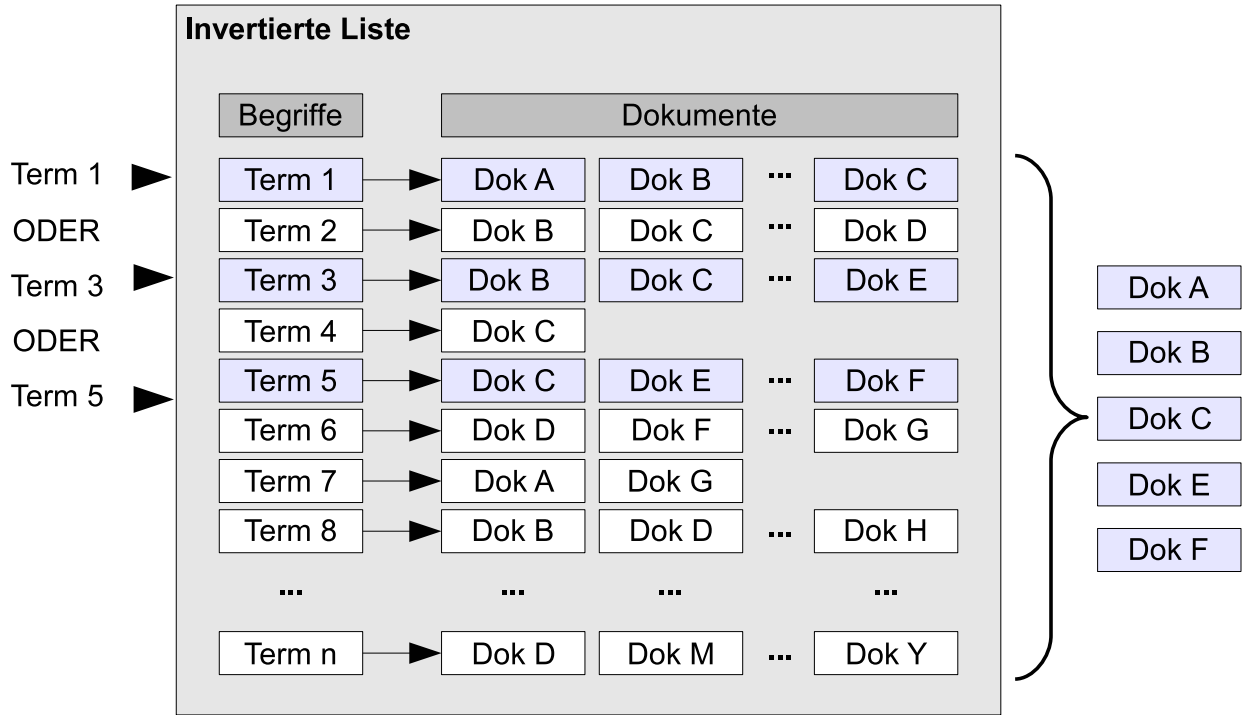

Abbildung 3.1: Prinzip der *Invertierten Liste* und ihre Anwendung

Abbildung 3.1 zeigt das Prinzip dieser Indexstruktur. Sei $\mathbb{D} = \{d_1, \ldots, d_n\}$ die Menge aller bekannten Dokumente d_i und ferner $T^{\mathbb{D}} = \{t_1, \ldots, t_{|T^{\mathbb{D}}|}\}$ die Menge aller in $\mathbb{D}$ suchbaren Begriffe t_j. Dann implementiert der Index eine Abbildung $I : T^{\mathbb{D}} \rightarrow \mathcal{P}(\mathbb{D}), t \mapsto I(t) := \{d \in \mathbb{D} | tf(t,d) > 0\}$ vom Begriffsraum in die Potenzmenge der Dokumente. Diese Abbildung ordnet also jedem Begriff t die Teilmenge der Dokumente zu, die t enthalten ($tf(t,d)$ ist die Anzahl der Nennungen des Begriffs t in Dokument d).

Um nun eine schlüsselwortbasierte Suchanfrage zu beantworten, wird für jeden der in der Anfrage enthaltenen Terme ein Index-Lookup durchgeführt (vgl. Ab-

bildung 3.1). Dieser Lookup liefert als Ergebnis die Teilmengen der Dokumente, die auf jeweils einen der Suchbegriffe passen. Für die korrekte Beantwortung der Anfrage, müssen diese Teilmengen im Anschluss entsprechend der in der Anfrage verwendeten Operatoren verknüpft werden. Im Beispiel in Abbildung 3.1 bewirkt die Disjunktion der Suchterme eine Vereinigung der Ergebnisteilmengen. Bei anderen Operatoren wie der Konjunktion oder der Negation würden entsprechende Operationen auf den Ergebnisteilmengen durchgeführt.

Offensichtlich kann ein solcher Index in Systemen wie dem o. g. Focused Crawler nicht oder nur eingeschränkt genutzt werden, da die Indexerstellung ein vorhergehendes Crawling entsprechender Teile des WWW erfordert. Für Suchmaschinen die ein solches Crawling jedoch durchführen können, bieten diese Indexstrukturen eine effiziente Möglichkeit, relevante Dokumente schnell aus einer großen Datenbasis zu selektieren und dem Benutzer zu präsentieren.

Für den Einsatz in Eigenentwicklungen stehen diverse Programm-Bibliotheken zur Verfügung. Die derzeit bekannteste unter diesen Bibliotheken ist die Suchmaschinen-Bibliothek „Lucene“ [HG04] der Apache Foundation. Sie bietet neben der grundlegenden Funktionalität der Generierung, Verwaltung und Nutzung eines Invertierten Index zusätzliche Funktionalitäten für die Analyse von Dokumenten und das Ranking von Suchergebnissen.

3.1.3 Ähnlichkeitsmaße für Texte und Suchanfragen

Neben der durch ein ausgedehntes Crawling erreichbaren Abdeckung des WWW und einem entsprechend umfangreichen Index definieren sich Suchmaschinen in erster Linie über ihre Fähigkeit, dem Benutzer die gefundenen Ergebnisse in einer für ihn möglichst Gewinn bringenden Reihenfolge zu präsentieren. Im Regelfall bedeutet dies, die am besten zur Suchanfrage passenden Dokumente zuerst und die weniger passenden Dokumente auf den hinteren Plätzen einzuordnen, so dass der Benutzer auf den ersten Blick die relevanten Dokumente erfassen kann. Das Aufstellen dieser Reihenfolge wird als „Ranking“ bezeichnet

und basiert meist auf den Ähnlichkeitswerten jedes Paares von Anfrage q und Ergebnisdokument d_i: $Sim(q, d_i)$.

Nachfolgend werden einige der in diesem Zusammenhang eingesetzten Ähnlichkeitsmaße vorgestellt. Dabei wird im Wesentlichen auf Ähnlichkeitsmaße zwischen Texten abgezielt. Auf eine explizite Betrachtung von Ähnlichkeitsmaßen zwischen Texten und Schlüsselwort-Suchanfragen wird verzichtet. Faktisch stellt dies jedoch keine Einschränkung dar, da für grundlegende Betrachtungen auch Schlüsselwortanfragen als – wenn auch kurze – Texte angesehen werden können [BYRN99]. In der Praxis wird eine derartige Sicht auf Schlüsselwortanfragen ebenfalls angewandt und wo nötig mit spezieller Funktionalität erweitert.

Mit den Jahren wurde eine Vielzahl solcher Ähnlichkeitsmaße entwickelt. Zunächst standen dabei syntaxbasierte Abstandsmaße im Vordergrund, wie bspw. auf dem Hamming- oder Levenshtein-Abstand [Ham50] [Lev66] bauende Kennzahlen. Sie bilden ein Maß für die Anzahl der Stellen, an denen sich zwei Texte unterscheiden. In der weiteren Entwicklung kamen neben der reinen Syntax zunehmend auch semantische Aspekte der Ähnlichkeit ins Spiel: durch eine definierte Semantik der Syntax von Schlüsselwortanfragen oder durch weiter gehende, zunächst statistische Analysen der betreffenden Texte. Im Folgenden soll eine Auswahl der für diese Arbeit relevanten Ansätze präsentiert und diskutiert werden.

3.1.3.1 Boolesches Modell

Suchmaschinen durchforsten das WWW und analysieren dabei jede gefundene Ressource, um sie dem Benutzer über einen Index zugreifbar zu machen. Den einfachsten Ansatz, diesen Index zu nutzen, stellt das Boolesche Modell dar [BYRN99]. Im Booleschen Modell ist eine Algebra definiert, die die Formulierung von Suchanfragen auf Basis von Schlüsselworten und logischen Operatoren erlaubt. Die unterstützten Operatoren sind für gewöhnlich `UND`, `ODER` sowie `NICHT`.

Auf Basis dieser Algebra lassen sich Suchanfragen definieren, die die in der booleschen Suchanfrage spezifizierten Schlüsselworte exakt enthalten.

Diese Art von Anfragen lassen sich mit Indexstrukturen wie der „Invertierten Liste“ schnell und effizient beantworten (vgl. Kapitel 3.1.2). Allerdings hat dieses Modell auch gravierende Schwächen [BC87]:

1. Das Verfahren fordert eine exakte Erfüllung des spezifizierten Prädikats. Viele relevante Ressourcen werden daher nicht gefunden, weil das Prädikat oftmals nicht vollständig erfüllt ist.
2. Werden große Mengen an Treffern zurückgemeldet, so ist jeder Treffer gleichwertig. Eine Ordnung der Treffer nach Relevanz ist nicht möglich, da das Modell kein ordinales Maß für die Relevanz besitzt sondern lediglich eine Klassifizierung in „Prädikat erfüllt“/„Prädikat nicht erfüllt“ vornimmt.
3. Anfrage und Ressource müssen im gleichen Vokabular vorliegen. Das bedeutet nicht nur, dass die gleiche Sprache gefordert ist, sondern auch, dass bspw. Synonyme nicht berücksichtigt werden.

Für einfache Suchanfragen liefert dieser Ansatz oftmals hinreichend gute Ergebnisse. Im Falle der komplexen Suchszenarien, die dieser Arbeit zugrunde liegen, werden jedoch erweiterte Konzepte benötigt, die neben der Syntax auch die Semantik der zu untersuchenden Texte erfassen und ferner bessere Möglichkeiten des Rankings von Suchergebnissen bieten.

3.1.3.2 Vektorraummodell

Um o. g. Schwächen zu kompensieren, wurden das *Vektorraummodell* (engl. *Vector Space Model, VSM*) entwickelt [SM86] [MRS08]. Diesem Modell liegt die Idee zugrunde, Texte bzw. Suchanfragen in einen mathematischen Raum abzubilden, für den ein Ähnlichkeitsmaß existiert, das einfach berechnet werden kann. Das Konzept der Ähnlichkeit ist dabei schwächer als die im Booleschen Modell geforderte Exaktheit und zielt somit auf die o. g. Punkte 1 und 3 ab. Auch das in Punkt 2 geforderte Konzept des Rankings kann durch ein solches Ähnlichkeitsmaß realisiert werden.

Als mathematischer Raum mit dem geforderten leicht berechenbaren und mindestens ordinal skalierten Ähnlichkeitsmaß bietet sich der Euklidische Raum an. Er beschreibt einen multidimensionalen Vektorraum, auf dem das Skalarprodukt definiert ist. Zwei Ideen liegen nun der Nutzung dieses Raumes zugrunde: i) die Abbildung eines Textes oder einer Suchanfrage auf einen Vektor in diesem Raum und ii) die Nutzung des Skalarprodukts zur Berechnung des Winkels zwischen solchen Vektoren. Dieser Winkel wird dann als Maß für die Ähnlichkeit der zugrunde liegenden Texte verwendet.

Sei $\mathbb{D} = \{d_1, \ldots, d_n\}$ die Menge aller bekannten Dokumente d_i und ferner $T^{\mathbb{D}} = \{t_1, \ldots, t_m\}$ die Menge aller in $\mathbb{D}$ suchbaren Begriffe t_j. Dann wird ein Dokument $d \in \mathbb{D}$ repräsentiert durch einen $|T^{\mathbb{D}}|$-dimensionalen Vektor $\vec{d}$ im Vektorraum $\mathbb{R}^{|T^{\mathbb{D}}|}$. Jeder Komponente a_i von $\vec{d} = (a_1, \ldots, a_{|T^{\mathbb{D}}|})^T$ wird nun das Gewicht zugewiesen das der Begriff, den sie repräsentiert, im zugrunde liegenden Dokument besitzt. Begriffe, die im Text nicht vorkommen, werden somit auf den Wert 0 abgebildet, alle anderen auf Werte größer 0.

Die Ähnlichkeit zweier Dokumente $d_1, d_2 \in \mathbb{D}$ kann nun über den Winkel definiert werden, in dem die sie repräsentierenden Vektoren $\vec{a}, \vec{b}$ zueinander stehen. Sie berechnet sich wie folgt über die Cosinus-Ähnlichkeit:

$$sim(d_1, d_2) := \cos \measuredangle(\vec{a}, \vec{b}) = \frac{\vec{a} \bullet \vec{b}}{|\vec{a}| \cdot |\vec{b}|} = \frac{\sum_{i=1}^{|T^{\mathbb{D}}|} a_i \cdot b_i}{\sqrt{\sum_{i=1}^{|T^{\mathbb{D}}|} a_i^2} \cdot \sqrt{\sum_{i=1}^{|T^{\mathbb{D}}|} b_i^2}} \tag{3.1}$$

Je geringer nun die Unterschiede zweier solcher Vektoren in jeder Komponente sind, desto kleiner ist der Winkel, den sie aufspannen und desto höher wird die Ähnlichkeit der zugrunde liegenden Dokumente bewertet.

Zentral für dieses Modell ist nun die Gewichtung der Begriffe und damit der skalaren Komponenten a_i der Vektoren. Im einfachsten Fall wird bei der Gewichtung lediglich gezählt, wie häufig ein Begriff im Text vorkommt. Alleine diese Häufigkeit für die Gewichtung zu verwenden, birgt allerdings die Gefahr, dass Begriffe über- oder unterbewertet werden und diese Bewertung die Relevanz

des Begriffs für den Text somit nicht korrekt widerspiegelt. Als Extrembeispiel seien so genannte Stop-Wörter wie *und, an, auf, bei, da* etc. angeführt, die in Texten zwar häufig vorkommen, jedoch wenig zur Bedeutung des Textes beitragen. Deren Einfluss auf die Ähnlichkeitsberechnung gilt es zu minimieren und im Gegenzug den Einfluss von kennzeichnenden Begriffen zu stärken. Erreicht wird dies bspw. durch das $tf \cdot idf$-Konzept [SM86].

Beim $tf \cdot idf$-Maß wird die oben diskutierte Häufigkeit $tf(t,d)$ eines Begriffs $t \in T^{\mathbb{D}}$ im Dokument $d \in \mathbb{D}$ durch einen Wert $df(t,\mathbb{D})$ dividiert, der ein Maß dafür darstellt, wie häufig der Begriff t in $\mathbb{D}$ und damit der betrachteten Sprache bzw. im betrachteten Textkorpus vorkommt. Damit ergibt sich die Komponente a_i des Dokumentenvektors $\vec{d}$ zum Gewicht $W(t_i,d)$ des Begriffs t_i für Dokument d wie folgt:

$$a_i := W(t_i,d) := \frac{tf(t_i,d)}{df(t_i,\mathbb{D})} \tag{3.2}$$

Der Wert $df(t,\mathbb{D})$ wird in der Praxis auf unterschiedliche Arten berechnet. Allen gemein ist jedoch, dass sie als Grundlage die Anzahl der Dokumente verwenden, die den Begriff enthalten:

$$df_{konzeptionell}(t,\mathbb{D}) := \left|\{d \in \mathbb{D} | tf(t,d) > 0\}\right| \tag{3.3}$$

Meist wird diese Häufigkeit logarithmiert, um den Einfluss von Extremwerten zu reduzieren. Verschiedene weitere Verfahren kommen für die Feinabstimmung sowie zur Kompensation von Sonderfällen zum Einsatz. Für eine Vertiefung sei an dieser Stelle auf die Literatur verwiesen [Fer03] [BYRN99] [GF04].

Das $tf \cdot idf$-Konzept kann nun sowohl auf den Vergleich von Dokumenten untereinander angewandt werden, als auch auf den Vergleich einer schlüsselwortbasierten Suchanfrage mit Dokumenten. Dabei wird die Anfrage ebenfalls

als Dokument interpretiert, bei dem allerdings i. d. R. nur wenige Komponenten a_i des Dokumentenvektors $\vec{d}$ einen von 0 verschiedenen Wert haben. Das an [GF04] angelehnte folgende Beispiel illustriert die Verwendung von $tf \cdot idf$.

Gegeben seien die Suchanfrage q sowie die Menge der suchbaren Dokumente $\mathbb{D} = \{d_1, d_2, d_3\}$:

q : a silver gold truck
d_1 : a shipment of gold was damaged in a fire
d_2 : delivery of silver arrived in a silver truck
d_3 : shipment of gold arrived in a truck

Die tf- und df-Werte der Dokumente d_1, d_2, d_3 und der Anfrage q ergeben sich damit wie in Tabelle 3.1 dargestellt:

Tabelle 3.1: Beispielhafte tf- und df-Werte

t	$tf(t,q)$	$tf(t,d_1)$	$tf(t,d_2)$	$tf(t,d_3)$	$df(t,\mathbb{D})$
a	1	2	1	1	3
arrived	0	0	1	1	2
damaged	0	1	0	0	1
delivery	0	0	1	0	1
fire	0	1	0	0	1
gold	1	1	0	1	2
in	0	1	1	1	3
of	0	1	1	1	3
shipment	0	1	0	1	2
silver	1	0	2	0	1
truck	1	0	1	1	2
was	0	1	0	0	1

Mittels der Formel 3.1 lassen sich nun die Ähnlichkeiten zwischen jeder Paarung (q, d_i) berechnen:

$$sim(q,d_1) = \frac{2/9+1/4}{\sqrt{1/9+1/4+1+1/4}\cdot\sqrt{4/9+1+1+1/4+1/9+1/9+1/4+1}} \approx 0,18$$

$$sim(q,d_2) = \frac{1/9+2+1/4}{\sqrt{1/9+1/4+1+1/4}\cdot\sqrt{1/9+1/4+1+1/9+1/9+4+1/4}} \approx 0,77$$

$$sim(q,d_3) = \frac{1/9+1/4+1/4}{\sqrt{1/9+1/4+1+1/4}\cdot\sqrt{1/9+1/4+1/4+1/9+1/9+1/4+1/4}} \approx 0,42$$

Das Ranking der Dokumente zur Suchanfrage ergibt sich damit zu d_2 vor d_3 vor d_1. An zwei Aspekten lassen sich hier die Stärken des $tf \cdot idf$-Konzepts erkennen:

- Der Begriff „silver“ kommt lediglich in Dokument d_2 vor, wird dort aber zweimal genannt. Es ist also davon auszugehen, dass „silver“ einen für d_2 kennzeichnenden Begriff darstellt. Da er auch in der Suchanfrage vorkommt, erhält d_2 eine hohe Wertung.
- Der Begriff „a“ wird zwar in Dokument d_1 ebenfalls zweimal genannt, da er jedoch in jedem Dokument vorkommt, wird ihm keine ausgeprägte Kennzeichnungskraft zugeordnet. Für die Wertung von d_1 fällt er damit kaum ins Gewicht.

Das Vektorraummodell ist die Grundlage vieler heute eingesetzter Systeme und prägt auch diese Arbeit. Insbesondere das hinter $tf \cdot idf$ stehende Konzept findet Anwendung, sowohl im hier beschriebenen Umfeld, als auch in modifizierter Form zur Bewertung der Ähnlichkeit von Hyperlink-Graphen (vgl. Kapitel 6).

3.1.3.3 Latent Semantic Indexing/Analysis

Ansätze, die auf dem oben diskutierten Vektorraummodell basieren, quantifizieren syntaktische Überlappung und damit die syntaktische Ähnlichkeit zweier Dokumente. Zwar bildet das $tf \cdot idf$-Konzept ansatzweise semantische Aspekte ab. Dennoch ist die Aussagekraft dieses Konzepts bezüglich Semantik äußerst limitiert. Wie oben angeführt, können sich zwei Dokumente mit dem selben Thema befassen oder sogar den selben Sachverhalt wiedergeben, auch wenn sie ein unterschiedliches Vokabular zur Beschreibung verwenden. Mit diesem Problem beschäftigt sich u. a. das Konzept der *Latent Semantic Analysis* (LSA) bzw. *Latent Semantic Indexing* (LSI) [DDL+90]. Bei LSA/LSI werden mehrere unterschiedliche Begriffe, die in einer Textsammlung synonym oder zumindest in einem ähnlichen Kontext verwendet werden, auf ein einziges so genanntes „Konzept“ projiziert. Die Idee hinter dieser Projektion ist die Reduzierung der Dimensionalität der Term-Dokument-Matrix einer Dokumentensammlung (vgl.

Abbildung 3.1) um i) die Berechnung effizienter durchführen zu können und ii) um sich auf semantische Konzepte anstatt auf syntaktische Einheiten konzentrieren zu können. Dies wird erreicht, indem zur Term-Dokument-Matrix eine niederrangige Approximation berechnet wird, die nicht mehr die einzelnen Terme der Dokumente beschreibt, sondern die von den Dokumenten behandelten „Konzepte“. Diese Dimensionalitätsreduktion wird durch eine Singulärwertzerlegung erreicht, bei der die n kleinsten Eigenwerte der Term-Dokument-Matrix ignoriert werden. Das Ignorieren dieser „irrelevanten“ Eigenwerte setzt die Idee der Konzeptionalisierung der Term-Dokument-Matrix um, weil damit irrelevante Begriffe herausfallen und semantisch ähnliche Begriffe zu einem Konzept zusammen gefasst werden. Eine Suchanfrage wird dann ebenfalls in den Konzept-Raum transformiert und anschließend analog zum Vektorraummodell die Ähnlichkeiten zu den transformierten Dokument-Vektoren berechnet. Für eine Diskussion über die Vor- und Nachteile von LSA/LSI sei auf die Literatur verwiesen. In [GF04] findet sich bspw. eine Übersicht über relevante Quellen.

Im Kontext der vorliegenden Arbeit ist der wesentliche Schwachpunkt dieses Ansatzes, dass zumindest eines von zwei zu vergleichenden Dokumenten Teil einer großen Dokumentensammlung sein muss, für die die aufwändige Eigenwertzerlegung im Vorfeld berechnet wurde. Es ist also nicht möglich, zwei beliebige Dokumente auf ihre Ähnlichkeit hin zu prüfen, sofern diese dem System bisher unbekannt sind. In vielen Bereichen, wie bspw. dem hier behandelten „Focused Crawling“, kann diese Vorverarbeitung aber nicht durchgeführt werden, weil es in der Natur des Problems liegt, dass a priori unbekannte Dokumente verarbeitet werden müssen. Dennoch bildet die der LSA/LSI zugrunde liegende Idee, einzelne Terme auf abstraktere Konzepte zu projizieren, die Grundlage für einen im Rahmen dieser Arbeit vorgeschlagenen Ansatz zur Berechnung von Dokumentenähnlichkeit (vgl. Kapitel 6).

3.1.3.4 WordNet, Thesauren, Wikipedia

Verschiedene neuere Ansätze [BP03] oder [JS03] bauen auf den Begriffs- und Textsammlungen von Projekten wie *WordNet* [Fel98] oder *Roget's Thesaurus* [Rog52] auf. Dabei machen sie sich die diesen Daten zugrunde liegenden Informationen über Zusammenhänge zwischen diesen Begriffen und Texten zu Nutzen und leiten daraus Aussagen über die Ähnlichkeit von Wortpaarungen oder vollständigen Texten ab. Derartige Datenbanken bieten wertvolle Informationen über semantische Beziehungen zwischen Worten oder Textphrasen und lassen sich somit Gewinn bringend zur Berechnung von Dokumentenähnlichkeit einsetzen. Allerdings ist ein hoher Aufwand von Nöten, um solche Datenbanken zu erstellen und zu pflegen. Zwar sind die Informationen, die aus diesen Datenbanken abgeleitet werden können, von hohem Wert, da bspw. die Semantik aller Beziehungen zwischen Begriffen oder Phrasen klar definiert ist. Jedoch existieren verhältnismäßig wenige Typen solcher Beziehungen (bspw. die Beziehungen *Is-A-Synonym*, *Is-An-Instance-Of*, *Is-Part-Of* etc.). Die weitaus meisten Beziehungstypen, die eine wie auch immer geartete Ähnlichkeit von Begriffen oder Phrasen anzeigen, sind aber nicht explizit modelliert, da dies i) nicht Sinn dieser Projekte ist und ii) aufgrund des Umfangs und nicht zuletzt bedingt durch Mehrdeutigkeit der Sprache nicht zu leisten wäre.

Andere Arbeiten setzen auf die freie Online-Enzyklopädie Wikipedia als zugrunde liegende Datenbank. Der wesentliche Unterschied zwischen Wikipedia und Datenbanken wie WordNet besteht darin, dass Wikipedia untypisierte Beziehungen zwischen den Entitäten in Form von Hyperlinks verwaltet. Zwar existieren Ansätze, genau diese Nicht-Typisierung der Beziehungen zu überwinden und statt dessen typisierte Beziehungen einzuführen, um eine „Semantische Wikipedia" zu erhalten [VKV+06]. Allerdings ist nach wie vor der Großteil aller Wikipedia-Daten untypisiert verknüpft, was sich aus vielerlei Gründen auch auf absehbare Zeit nicht ändern wird. Da auch die vorliegende Arbeit stark auf der Nutzung von Wikipedia-Informationen baut, sollen im Folgenden zwei Ansätze betrachtet werden, die sich ebenfalls mit dieser Enzyklopädie befassen und eine entsprechende Nähe zur vorliegenden Arbeit vorweisen.

3.1.3.5 WikiRelate!

In [SP06] wurde der Algorithmus *WikiRelate!* entwickelt, der die „semantic relatedness“ zweier Begriffe berechnet. D. h. er quantifiziert, in wie weit zwei Begriffe etwas miteinander zu tun haben. *WikiRelate!* nutzt dabei zwei Metriken, um diese Beziehung zu berechnen: Zunächst wird die syntaktische Überlappung der mit den Begriffen direkt assoziierten Wikipedia-Artikel berechnet. Zudem wird die konzeptionelle Entfernung der Begriffe anhand des „information content“ [Res95] berechnet und ferner ein Pfadlängenmaß auf das Kategoriensystem der Wikipedia angewandt. Dabei werden die Wikipedia-Kategorien, die einem Artikel zugeordnet sind, als abstrakte Konzepte angesehen, die die Semantik des durch diesen Artikel repräsentierten Begriffs widerspiegeln.

Die Schwachpunkte dieses Konzepts bzw. in dessen Realisierung liegen darin, dass *WikiRelate!* lediglich die Ähnlichkeit von Begriffspaaren berechnen und keine kompletten Texte verarbeiten kann. Ferner müssen die auf Ähnlichkeit zu untersuchenden Begriffe durch Wikipedia-Artikel repräsentiert sein, was zwar für viele, aber bei weitem nicht alle Begriffe zutrifft. Zudem beschränkt sich *WikiRelate!* bei der Nutzung struktureller Informationen auf das hierarchische Kategoriensystem der Wikipedia und ignoriert die zwischen den Wikipedia-Artikeln existierenden Verknüpfungen.

3.1.3.6 Explicit Semantic Analysis (ESA)

Ein weiterer vielversprechender Ansatz auf Basis von Wikipedia ist die *Explicit Semantic Analysis* (ESA) [GM07][SC08]. Bei *ESA* wird für jeden zu vergleichenden Text ein gewichteter Vektor aus Wikipedia-Konzepten (=Wikipedia-Artikeln) aufgebaut. Dazu werden konventionelle Algorithmen zur Textklassifizierung eingesetzt, die die Relevanz eines jeden Wikipedia-Artikels für den jeweiligen Text ermitteln und damit die Einträge des Vektors bestimmen. Die Ähnlichkeit wird dann anhand des auf die resultierenden Vektoren angewandten Cosinus-Maßes ermittelt (vgl. Kapitel 3.1.3.2). Der wesentliche Schwachpunkt von *ESA* liegt darin, dass lediglich der textuelle Inhalt der Wikipedia-Datenbank

für die Berechnung herangezogen wird und strukturelle Informationen wie die Verknüpfung von Artikeln außen vor bleiben.

3.1.3.7 Co-Citation/Companion Link Analysis

Neben dem reinen in Hypertextdatenbanken enthaltenen Text, werden in anderen Ansätzen auch strukturelle Informationen genutzt, um die Ähnlichkeit von – in diesem Fall – Hypertexten zu berechnen. Ein prominentes Beispiel ist der in [DH99] beschriebene Ansatz. Dabei werden ein- und ausgehende Verbindungen (=Hyperlinks) zweier Hypertext-Ressourcen anhand der dort *co-citation* und *companion* genannten Algorithmen miteinander verglichen und somit die Dokumente in den Vergleich einbezogen, die die zu untersuchenden Dokumente im Netz umgeben. Der wesentliche Schwachpunkt dieses Ansatzes ist jedoch, dass die beiden zu vergleichenden Texte auch Hypertexte im selben Netz, bspw. dem WWW, sein müssen, da ansonsten keine Berechnung anhand gemeinsamer Verweise möglich ist. Insbesondere können somit keine beliebigen Dokumente aus unbekannten Quellen miteinander verglichen werden. Ein weiterer wesentlicher Schwachpunkt beim Einsatz im vorliegenden Szenario liegt in der Behandlung von Kopien eines Textes: wenn eine Kopie d' eines Dokuments d erzeugt und an einer beliebigen Stelle des Hypertextnetzes abgelegt wird, dann ist die Wahrscheinlichkeit sehr groß, dass für d und d' eine geringe Ähnlichkeit errechnet wird, da deren jeweilige Hypertextumgebung sich im Allgemeinen unterscheiden wird, wenn nicht zusätzlich auch alle Verknüpfungen dupliziert werden. Damit würden aber zwei identische Texte eine geringe Ähnlichkeit aufweisen, was im Kontext dieser Arbeit keinen sinnvollen Ansatz darstellt.

3.1.3.8 Dokumentenclustering

Wie in Kapitel 3.2 näher ausgeführt, ist Dokumentenclustering eine Anwendung für die Berechnung von Dokumentenähnlichkeit, die auf der automatisierten Gruppierung gleichartiger Dokumente beruht. Es existieren verschiedene Ansätze, die teilweise auch auf externe Daten wie Wikipedia setzen. Bspw. werden

in [WHZ+07] Wikipedia-Kategorien als eine abstrakte Repräsentation von Dokumenten benutzt und auf Basis dieser Kategorien eine Cluster-Berechnung durchgeführt. Weitere Ausführungen zum diesem Thema finden sich in den folgenden Abschnitten 3.1.4 „Schnittstelle zum Benutzer“ und 3.2 „Textanalyse“.

3.1.4 Schnittstelle zum Benutzer

Das „User Interface“, stellt die Schnittstelle zwischen Suchsystem und Benutzer dar. Neben der reinen Funktionalität des Suchsystems ist das User Interface ein wesentliches Kriterium für die Benutzbarkeit eines solchen Systems und damit für die Akzeptanz, die es beim Benutzer erfährt. Im Umfeld der Websuchen haben sich einfache Schnittstellen etabliert, die ohne großen Einarbeitungsaufwand erlernbar sind und eine für viele Szenarien ausreichende Funktionalität zur Verfügung stellen. Sie bieten einfache Eingabefelder für Suchbegriffe, leiten diese an das Suchsystem weiter und stellen dessen Antworten in Form von Ergebnislisten dar. Diese Ergebnislisten zeigen meist die URL der gefundenen Ressource und eine knappe Zusammenfassung der zu den Suchbegriffen passenden Textstellen, so genannte „snippets“.

Für einfache Recherchen, in denen es primär um das Auffinden relevanter Dokumente geht, ist diese Form der Interaktion zwischen Benutzer und System meist ausreichend. Komplexere Suchen oder Anforderungen in Richtung der Weiterverarbeitung von Suchergebnissen, der Identifizierung von Querbezügen oder der (semi-)automatischen Informationsextraktion erfordern jedoch weiter gehende Ansätze.

Bereits seit den Anfängen des Information Retrieval machen Industrie und Forschung sich Gedanken über die Schnittstellen zwischen Benutzern und Suchsystemen. So wurden eine Vielzahl von grafischen Eingabe- und Analysesystemen vorgeschlagen und teilweise auch erfolgreich eingesetzt. In [BYRN99] wird hierzu eine umfassende Übersicht präsentiert.

In den letzten Jahren haben einige dieser Konzepte auch in allgemein verfügbare Websuchmaschinen Einzug gefunden. Bspw. bietet die Suchmaschine *Vivisimo*

[Viv] die Möglichkeit, Suchergebnisse nach Themenbereichen geclustert anzuzeigen. Diese Themenbereiche sind dabei nicht statisch vordefiniert, sondern ergeben sich aus den Treffer-Dokumenten selbst, indem Dokumente mit hoher Ähnlichkeit zu einem Cluster zusammen gefasst werden und Dokumente mit geringer Ähnlichkeit in getrennten Clustern eingeordnet werden. Dieses Prinzip kann der Benutzer wiederum auf jeden einzelnen Cluster anwenden und erhält somit die Möglichkeit, sich schnell auf diejenigen Unterthemen zu fokussieren, die ihn interessieren. Erleichtert wird dies durch eine Verschlagwortung der einzelnen Cluster, anhand derer der Benutzer sich orientieren kann.

Andere Suchmaschinen, wie bspw. *kartoo* [Kar] legen ihr Augenmerk auf die Darstellung des netztopologischen Umfelds der bei der Suche gefundenen Dokumente. Auch hier kommt wieder die Idee ins Spiel, die die Entwicklung des Konzepts der Focused Crawler motivierte: thematisch zusammenhängende Dokumente sind auch häufig über Verweise direkt miteinander verbunden. Durch die Darstellung der Netztopologie um die Suchergebnisse herum, können solche Themeninseln visualisiert und dem Benutzer zugänglich gemacht werden.

Wieder andere Suchmaschinen, die auf ein bestimmtes Thema spezialisiert sind, können an der Benutzerschnittstelle ebenfalls gezielte Unterstützung liefern. Beispielsweise existieren spezielle Patentsuchmaschinen, mittels derer Patentdatenbanken nach bestimmten Kriterien durchsucht werden können. Im Rahmen des EU-Projektes *PatExpert* wurde eine solche Software entwickelt, die u. a. verschiedene Sichten auf Suchanfragen und Ergebnismengen bietet und so den Analyseprozess unterstützt [KB11].

Auffallend ist, dass o. g. Suchmaschinen derzeit noch ein Nischendasein führen. Zwar bieten sie in Bezug auf die Schnittstelle Funktionalität, die weit über die der großen Suchmaschinen hinausgeht, dennoch haben sie in der Praxis noch keine große Verbreitung erfahren. Dies hat mehrere Gründe: zum einen ist es schlicht die Macht der Gewohnheit, die den Benutzer stets eine ihm bereits bekannte Suchmaschine wieder benutzen lässt. Eine neue Suchmaschine mit unbekannter Benutzer-Schnittstelle erfordert einen gewissen Einarbeitungsaufwand, den viele Benutzer nicht zu leisten bereit sind. Gewichtiger ist jedoch,

dass die meisten Webrecherchen einfacher Natur sind und weiter gehende Funktionalität überhaupt nicht benötigen. Analysen von Anfrageprotokollen großer Suchmaschinen haben gezeigt, dass die Mehrzahl aller Suchanfragen aus ein bis drei Schlüsselworten besteht, die lediglich implizit mittels des für die jeweilige Suchmaschine gültigen Standardoperators (*UND/ODER*) verknüpft sind. Es erfolgt keine explizite Verknüpfung mittels weiterer Operatoren wie *NICHT*, ebenso wenig werden suchmaschinenspezifische Funktionen wie die Beschränkung auf bestimmte Domains o. ä. in großem Umfang eingesetzt [SMHM99][JSP05][JS06].

Dennoch existieren auch andere Typen von Webrecherchen. Im Szenario, das diese Arbeit motiviert, sind kleine und mittelständische Unternehmen u. a. im WWW auf der Suche nach Informationen zu neuen Technologien, wo diese eingesetzt werden, wer damit Erfahrung hat, wie diese Technologien mit anderen Technologien zusammen spielen etc. Ihr Reifegrad und das Marktpotenzial werden analysiert und vieles mehr. Um diese Art von Recherchen durchzuführen, werden sehr wohl weiter gehende Konzepte zur Interaktion mit dem Benutzer benötigt und auch eingesetzt. Die o. g. Suchmaschinen bieten hierbei allerdings nur geringen Mehrwert gegenüber herkömmlichen Suchmaschinen, da sie einen gravierenden Schwachpunkt besitzen: Zwar stellt jede für sich betrachtet eine gelungene Umsetzung des jeweils fokussierten Konzeptes dar, diese verschiedenen Konzepte stehen jedoch nicht integriert zur Verfügung. Insbesondere für komplexe Suchen wäre die Integration verschiedener Konzepte und Herangehensweisen sowie die durchgängige Nutzbarkeit von (Zwischen-) Ergebnissen über diese Systemgrenzen hinweg ein erstrebenswertes Ziel. In Kapitel 4 wird daher eine integrierte Suchplattform vorgeschlagen und deren wesentliche Konzepte diskutiert.

3.1.5 Marktübersicht

Als Abschluss des Grundlagen-Kapitels „Web-Suchmaschinen“ soll an dieser Stelle eine knappe Übersicht über aktuelle Suchmaschinen und deren Charakteristik gegeben werden.

Generell muss unterschieden werden zwischen den großen Standardsuchmaschinen und kleineren Spezialsuchmaschinen, die sich auf bestimmte Themen oder Suchkonzepte spezialisiert haben. Als „standard“ werden hier bspw. die Websuchmaschinen *Yahoo*, *Google* oder *Bing* angesehen. Sie indexieren Dokumente des WWW und machen dem Benutzer den resultierenden Index über eine schlüsselwortbasierte Anfrageschnittstelle zugänglich. Neben der Möglichkeit, einzelne Suchbegriffe als Schlüsselworte zu definieren, können auch verschiedene Operatoren eingesetzt werden: `UND`, `ODER`, `NICHT` um nur Dokumente mit einer bestimmten Kombination von Suchbegriffen zu erhalten. Es können exakt zu suchende Phrasen oder Platzhalter für beliebige Begriffe oder Texte angegeben werden. Ferner wird die Möglichkeit angeboten, die Suche auf einzelne Domains zu beschränken oder Suchbegriffe lediglich in der URL von Dokumenten zu suchen anstatt im eigentlichen Dokumententext. Die Suchergebnisse werden dem Benutzer in einer listenartigen Darstellung präsentiert, wobei für jedes zurückgemeldete Dokument der Titel, die URL sowie ein knapper Ausschnitt des Dokumententextes angegeben wird. Zusätzlich überprüfen diese Suchmaschinen die vom Benutzer angegebenen Suchbegriffe und machen Korrekturvorschläge wenn ein vermeintlicher Schreibfehler entdeckt wurde. Neben der direkten Benutzerschnittstelle bieten die meisten dieser Standardsuchmaschinen auch eine auf den Zugriff durch Softwaresysteme optimierte Schnittstelle, ein sog. API (Application Programming Interface) an. Über diese APIs kann Web-Suchfunktionalität in andere Software integriert und so der Index der jeweiligen Suchmaschinen für anwendungsspezifische Zwecke genutzt werden.

Diese Standardsuchmaschinen werden ergänzt durch eine Vielzahl von Spezialsuchmaschinen. Die drei oben genannten Anbieter erlauben neben der reinen schlüsselwortbasierten Dokumentensuche auch eine spezielle Suche nach Bildern oder Videos. Es können aktuelle Nachrichtentexte durchsucht werden und teilweise wird auch die Möglichkeit angeboten, Onlineshops nach Produkten zu durchsuchen und die einzelnen Angebote miteinander zu vergleichen. Andere Suchmaschinen tun sich durch alternative Konzepte in der Benutzerführung hervor. So präsentieren *yippy.de* oder *vivisimo.de* die Suchergebnisse in Clus-

tern, also in mehreren thematisch voneinander getrennten Blöcken, was dem Benutzer die Analyse der Ergebnisse erleichtern und eine Konkretisierung der Suchanfrage vereinfachen soll. Wieder andere Suchmaschinen wie *yasni.de* oder *123people.de* stellen alle Informationen zusammen, die sie über eine Person bzw. einen Namen in WWW finden können. *frag-finn.de* oder *blinde-kuh.de* bieten Suchfunktionalität speziell für Kinder, wobei neben einer kindgerechten Benutzerschnittstelle auch die zurückgemeldeten Suchergebnisse kindgerecht sind oder sein sollen. Sogenannte *Semantische Suchmaschinen* wie *wolframalpha.com* oder *semager.de* versuchen, die Semantik einer Suchanfrage zu erfassen und liefern neben syntaktischen Treffern auch semantische Treffer als Ergebnis.

All diese Suchmaschinen bieten entweder generische Suchkonzepte oder sind spezialisiert auf bestimmte Themen oder Arten von Suchen und insbesondere die Liste der Spezialsuchmaschinen ließe sich beliebig fortsetzen. Ihnen allen ist jedoch gemein, dass jede Suchanfrage für sich behandelt wird. Es gibt keine Möglichkeit, so etwas wie einen Kontext zu spezifizieren, in dessen thematischem Rahmen weitere Suchen durchgeführt werden können. Insbesondere lassen sich Suchergebnisse der einen Suchmaschinen nicht in eine andere Suchmaschine übertragen, um die Ergebnismenge einer Suchmaschine mit der Technik einer anderen Suchmaschine weiter zu analysieren. Für die Durchführung von komplexen Webrecherchen sind dies die zwei wesentlichen Schwachpunkte, für die in den Kapiteln 4-6 Lösungskonzepte vorgeschlagen werden.

3.2 Textanalyse

Wie in den vorangegangen Abschnitten diskutiert, existieren verschiedene Ansätze, um das WWW nach Informationen zu durchsuchen. Sie unterscheiden sich insbesondere im Grad der Automatisierbarkeit, in der Komplexität der automatisch aus den Suchergebnissen ableitbaren Informationen und in der Reaktionszeit von Eingabe der Suchanfrage bis zur Ausgabe der Ergebnisse.

In den letzten Jahren hat sich nun im Bereich des Information Retrieval ein weiteres Konzept etabliert, die sog. *Textanalyse* (engl. *Textanalytics*) [Gri08]. Der Begriff *Textanalyse* zielt dabei nicht eindeutig auf eine bestimmte Technik oder eine konkrete Sammlung von Techniken, sondern beschreibt vielmehr ein Konzept. Dieses Konzept orientiert sich an dem von strukturierten Daten her bekannten *Data Mining* und versucht gleiche bzw. ähnliche Techniken auf unstrukturierte und semistrukturierte Daten wie die im WWW vornehmlich anzutreffenden natürlichsprachigen Texte anzuwenden.

Motiviert wird dieser Ansatz dadurch, dass im Umfeld der strukturierten Daten (ERP-/PPS-Systeme etc.) bereits seit langem bekannt ist, dass in der Datenbasis eines Unternehmens kostbare Informationen verborgen liegen und dass diese Informationen mittels verschiedener Analysetechniken aus der Datenbasis extrahiert werden können. Da nun aber, insbesondere im Zuge des stetig günstiger werdenden Speichers, mehr und mehr Daten in unstrukturierter Form abgelegt werden, d. h. in Form von Texten, Bildern, Video- und Audio-Aufzeichnungen, müssen die Konzepte des Data Mining an diese Datenformen und ihre Besonderheiten angepasst werden.

Bei strukturierten Daten ist per Definition explizites Wissen über deren Struktur vorhanden. Für das Mining auf unstrukturierten Daten hingegen muss dieses Strukturwissen offensichtlich aus der Datenbasis selbst oder aus zusätzlichen Quellen abgeleitet werden, da andernfalls keine Analysen über einer solchen Struktur durchgeführt werden können. Damit stellt die Textanalyse prinzipiell eine Vorstufe zum Mining dar. In der Praxis ist diese Grenze jedoch selten so eindeutig definiert und verschwimmt meist.

3.2.1 Techniken & Konzepte

Wie bereits angemerkt, wird unter dem Begriff „Textanalyse“ gemeinhin nicht eine spezielle Technik zur Analyse von natürlichsprachigen Texten verstanden, sondern vielmehr ein Konzept, um Informationen aus solchen Textdaten bzw. Textsammlungen zu extrahieren und nicht-triviale Verbindungen aufzudecken.

Um dieses Konzept umzusetzen, existieren verschiedene Techniken und Vorgehensweisen, die zum Teil im Bereich des Information Retrieval bereits seit längerem eingesetzt werden. Im Rahmen der Textanalyse werden diese kombiniert und ggf. erweitert. Der folgende Abschnitt fasst die Wichtigsten und Geläufigsten dieser Techniken zusammen.

Herkömmliche Suchmaschinen implementieren vornehmlich Techniken, die einen verhältnismäßig geringen Aufwand bei der Verarbeitung eines Dokuments erfordern. Andernfalls wäre die hohe Anzahl der von diesen Suchmaschinen indexierten Dokumente nicht erreichbar. Solche Methoden sind bspw. das Zurückführen eines Wortes auf seinen Wortstamm (engl. „stemming“). Dadurch wird die Suche nach einem Begriff erleichtert, weil nicht nur die vom Suchenden spezifizierte Form des Begriffs berücksichtigt wird, sondern auch andere durch Konjugation, Deklination etc. gebildete Ausprägungen des Begriffs gefunden werden. Daneben existiert die Technik der Entfernung von semantisch unbedeutenden Begriffen (engl. „Stopword removal“) aus dem Text. Dabei wird aus Gründen der höheren Verarbeitungsgeschwindigkeit und des geringeren Platzbedarfs für den Index der Umfang der zu indexierenden Texte reduziert, indem irrelevante Worte ignoriert werden. Zudem schwindet dadurch der Einfluss solcher irrelevanter Worte bei einem Vergleich der Ähnlichkeit von zwei Dokumenten, da dieser Vergleich lediglich auf den relevanten Begriffen durchgeführt wird und irrelevante Begriffe wie Binde- und Füllwörter ignoriert werden. Auch die Bestimmung der in einem Text verwendeten Sprache gehört zur Gruppe der grundlegenden Techniken. Sie wird bspw. für die Feinabstimmung von Analysetechniken für einen konkreten Text eingesetzt. All diese und weitere grundlegende Techniken kommen ebenfalls im Rahmen der Textanalyse zum Einsatz. Sie liefern zwar selbst noch keine höherwertigen Informationen, dienen aber als Enabler für die im Folgenden aufgeführten Ansätze zur Informations- und Wissensextraktion.

Abbildung 3.2 zeigt drei Stufen der Extraktion und Nutzung von Informationen, die sowohl in Komplexitätsbelangen, als auch in der zeitlichen Abfolge aufeinander aufbauen.

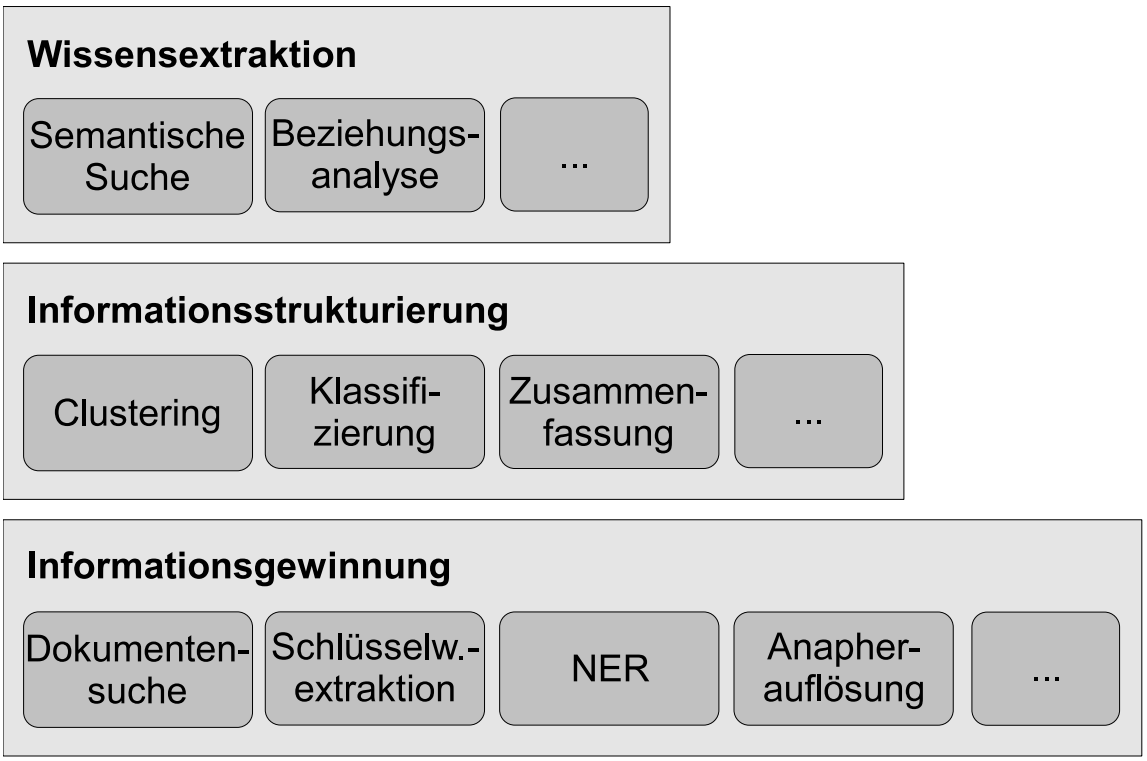

Abbildung 3.2: Stufen der Wissensextraktion

Um höherwertige Informationen aus Texten zu extrahieren, ist zunächst deren grundlegende Analyse zur „Informationsgewinnung“ nötig. Ziel einer solchen Analyse ist bspw. die Identifizierung von für den jeweiligen Text bedeutsamen Orten, Personen, Firmen oder deren Produkten. Das Wissen über den Bezug eines Textes zu oben genannten Objekten ist ein erster Schritt hin zum „Verständnis“ des Textes. Ansätze die dieses Ziel zu erreichen suchen, werden gemeinhin unter dem Begriff „NER“ (Named Entity Recognition/Extraction) zusammengefasst. Kapitel 7 widmet sich diesem Themengebiet im Detail, weshalb an dieser Stelle auf eine tiefer gehende Diskussion verzichtet wird.

Die Extraktion von Begriffen oder Begriffssequenzen, die einen Text kennzeichnen, liefert einen weiteren Beitrag zur Kategorisierung bzw. Grobeinordnung des Textes. Sie kann helfen, einen Überblick über den Text zu bekommen, ohne diesen Wort für Wort gelesen zu haben. Damit stellt sie eine Erleichterung für die manuelle Analyse großer Textmengen dar.

Einen weiter gehenden Ansatz stellt die Identifikation von Beziehungen zwi-

schen den o. g. Entitäten und den diese repräsentierenden Textstellen dar. Ein Spezialfall der Identifikation solcher Beziehungen ist bspw. die Auflösung von Anaphern. Dabei werden namentlich genannte Entitäten wie bspw. handelnde Personen mit an anderen Textstellen an ihrer statt genannten Pronomen identifiziert. Dies stellt einen weiteren Schritt für ein verbessertes Textverständnis dar und ermöglicht das Aufdecken von Verbindungen, die auf den ersten – rein syntaktischen – Blick hin nicht erkennbar wären.

Aufbauend auf den o. g. grundlegenden Techniken existieren im Kontext der Textanalyse weitere Ansätze, um eine (semi-) automatisierte Verarbeitung von natürlichsprachigen Texten zu vereinfachen. Abbildung 3.2 skizziert diese Ansätze in der mittleren Ebene „Informationsstrukturierung“.

Ein Mittel der Vereinfachung ist die Reduktion der Komplexität der zu betrachtenden Texte. Dies wird durch das Fokussieren auf relevante Textstellen erreicht. Bspw. können die Einleitung oder Zusammenfassung eines Textes automatisch extrahiert, oder bei weiter gehendem Textverständnis auch kennzeichnende, d. h. relevante Textbereiche aus dem gesamten Text herausgelöst werden. Dies ermöglicht dem Benutzer ein schnelleres Begreifen und Einordnen des Textes, da im Vergleich zum gesamten Text nur verhältnismäßig kleine Textblöcke manuell analysiert werden müssen.

Einen weiteren Ansatz stellt die Gruppierung von Texten nach bestimmten Kriterien dar. Sie erlaubt es dem Benutzer, anstelle einzelner Texte nur noch Klassen von Texten zu betrachten, die in den genannten Kriterien in einem geforderten Maß übereinstimmen. Hierzu existieren im Wesentlichen zwei Methoden, die Klassifikation und das so genannte Clustering. Bei der Klassifikation wird ein Text einer von mehreren, ggf. hierarchisch strukturierten und im Vorfeld definierten Themenklassen zugeordnet. Beim Clustering hingegen werden Texte so gruppiert, dass die Elemente einer Gruppe untereinander eine maximale Ähnlichkeit bezüglich der geforderten Kriterien besitzen und gleichzeitig die Ähnlichkeit zwischen Elementen unterschiedlicher Gruppen minimal ist.

Die obere Ebene „Wissensextraktion“ in Abbildung 3.2 repräsentiert die Nutzung der in den darunter liegenden Ebenen extrahierten Informationen. Bei der

Suche können damit sowohl verhältnismäßig einfache Suchprädikate verwendet werden, wie z. B. die von herkömmlichen Websuchmaschinen bekannten booleschen Verknüpfungen von Schlüsselworten. Daneben können aber auch komplexe Prädikate eingesetzt werden, die bspw. Beziehungen zwischen Worten, Satzteilen oder auch zwischen verschiedenen Texten betreffen. Damit kann eine Analyse nicht nur allein anhand syntaktischer Kriterien erfolgen; vielmehr rücken semantische Aspekte mehr und mehr in den Vordergrund.

3.2.2 Anwendungsfälle

Textanalyse ist, wie bereits angeführt, keine neue Technologie sondern definiert sich vielmehr über eine Sammlung von bereits bekannten Technologien und deren gemeinsame Nutzung. Daher ist dieser Bereich weniger im universitären Umfeld angesiedelt, sondern stark industriegetrieben und anwendungsgeprägt. Nachfolgend sollen einige Anwendungsfälle aufgezeigt werden, in deren Rahmen Textanalyse heute bereits erfolgreich eingesetzt wird.

Unter dem Stichwort „Sentiment Detection“, auf Deutsch am ehesten mit „Meinungsanalyse“ bzw. „Stimmungsanalyse“ zu übersetzen, wird ein Anwendungsgebiet der Textanalyse verstanden, in dem es darum geht, herauszufinden, was bestimmte Personen oder Personengruppen über ein Unternehmen und dessen Produkte denken. Quellen für solche Analysen sind in erster Linie Blogs und Onlineforen. Hier können sowohl positive als auch negative Aspekte des Untersuchungsgegenstandes extrahiert werden und allgemein positive bzw. negative Einstellungen der jeweiligen Personengruppe gegenüber diesem quantifiziert werden. Die besonderen Herausforderungen liegen hier in der Tatsache, dass die Sprache in solchen Foren/Blogs oftmals nicht strikt den Regeln der Grammatik folgt und somit ein echtes Textverständnis deutlich schwieriger zu erreichen ist als für grammatikalisch hochwertigere Texte.

Ein weiteres wichtiges Anwendungsgebiet ist das sog. „Quality Early Warning“. Ziel ist hier, anhand von Kundenbeschwerden, Reparaturberichten oder anderen textuellen Quellen existierende Schwachpunkte von Produkten oder

Dienstleistungen zu identifizieren, um entsprechende Gegenmaßnahmen ergreifen zu können. Der zweite Aspekt, das „Early“ (deutsch: frühzeitig), zielt darauf ab, solche Schwachpunkte bereits dann zu identifizieren, wenn sich ihre Auswirkungen erstmals in signifikantem Maße zeigen. Die Schwierigkeit hierbei liegt darin, verschiedene Formulierungen als die Beschreibung eines einzigen Problems zu identifizieren und ein entsprechendes Monitoring über die Häufigkeitsentwicklung zu unterhalten.

Neben den beiden oben Genannten existieren weitere Anwendungsfälle im Bereich der nachrichtendienstlichen Tätigkeit und Terrorismusabwehr, der Kriminalitätsbekämpfung, der automatisierten Abwicklung von Helpdesk-Calls, der Auswertung medizinischer Studien, der Marktanalyse und Patentanalyse sowie vieles mehr. Sie alle haben gemeinsam, dass große Mengen unstrukturierter Texte analysiert werden, daraus strukturierte Informationen extrahiert werden und diese ggf. mit weiteren Informationen aus anderen strukturierten oder unstrukturierten Datenquellen kombiniert werden, um größtmöglichen Nutzen aus der vorhandenen Datenbasis zu ziehen.

3.2.3 Marktübersicht

Wie oben angeführt, ist die „Textanalyse“ derzeit ein aufstrebendes Thema. Während im universitären Umfeld vornehmlich Grundlagen zu Themen wie Textverständnis, semantische Suche und ähnliche behandelt werden, so platzieren sich mehr und mehr Unternehmen mit dem Produkt „Textanalyse“ am Markt. Im Folgenden soll eine knappe Übersicht über partizipierende Unternehmen und deren Textanalyse-Produkte gegeben werden. Die Liste der präsentierten Software erhebt keinen Anspruch auf Vollständigkeit und repräsentiert lediglich einen Teil der Themengebiete, die mit Textanalyseprodukten untersucht werden können.

Als Unternehmen mit langjähriger Erfahrung im Bereich der Verarbeitung strukturieter Daten zielt auch IBM mehr und mehr in den Bereich der unstrukturierten Daten. Eines der Produkte in diesem Bereich ist der *Content Analyzer* [IBM]

(früher *OmniFind Analytics Edition* (OAE)). IBMs Hintergrund der strukturierten Daten wird bei der Zielsetzung und der Anwendung von *Content Analyzer* deutlich. Es lassen sich große Mengen unstrukturierter Textdaten gemeinsam mit strukturierten Daten analysieren. Somit stellt der *Content Analyzer* ein Bindeglied zwischen der reinen Analyse von Textdaten und dem klassisch an strukturierten Daten orientierten Data Mining dar.

Andere Anbieter, wie bspw. das italienische Unternehmen *Expert System* bieten originäre Textanalyse-Systeme an. Mit der *COGITO*-Anwendungspalette [Expb] stellt *Expert System* eine Sammlung von sich teilweise ergänzenden Anwendungen zur Verfügung, die zur Analyse großer Mengen von Texten genutzt werden können. Der Schwerpunkt dieser Anwendungen liegt in einem weitgehenden Sprachverständnis und einer darauf aufbauenden semantischen Analyse der untersuchten Texte. Das Herzstück dabei sind sprachspezifische semantische Netze, die tausende Begriffe und deren semantische Beziehung untereinander definieren.

Während die beiden zuvor genannten Systeme vornehmlich für die Nutzung durch Textanalyse-Experten konzipiert sind, so geht das französische Unternehmen *Temis* einen anderen Weg. Deren Textanalyse-Lösung *Luxid* [Tem] setzt einen Schwerpunkt auf die Generierung von Reports. Diese, einmal definiert durch einen Textanalyse-Experten, können dann von ungeschulten Benutzern bei Bedarf neu berechnet und aktualisiert werden. Dazu bietet *Luxid* ein Rollenkonzept, das die Fähigkeiten und Berechtigungen der jeweiligen Benutzer ins System abbildet und entsprechende Funktionen zugänglich macht. Gleichzeitig stellt es eine Erweiterungsplattform zur Verfügung, über die eigene Analyse- und Report-Komponenten eingebunden werden können.

3.2.4 Abgrenzung

Die Textanalyse stellt einen interessanten Ansatz dar und ist als Konzept prinzipiell auch für die Lösung vieler der dieser Arbeit zugrunde liegenden komplexen

Fragestellungen geeignet. Aktuell verfügbare Produkte werden jedoch den speziellen Anforderungen dieser Fragestellungen nicht gerecht, da sie unter anderen Zielvorgaben entwickelt wurden.

So sind diese Produkte stark auf den jeweiligen Einsatzbereich fokussiert und bieten damit im Wesentlichen Unterstützung für Fragestellungen aus den jeweils adressierten Bereichen (Sentiment Detection, Quality Early Warning, Intelligence Services etc., vgl. Kapitel 3.2.2). Erweiterungen und Anpassung für andere Fragestellungen sind nur bedingt möglich. Im Gegensatz dazu sollen im Rahmen dieser Arbeit Methoden und Techniken entwickelt werden, die jede Art von komplexer Suche im WWW unterstützen, ohne einen bestimmten Themenbereich oder Einsatzzweck vorzugeben, sowie Techniken, die eine Anpassung existierender Systeme an neue Suchaufgaben ermöglichen. Dass hierbei Kompromisse zwischen Universalität und potenzieller Ergebnisqualität eingegangen werden müssen, ist offensichtlich.

Eine prinzipielle Abgrenzung gegenüber dem Konzept der Textanalyse an sich soll an dieser Stelle jedoch nicht vorgenommen werden. Textanalyse ist ein Oberbegriff für eine Sammlung verschiedener Konzepte und Techniken, wobei der Schwerpunkt auf der Nutzung sowohl strukturierter als auch unstrukturierter Datenquellen liegt. Neben den angesprochenen Unterschieden zu etablierten Produkten existieren demnach auch viele Gemeinsamkeiten mit den in dieser Arbeit eingesetzten oder entwickelten Konzepten.

3.3 Explorative Suche

Die „explorative Suche“ [Mar06][WR09] bezeichnet eine Klasse von Suchen, die eine aktive Beteiligung des Suchenden über den gesamtem Suchprozess hinweg erfordern. Eine solche Beteiligung des Suchenden ist genau dann nötig, wenn der Suchende bspw. nicht exakt formulieren kann, wonach er sucht. Oder aber wenn die gesuchten Informationen nicht in Form expliziter Fakten zu finden sind, sondern eine Analyse von Texten und ggf. deren Querbezügen

durchgeführt werden muss (vgl. Kapitel 2). In beiden Fällen kann das Suchsystem nicht die fertige Antwort auf die Fragestellung des Suchenden liefern, sondern kann ihn lediglich unterstützen, indem es bspw. Hilfestellung bei der Formulierung der Fragestellung gibt und die Auswertung von Textsammlungen und Suchergebnissen unterstützt.

Im Bereich der *Human-Computer Interaction* (HCI) werden hier zum einen Strategien zur Visualisierung von Suchergebnissen diskutiert, um dem Suchenden die Orientierung in den von Suchmaschinen zurück gemeldeten Ergebnissen zu erleichtern [WKB05][CM08][KDH11]. Zum anderen werden Ansätze entwickelt, die den Prozesscharakter solcher Suchen besser unterstützen, als dies durch herkömmliche Suchmaschinen geleistet wird. Häufig werden hierzu integrierte Suchwerkzeuge vorgeschlagen, die verschiedene Such- und Analysekomponenten anbieten und miteinander verbinden, um Suchen bzw. Ergebnisse aus unterschiedlichen Blickwinkeln bearbeiten zu können. Beispiele hierfür sind der *SocialSearchBrowser* [CNCO10] zur Unterstützung kollaborativer Suchen im mobilen Umfeld oder Werkzeuge zur Suche in Multimediadatenbanken [Chr08]. Meist sind diese jedoch speziell für eine Domäne entwickelt und können somit nicht oder nur umständlich für Suchen auf anderen Themenbereichen eingesetzt werden.

3.4 Expertensuche

Wie in Kapitel 2 ausgeführt, ist die Problemstellung der Suche nach Experten zu bestimmten Themen eine der Motivationen für die vorliegende Arbeit. Zwar stellt sie nur einen Spezialfall der übergeordneten Problemstellung komplexer Webrecherchen dar. Da die Expertensuche im Rahmen dieser Arbeit an verschiedenen Stellen als Beispiel und Anwendungsszenario dient, wird an dieser Stelle dennoch ein knapper Überblick über relevante Arbeiten gegeben.

Die Suche nach Experten war in den vergangenen Jahren Gegenstand verschiedener Forschungsarbeiten. [MA00] sowie [YSK03] bieten dazu einen guten

Überblick. Die meisten Arbeiten bewegen sich dabei im Kontext des Wissensmanagements und fokussieren auf eine firmeninterne Suche nach Experten, ähnlich zu Gelbe-Seiten-Systemen. Im Folgenden sollen verschiedene beispielhafte Ansätze und Werkzeuge vorgestellt werden, die sich mit der Expertensuche oder allgemein der Websuche beschäftigen. Ihre Stärken und Schwächen im Kontext der hier bearbeiteten Problemstellung werden diskutiert.

Am Fraunhofer-IPA wurde der *Xpertfinder* [HS02] entwickelt. Er bietet ein Softwarewerkzeug zum Monitoring von E-Mail-Kommunikation und zur Analyse von Dateien im PDF- und MS-Wordformat, die sich auf verschiedenen Servern innerhalb eines Unternehmensnetzwerkes befinden können. Die Software extrahiert Kommunikationspartner aus dem E-Mail-Verkehr, ermittelt Inhalte, bringt sie mit den Autoren in Verbindung und leitet aus diesen Daten eine Abbildung von Mitarbeitern auf vordefinierte Expertise-Bereiche ab. Damit vereinfacht der *Xpertfinder* die Erstellung eines Gelbe-Seiten-Systems und kann helfen, den zu dessen Pflege nötigen Wartungsaufwand erheblich zu reduzieren. Allerdings ist dieser Ansatz auf den Einsatz in relativ kleinen, klar abgegrenzten Netzen, wie bspw. Firmen-Intranets, beschränkt und kann daher nicht direkt auf eine Suche im WWW angewandt werden. Der Grund dafür liegt im Wesentlichen in der Tatsache, dass Suchergebnisse sich auf eine verhältnismäßig kleine Anzahl von Dokumenten bzw. Mails beziehen, die zudem einer entsprechenden Vorverarbeitung durch das System bedürfen. Offensichtlich kann aber diese Vorverarbeitung aller Dokumente des Suchraumes nur dann durchgeführt werden, wenn entweder der Suchraum klein ist, wie bspw. im Falle eines Firmen-Intranets, oder wenn sehr hohe Kapazitäten in Bezug auf Bandbreite, Rechen- und Speicherkapazität zur Verfügung stehen. Letzteres wäre für etablierte Suchmaschinen der Fall, ist aber im Kontext eines hier angestrebten Suchwerkzeugs nicht realisierbar (vgl. Kapitel 3.1.1). Ferner liegt der Fokus von *Xpertfinder* auf der Analyse von E-Mail-Kommunikationsstrukturen. Im Kontext eines Firmennetzes kann das wertvolle Informationen über Kompetenznetze aufdecken und als adäquates Mittel angesehen werden. Für E-Mail im Allgemeinen, also im Kontext des gesamten Internet, ist dieser Weg aber nicht gangbar, da nur solche Kommunikation analysiert werden kann, die über von der Soft-

ware kontrollierte Mail-Server abgewickelt wird und damit insbesondere keine beliebigen WWW-Inhalte analysierbar sind.

Ein anderer Aspekt der Expertenidentifikation wird von der Forschung im Bereich *virtueller Gemeinschaften* (engl. “virtual communities”) behandelt. Dort werden bspw. Methoden und Werkzeuge zur Bestimmung des Grades an Interaktion zwischen den Teilnehmern einer Diskussionsrunde entwickelt. Während sich daraus zwar schwerlich direkt Informationen bezüglich der Expertise einer Person ableiten lassen, so liefern solche Verfahren dennoch wertvolle Einsichten über die Entwicklung einer Diskussion und deren Teilnehmer. Aus den so gewonnenen Daten kann herausgelesen werden, wer zu den aktivsten Teilnehmern der Diskussion zählt, welche Gruppen oder Subnetze sich bilden und daraus auf sich entwickelnde Wissensnetze geschlossen werden. Das *Management Cockpit* [Tri05] stellt ein solches Werkzeug zur Visualisierung von Wissensnetzen dar. Es ermöglicht die Analyse und Visualisierung der Kommunikationsstrukturen diverser Web-Foren. Für die Suche nach Experten im WWW ist dieser Ansatz jedoch zu eingeschränkt. Der Benutzer muss zunächst eigenhändig passende Foren identifizieren und entsprechende Zugriffskomponenten implementieren, um die Daten einer automatisierten Verarbeitung zugänglich zu machen. Ferner visualisiert das *Management Cockpit* zwar die Kommunikationsstrukturen, differenziert aber nicht auf inhaltlicher Ebene zwischen einzelnen Beiträgen. Es kann also lediglich eine Aussage über Strukturen bezüglich des gesamten Forums getroffen werden, eine Detaillierung auf einzelne Unterthemen ist aber nicht möglich. Sollen aber hochwertige Ergebnisse bei der Suche nach Experten produziert werden, so ist genau diese Ebene der Detaillierung von Nöten.

Neben den o.g. Forschungsansätzen existieren auch verschiedene kommerzielle Plattformen zur Suche nach Experten. Prominente Beispiele hierfür sind sog. Soziale Netzwerke (engl. „social networks“) wie *BizWiz* [Biz], *LinkedIn* [Linb] oder im deutschsprachigen Raum *XING* [XIN]. Hier registrieren sich Nutzer anhand eines selbst definierten Kompetenz- und Interessenprofils und können über diese Plattform von anderen Nutzern gefunden und kontaktiert werden. Auch klassische Jobbörsen wie *Monster* [Mon] oder *Experteer* [Expa] lassen sich in diese Kategorie einordnen. Der gravierende Nachteil solcher Plattformen liegt

für dieses Szenario jedoch darin, dass Benutzer sich eigeninitiativ registrieren müssen und in der Beschreibung ihrer Kompetenzen keinen Beschränkungen oder Kontrollen unterliegen. Zwar mag die Suche in solchen Netzen damit als ein erster Schritt dienen. Da jedoch oftmals nur in ausgesuchten Bereichen wie bspw. der IT eine signifikante Anzahl an registrierten Benutzern existiert und zudem deren Daten nicht von unabhängiger Stelle verifiziert werden, ist der Nutzen solcher Plattformen im Kontext der Expertensuche beschränkt.

Auch im Rahmen der TREC-Konferenzen wurde das Thema Expertensuche (engl. „expert finding“) seit 2005 behandelt [CdVS05] [SdVC06] [BdVCS07] [BST+08] [BdVS+10]. Die Problemstellung hierbei war, in einer fixen Dokumentensammlung Personen zu identifizieren, diese bestimmten vorgegebenen Themen zuzuordnen und ihre Expertise bezüglich der jeweiligen Themen zu bewerten. Das Szenario hinter der Aufgabenstellung unterscheidet sich jedoch signifikant vom dieser Arbeit zugrunde liegenden Szenario: bei TREC geht es um die Identifizierung von Personen in einer bekannten Dokumentenmenge und deren Zuordnung zu vorab bekannten Themen. Bspw. lautete die Aufgabenstellung in TREC 2008:

Ein Mitarbeiter des Unternehmens (hier: CSIRO) beantwortet eine E-Mail-Anfrage zu einem bei CSIRO behandelten Thema. Dazu durchsucht er den öffentlichen Web-Auftritt von CSIRO nach Ressourcen und möglichen Antworten. Zudem versucht er, Experten zu dem betreffenden Thema innerhalb von CSIRO zu finden, die ihm bei der Beantwortung der Anfrage mit Detailinformationen behilflich sein können.

Für die praktische Anwendung in einem Unternehmen bedeutet das, dass entsprechende Personen/Themen-Zuordnungen vorab berechnet werden können und dass insbesondere die in Frage kommenden Personen und Themen vorab bekannt sind. Im diese Arbeit motivierenden Szenario hingegen trifft keine der genannten Einschränkungen zu, weshalb notwendigerweise andere Techniken zum Einsatz kommen müssen: es müssen komplexere Techniken zur Personenidentifikation, zur Beschreibung von Suchthemen, zur Identifikation relevanter Webressourcen und zur Bewertung der Expertise von Personen oder Institutionen eingesetzt werden.

3.5 Zusammenfassung

In diesem Grundlagenkapitel wurden Konzepte erläutert, die im Rahmen der vorliegenden Arbeit besondere Relevanz besitzen. Zwei Schwerpunkte wurden dabei betrachtet: zum einen grundlegende Suchmaschinentechnologien und rein syntaktische Analysen auf Textdokumenten. Zum anderen die Analyse von Zusammenhängen und die Extraktion höherwertiger Informationen aus diesen Texten.

Neben der Diskussion grundlegender Konzepte, die bereits im industriellen Umfeld Anwendung finden, wurde auch aufgezeigt, in welchen Bereichen die Forschung derzeit aktiv ist, welche Konzepte dort entwickelt werden und in wie weit diese zur Lösung der hier diskutierten Probleme beitragen können.

Zusammenfassend lässt sich festhalten, dass viele Ansätze zur Unterstützung bei der Durchführung von Websuchen existieren und diese auch, teilweise seit längerem, erfolgreich eingesetzt werden. Für die Bearbeitung von komplexen Websuchen jedoch sind diese existierenden Ansätze nicht immer ausreichend. Am Beispiel der Expertensuche wurde eine solche Suche diskutiert. In den beiden folgenden Kapiteln werden daher Vorschläge ausgearbeitet, wie eine effizientere Unterstützung des Benutzers bei der Bearbeitung solcher komplexer Fragestellungen erreicht werden kann.

KAPITEL 4

SOFTWAREPLATTFORM FÜR SUCHANWENDUNGEN

Dieses Kapitel beschäftigt sich mit dem Problem der Entwicklung von Suchwerkzeugen und der in Abhängigkeit der Fragestellung wechselnden Anforderungen an diese. Es wird ein Konzept vorgeschlagen und diskutiert, das die Entwicklung solcher Werkzeuge in Form eines Frameworks, der „Suchplattform“, unterstützt. Im Anschluss wird die Implementierung einer Software zur Suche nach Experten beschrieben, die auf dieser Plattform aufsetzt.

Ausgangspunkt für diese Arbeit war die Beobachtung, dass sowohl im industriellen Umfeld als auch im Bereich der Forschung lediglich suchunterstützende Softwaresysteme existieren, die auf spezielle Fragestellungen oder Typen von Suchen zugeschnitten sind, bspw. Textanalyse-Tools für das sog. „Sentiment Detection“ (vgl. Kapitel 3.2), die Patentrecherche [KB11] oder herkömmliche Web-Suchmaschinen. Allgemeinere Werkzeuge, die eine umfassende Unterstützung für verschiedene Suchanwendungen bieten, sind dahingehend statisch, dass sie lediglich übergreifende Basisfunktionalität vorhalten und dem Benut-

zer die Auswertung und weiter gehende Analyse der erzielten Suchergebnisse überlassen[BBCF10][BBC+11]. Letzteres kann auch kaum verwundern, da beliebige Fragestellungen denkbar sind und jede dieser Fragestellungen ggf. andere oder angepasste Analyseansätze verlangt. Auch nimmt die Zahl der möglichen Anwendungsszenarien für ein solches Werkzeug mit der steigenden Informationsmenge des WWW stetig zu, was ebenfalls spezielle Suchtechniken nötig machen kann. Daher ist ein starres System mit einer festgelegten Menge an Such- und Analysewerkzeugen nicht zielführend, da es niemals vollständig sein kann und immer Fragestellungen existieren werden, für die es keine oder nur unzureichende Suchunterstützung bietet.

Aus diesen Überlegungen heraus wurde im Rahmen der vorliegenden Arbeit ein Konzept für eine Software entwickelt, die als Basis für eine effiziente Suche im Web dienen kann, entsprechende Erweiterungsmöglichkeiten bietet und eine Anpassbarkeit an verschiedenste Suchszenarien sicher stellt. Dieses Konzept wurde in Form einer „Suchplattform“ umgesetzt. Die Plattform implementiert verschiedene Konzepte und Techniken, die sich in der praktischen Anwendung als hilfreich für die Strukturierung und Analyse von Suchergebnissen und damit für die Lösung von komplexen Such-Fragestellungen erwiesen haben [KSJ07] und stellt diese als Grundfunktionalität zur Verfügung. Großer Wert wurde dabei auf die Erweiterbarkeit gelegt, um die Plattform an Suchszenarien anpassbar zu machen, die zum Zeitpunkt der Entwicklung noch nicht vorgesehen waren. Der zentrale Leitfaden bei der Entwicklung der Plattform war daher der, den einzelnen Komponenten nötige Grundfunktionalitäten und eine Infrastruktur zu bieten, auf der weitere Komponenten aufbauen und insbesondere interagieren können. Im Vorgriff auf die folgenden Abschnitte seien an dieser Stelle zur Verdeutlichung einige dieser Komponenten beispielhaft genannt: ein Focused Crawler, Schnittstellen zu herkömmlichen Suchmaschinen, Werkzeuge zur Analyse von Hyperlink-Graphen und zur Generierung von Suchanfragen.

4.1 Konzept der Suchplattform

Wie oben bereits erläutert, kann es nicht das Ziel der Bemühungen sein, eine umfassende monolithische Anwendung zur Lösung aller Suchprobleme zu entwickeln. Vielmehr ist eine Sammlung von Techniken und Strategien gefordert, auf die zur Lösung eines Suchproblems zurückgegriffen werden kann. Dieses Prinzip, einzelne funktionale Komponenten anzubieten, sie um zusätzliche Komponenten zu erweitern und eine gemeinsame Integrationsbasis zu schaffen, ist das klassische Ziel des Framework-Ansatzes und stellt auch die Grundlage für die im Folgenden vorgestellte Suchplattform dar. Die Umsetzung erfolgt in Form eines modularen Frameworks: Neben verschiedenen Komponenten, die Basisfunktionalitäten implementieren, bietet die Suchplattform dabei eine Infrastruktur für Erweiterungen und zur Interaktion der angebundenen Komponenten. Somit lassen sich an die jeweilige Suchaufgabe angepasste Analyse-Werkzeuge entwickeln, die auf der Infrastruktur und Funktionalität der Suchplattform und ihrer übrigen Komponenten aufsetzen und diese benutzen können [KJWS07].

Architektonisch orientiert sich die Suchplattform dabei an der klassischen Dreiteilung in *Präsentationsschicht*, *Verarbeitungsschicht* und *Datenhaltungsschicht* (vgl. Abbildung 4.1). Die drei Schichten sind weitgehend unabhängig von einander konzipiert und greifen gegenseitig auf benötigte Funktionalität über klar definiert Schnittstellen zu. Dieses Konzept bildet die Grundlage zum einen für eine leichte Erweiterbarkeit durch Eigenimplementierung für spezielle Suchanforderungen. Zum anderen ermöglicht dies die Anwendung der Suchplattform in anderen Kontexten als dem ursprünglich vorgesehenen. So wurde die Suchplattform bspw. für den Einsatz in Applikationen entwickelt, die als eigenständige Prozesse auf dem Rechner des Benutzers laufen und mittels klassischer GUI-Elemente mit diesem interagieren. Durch den Austausch bzw. eine Anpassung der Präsentationsschicht jedoch kann sie mit geringem Aufwand auf die Anforderungen bspw. einer Web-Anwendung abgestimmt werden.

Im Folgenden werden die drei Schichten der Suchplattform in der Reihenfolge ihrer Sichtbarkeit für den Benutzer skizziert.

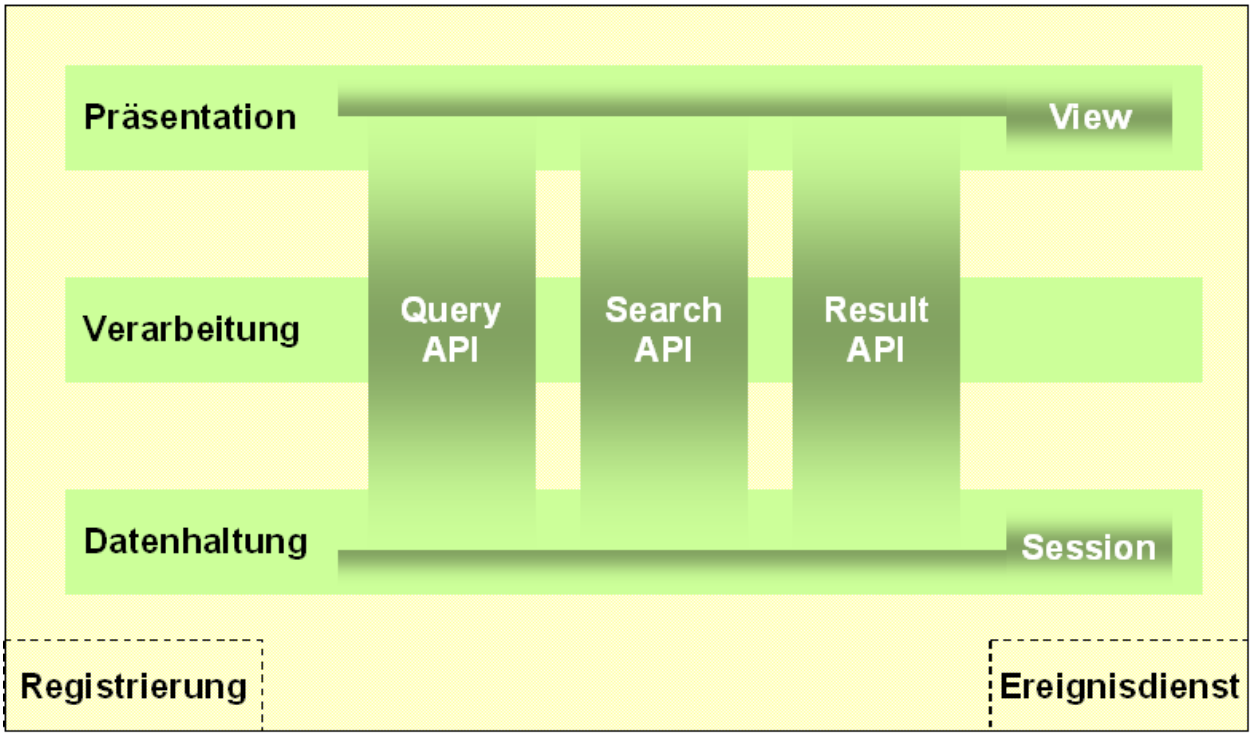

Abbildung 4.1: Komponenten der Suchplattform

4.1.1 Präsentationsschicht

Die GUI (engl. Graphical User Interface) ist die Schnittstelle zwischen Benutzer und Anwendung. Aufgabe der Präsentationsschicht ist es, GUI-Komponenten und -Dienste bereit zu stellen, die in verschiedenen Suchanwendungen eingesetzt werden können, ohne jedoch konkrete Vorgaben hinsichtlich optischer Gestaltung und Layout zu machen. Letztere Aspekte sind Aufgabe der konkreten Anwendung und können von einem Anwendungs-Framework schwerlich festgelegt werden. Die Suchplattform stellt daher keine komplette Anwendungsoberfläche zur Verfügung, sondern in erster Linie Basis-Komponenten zur Entwicklung und Integration von eigenen Oberflächenelementen.

Die zentrale nach außen sichtbare Struktur dieser Schicht ist die `ViewAPI`, die Klassen, Interfaces und Methoden für die Interaktion der Komponente mit der Plattform deklariert. Diese Interaktionen umfassen bspw. die Regelung der Sichtbarkeit einer Komponente, das Verwalten von Listen über GUI-Komponenten

für bestimmte Prozessschritte etc.

Aufbauend auf diesem Konzept sind verschiedene Komponenten implementiert, die jedoch selbst nicht essentieller Bestandteil der Plattform sind. Sie werden in Form einer Bibliothek vorgehalten und können von anderen Anwendungen bei Bedarf eingesetzt werden. Beispiele für solche Komponenten sind der *QueryTree* zur grafischen Modellierung von komplexen booleschen (Schlüsselwort-) Suchanfragen, der *QueryEditor* zur Bearbeitung der textuellen Form von Schlüsselwort-Suchanfragen, die *ResultList* zur Darstellung von Ergebnislisten einer Suche oder Analysekomponenten wie die *LinkStructure* zur Visualisierung der Hyperlinks zwischen einer Dokumentenmenge (bspw. der aktuell angezeigten Ergebnisliste).

4.1.2 Verarbeitungsschicht

Die Verarbeitungsschicht stellt den Kern der Suchplattform dar. Sie implementiert den dreistufigen Suchprozess „Anfrage-Verarbeitung-Ausgabe“ und definiert Schnittstellen für Suchanfragen bzw. Suchergebnisse sowie für die Weiterleitung dieser Strukturen zwischen den einzelnen Komponenten.

Die drei wichtigsten Schnittstellen der Verarbeitungsschicht sind die `QueryAPI`, die `SearchAPI` sowie die `ResultAPI`. Sie bilden den Kern der Verarbeitungsschicht und stellen damit die Grundlage für den Daten- und Kontrollfluss dar. Ihre Modellierung erfolgte entlang des genannten dreistufigen Prozesses.

4.1.2.1 QueryAPI

Die `QueryAPI` bietet Datenstrukturen und Methoden zur Spezifikation und Verwaltung von Suchanfragen. Um eine Wiederverwendbarkeit und Interoperabilität zwischen verschiedenen Komponenten zu gewährleisten, wurde ein allgemeines Konzept für die Repräsentation von Suchanfragen entwickelt. Es sieht dabei die Möglichkeit vor, zwischen den im jeweiligen Anwendungsfall benötigten Darstellungen der Anfrage (bspw. die Form der Schlüsselwortanfrage

oder des Operatorbaums) ohne Informationsverlust wechseln zu können und ggf. in einer Darstellung nicht definierte Aspekte zu ignorieren.

Erreicht wird dies durch eine Anlehnung an die in Schlüsselwortanfragen verbreiteten hierarchischen Strukturen aus Operatoren (bspw. UND, ODER, NICHT) und Operanden (in diesem Falle die Schlüsselworte). Dies ermöglicht eine direkte Abbildung dieses häufig verwendeten Anfragetyps. Zusätzliche Konzepte wie bspw. die von verschiedenen Suchmaschinen unterstützten „erweiterten Funktionalitäten“ (Beschränkung der Suche auf einzelne Domains, zeitliche Aspekte etc.) können über frei definierbare Operatoren in die Anfragestruktur eingebunden werden. Auch erweiterte Konzepte, wie bspw. die Spezifikation einer Anfrage anhand von „ähnlichen Dokumenten“ (vgl. Kapitel 6) lassen sich auf dieses Schema der Operatoren und Operanden abbilden.

Ein einfaches Beispiel für die Anwendung dieses Konzepts ist die Erstellung einer Suchanfrage in einer Komponente zur Eingabe von Schlüsselworten und die (ggf. parallele) Änderung der Anfrage mittels eines booleschen Operanden-/Operatorenbaumes. Änderungen in einer Darstellung werden dabei automatisch an die übrigen Komponenten gemeldet, wodurch eine Synchronisation aller beteiligten Komponenten erreicht werden kann. Der Benutzer kann damit jederzeit zwischen den einzelnen Ausprägungen der Anfrage wechseln und sie in einer der jeweiligen Situation angemessenen Repräsentation verarbeiten.

Da nicht zwangsläufig jede Repräsentation alle Konzepte einer Suchanfrage darstellen kann, kann auch die Transformation von einer Darstellung in die andere nicht immer völlig verlustfrei erfolgen. Die QueryAPI sieht daher Möglichkeiten vor, neben der Repräsentation der Basis-Suchanfrage auch kontextspezifische Aspekte der Anfrage zu verwalten. Dies erfolgt über eine Sammlung sog. *Properties*, die jede Komponente mit einer Suchanfrage assozieren kann. Der Vorteil dieses Ansatzes liegt darin, dass der Benutzer sich nicht auf eine Methode zur Anfragespezifikation festlegen muss, sondern die Stärken verschiedener Methoden für die Erstellung oder Optimierung einer Anfrage nutzen kann. Dabei erfolgt eine automatische und weitgehend vollständige Transformation zwischen den unterschiedlichen Ausprägungen der Suchanfrage.

4.1.2.2 SearchAPI

Mittels der *SearchAPI* können Suchstrategien implementiert werden. Diese API bietet Schnittstellen zu herkömmlichen Suchmaschinen wie Google, Yahoo oder MSN und erlaubt damit deren Einbindung in den Suchprozess. Neben der klassischen Spezifikation von Suchanfragen in Form von Schlüsselworten, erlauben diese Konnektoren auch den Einsatz der von der jeweiligen Suchmaschine angebotenen erweiterten Syntax. Über die o. g. *QueryAPI* erfolgt dabei wo möglich eine Transformation von einer Syntax in die andere. Die Liste der unterstützten Suchmaschinen kann durch die Entwicklung neuer Konnektoren entsprechend erweitert werden. Kapitel 5 widmet sich dieser Fragestellung im Detail.

Neben den Schnittstellen zu existierenden Suchmaschinen bietet die *SearchAPI* auch die Möglichkeit, eigene Entwicklungen zu integrieren. Auf dieser Basis sind bspw. der in Kapitel 4.2 im Detail diskutierte Focused Crawler sowie eine Metasuchmaschine zur Integration der Ergebnisse mehrerer der o. g. existierenden Suchmaschinen implementiert.

Da sich diese Suchen unter Umständen über einen längeren Zeitraum erstrecken und nicht, wie von herkömmlichen Suchmaschinen gewohnt, innerhalb weniger Millisekunden terminieren, bietet die *SearchAPI* entsprechende Funktionalität zur Parallelisierung mehrerer Suchanfragen, sowie zu deren Überwachung und Steuerung.

4.1.2.3 ResultAPI

Die Verwaltung von Ergebnissen bzw. Zwischenergebnissen einer Suche erfolgt anhand von Komponenten, die über die *ResultAPI* zugreifbar gemacht werden. Dabei wird für jede Suchkomponente eine separate Ergebnisliste verwaltet, die ab dem Start der Suche für andere Komponenten zugreifbar ist. Dies ermöglicht die Darstellung und Analyse nicht nur der Ergebnisse, sondern auch des Fortschritts der Suche, während der Suchvorgang noch aktiv ist. Insbesondere für

lang laufende Suchen, wie beim Einsatz eines Focused Crawlers, ist diese Funktionalität zwingend erforderlich. Neben den eigentlichen Ergebnislisten machen die Suchkomponenten über diese Schnittstelle auch zugehörige Metadaten zur weiteren Verwendung publik. Diese Metadaten umfassen bspw. Fehlermeldungen des Suchdienstanbieters sowie die tatsächlich an den Suchdienst gestellte Suchanfrage. Letztere kann sich von der vom Benutzer spezifizierten Anfrage unterscheiden, da nicht jede Suchmaschine alle von der Suchplattform unterstützten Syntax-Elemente einer Anfrage unterstützt (vgl. Ausführungen zur *QueryAPI*).

Viele Komponenten der Verarbeitungsschicht haben entsprechende Pendants in der Präsentationsschicht. Dies ist allerdings nicht zwingend erforderlich, und so können Komponenten existieren, die in der GUI nicht präsent und damit dem Benutzer nicht direkt zugänglich sind. Meist kapseln sie Aufgaben, die von mehreren Komponenten verwendet werden. Die horizontale Trennung in Anfrage, Suche und Analyse schränkt dabei jedoch nicht den Zugriff auf diese Komponenten ein, d. h. diese Werkzeug-Komponenten stehen – wie alle anderen Komponenten auch – in der gesamten Plattform zur Verfügung.

4.1.3 Datenhaltungsschicht

Die unterste Schicht in Abbildung 4.1 stellt die Datenhaltungsschicht dar. Sie enthält sämtliche Funktionalität zur Speicherung von Daten, die rund um Suchprozesse anfallen. Die zentrale API dieser Schicht ist die *SessionAPI*. Über sie erfolgt die Verwaltung aller Daten im Rahmen einer sog. Sitzung. Eine Sitzung stellt dabei den Rahmen für die Suche zu einem Thema dar. Sie kann beliebig oft unterbrochen und fortgesetzt werden, wobei der aktuelle Suchstand jeweils erhalten bleibt. Über die *SessionAPI* werden heruntergeladene Dokumente zugreifbar gemacht, es können Statistiken zum Verlauf der Suche, Suchanfragen und Ergebnislisten eingesehen werden, ebenso wie Bookmarks und benutzerdefinierte Notizen.

Der Persistenz-Dienst kann dabei von allen Komponenten genutzt werden, unabhängig von der Schicht, in der sie implementiert sind.

4.1.4 Registrierung und Ereignisdienst

Im Allgemeinen setzt sich ein mit Hilfe der Suchplattform implementiertes Suchwerkzeug aus mehreren Komponenten zusammen, die sich über mehrere der drei skizzierten Schichten erstrecken können. Eine 1:1-Beziehung zwischen GUI und Verarbeitung ist jedoch nicht gefordert. Vielmehr können beliebige m:n-Beziehungen zwischen Komponenten der drei Schichten aufgebaut werden, wo dies aus Gründen der Flexibilität oder der erweiterten Funktionalität sinnvoll ist. Bspw. können ein und der selben Verarbeitungskomponente als Schnittstelle mehrere GUI-Komponenten zugeordnet sein, die dem Benutzer jeweils eine andere Sicht auf die verarbeiteten Daten ermöglichen. Ebenso kann es alternative Implementierungen für ein Werkzeug geben, wie bspw. Konnektoren zu verschiedenen Suchmaschinen.

Diese Flexibilität ist der Kerngedanke der Suchplattform und Grundvoraussetzung für die geforderte Erweiterbarkeit. Sie ermöglicht die Entwicklung neuer Komponenten auf allen drei Schichten und erlaubt auch, bestehende Komponenten durch alternative Implementierungen zu ersetzen bzw. für eine Aufgabe zwischen mehreren existierenden Implementierungen zu wählen. Diese einzelnen Komponenten und ihr Zusammenspiel zu koordinieren, ist dabei Aufgabe der komponentenübergreifenden Module *Registrierung* und *Ereignisdienst*.

Die *Registrierung* verwaltet eine Liste aller in der Suchplattform verfügbaren Komponenten. Um eine neue Komponente einzubinden, wird diese bei der Registrierung angemeldet und kann fortan mit allen anderen Komponenten interagieren. Dies bedeutet zum einen, dass sie auf andere Komponenten zugreifen kann, zum anderen aber auch, dass sämtliche andere Komponenten auf sie zugreifen können. Der komponentenübergreifende Zugriff erfolgt dabei stets anhand einer der jeweiligen Komponente zugeordneten Schnittstelle. Für die Interaktion zweier Komponenten ist also lediglich die gegenseitige Kenntnis

der öffentlichen Schnittstelle nötig, Implementierungsdetails sind hierbei nicht von Belang. Die öffentliche Schnittstelle wird bei der Anmeldung einer Komponente in der Registrierung hinterlegt und kann von anderen Komponenten dort angefordert werden.

Neben der Schnittstellenverwaltung bildet die Registrierung auch die Grundlage für den *Ereignisdienst*. Mittels des Ereignisdienstes können Komponenten auf Ereignisse in anderen Komponenten reagieren, bspw. auf den erfolgreichen Download einer Webressource. Auch hier muss die ereigniserwartende Komponente keine Kenntnis von der Implementierung der ereignisauslösenden Komponente haben. Vielmehr kann sie einen *Listener* auf den Ereignistyp registrieren und erhält von da an alle Nachrichten dieses Typs, die von einer beliebigen registrierten Komponente über den Ereignisdienst verschickt werden. Das impliziert, dass auch die ereignisauslösende Komponente keine Kenntnis von den ereigniserwartenden Komponenten haben muss, was entsprechend die Abhängigkeiten zwischen den einzelnen Komponenten reduziert und die Anwendung somit einfacher wartbar macht.

Die wichtigsten Ereignis-Schnittstellen sind in den APIs *QueryAPI*, *SearchAPI* und *ResultAPI* definiert. Sie sind für die Erweiterbarkeit der Suchplattform von zentraler Bedeutung. Anhand dieser Schnittstellen kann eine Komponente Informationen über den gesamten Suchprozess sammeln, indem sie Listener für die jeweiligen Ereignisse implementiert und registriert. Über die Ereignisse der *SearchAPI* kann bspw. der aktuelle Stand der Suche ermittelt werden, über die *QueryAPI* die zur Suche gehörenden Suchanfragen. Über die *ResultAPI* werden entsprechend Informationen zu Suchergebnissen publiziert. Neue Komponenten können anhand dieser drei Schnittstellen in alle drei Phasen des Suchprozesses eingebunden werden und auf alle relevanten Informationen zugreifen.

Registrierung und Ereignisdienst bilden somit den integrierenden Rahmen für alle Komponenten und ermöglichen diesen einen Austausch aller anfallenden Informationen.

4.2 Implementierung der Expertensuche auf Basis der Suchplattform

Im Folgenden wird die prototypische Implementierung einer Suchanwendung vorgestellt, die auf der oben vorgestellten Suchplattform aufsetzt. Es wird ein einfacher Suchprozess eingeführt und gezeigt, wie die Suchplattform diesen Prozess unterstützt, wie verschiedene Komponenten der Plattform miteinander interagieren und gegenseitig auf (Teil-)Ergebnissen aufbauen können. Dieser Abschnitt weist somit die Praxistauglichkeit der im vorangegangenen Kapitel 4.1 vorgestellten Suchplattform nach und bildet die Grundlage für die in den folgenden Kapiteln 5-7 geführten Diskussionen über zentrale Konzepte.

Auch die diese Arbeit mit motivierende Fragestellung der Expertensuche ist in die in Kapitel 2 beschriebene Klasse der komplexen Fragestellungen einzuordnen. Sie wird vornehmlich in Unternehmen aufgeworfen und weniger in privaten Bereichen des alltäglichen Lebens. In größeren Unternehmen und Konzernen findet sich benötigtes Wissen oftmals firmenintern. Kleine und mittelständische Unternehmen (KMU) hingegen beklagen in dieser Hinsicht häufig einen Mangel, der sich nicht zuletzt aus der geringeren Mitarbeiterzahl und der häufig größeren Spezialisierung dieser ergibt. Insbesondere dann, wenn das benötigte Wissen eher im Randbereich der eigentlichen Firmenkompetenzen angesiedelt ist oder wenn gar technologisches Neuland betreten werden soll. Um diesen Mangel an firmeninternem Wissen zu kompensieren, werden in der Regel externe Experten als Berater zur Durchführung von Projekten oder zur Einschätzung bestimmter Sachverhalte hinzugezogen. Um solche Experten ausfindig zu machen, können verschiedene Quellen wie Fachzeitschriften und -bücher oder auch persönliche Kontakte genutzt werden. Vor dem Hintergrund, dass das zentrale Thema dieser Arbeit aber die Suche im WWW ist und aus der Tatsache heraus, dass das WWW eine große und stetig wachsende Informationsquelle darstellt, soll an dieser Stelle eine Fokussierung auf eine Suche nach Experten im WWW erfolgen und die Nutzbarkeit von dort verfügbaren Informationen für diesen Zweck untersucht werden.

Wie in Kapitel 3.4 dargestellt, ergeben sich verschiedene Schwierigkeiten, die eine Anwendung existierender Suchwerkzeuge auf die Expertensuche nicht sinnvoll erscheinen lassen bzw. wo diese an ihre Grenzen stoßen (insbesondere deren Beschränkung auf Intranets bzw. auf bekannte und vorab umfassend analysierbare Suchräume). Allerdings kommen aber auch bei diesen Systemen teilweise Techniken zum Einsatz, die so oder in ähnlicher Art und Weise auch im Kontext der Expertensuche gewinnbringend genutzt werden können. Auch wenn also keine direkte Verwendung dieser Werkzeuge möglich bzw. sinnvoll ist, so kann doch auf ihnen oder den ihnen zugrunde liegenden Ansätzen aufgebaut und an entsprechender Stelle notwendige Erweiterungen und Anpassungen vorgenommen werden.

Im Folgenden wird daher ein Lösungskonzept vorgeschlagen, das verschiedene im Information Retrieval bekannte Ansätze zur Grundlage nimmt, sie kombiniert und an zentralen Stellen um neue Konzepte erweitert. Das Ergebnis dieser Überlegungen ist die Expertensuchmaschine *EXPOSE* [KSJ07].

Im Rahmen des Forschungsprojektes *nova-net – Innovation in der Internetökonomie* wurden in Zusammenarbeit mit Partnern aus Forschung und Industrie die Anforderungen an eine solche Expertensuchmaschine analysiert. Ziel war es, mit möglichst geringem Aufwand Personen, Institutionen und Netzwerke zu identifizieren, die sich auf einem der Suchmaschine als Eingabe zu spezifizierenden Themenbereich einen Namen gemacht haben und ggf. als externe Experten zur Lösung von Problemen angeworben werden können.

Auf methodischer Ebene wurde ein einfacher dreistufiger Prozess definiert, der eine typische Suche dieser Art abbildet und durch Softwarewerkzeuge unterstützt werden kann (Abbildung 4.2):

1. Spezifikation des Informationsbedürfnisses
2. Suche nach relevanten Dokumenten im WWW
3. Identifikation und Bewertung von auf diesen Dokumenten genannten potenziellen Experten

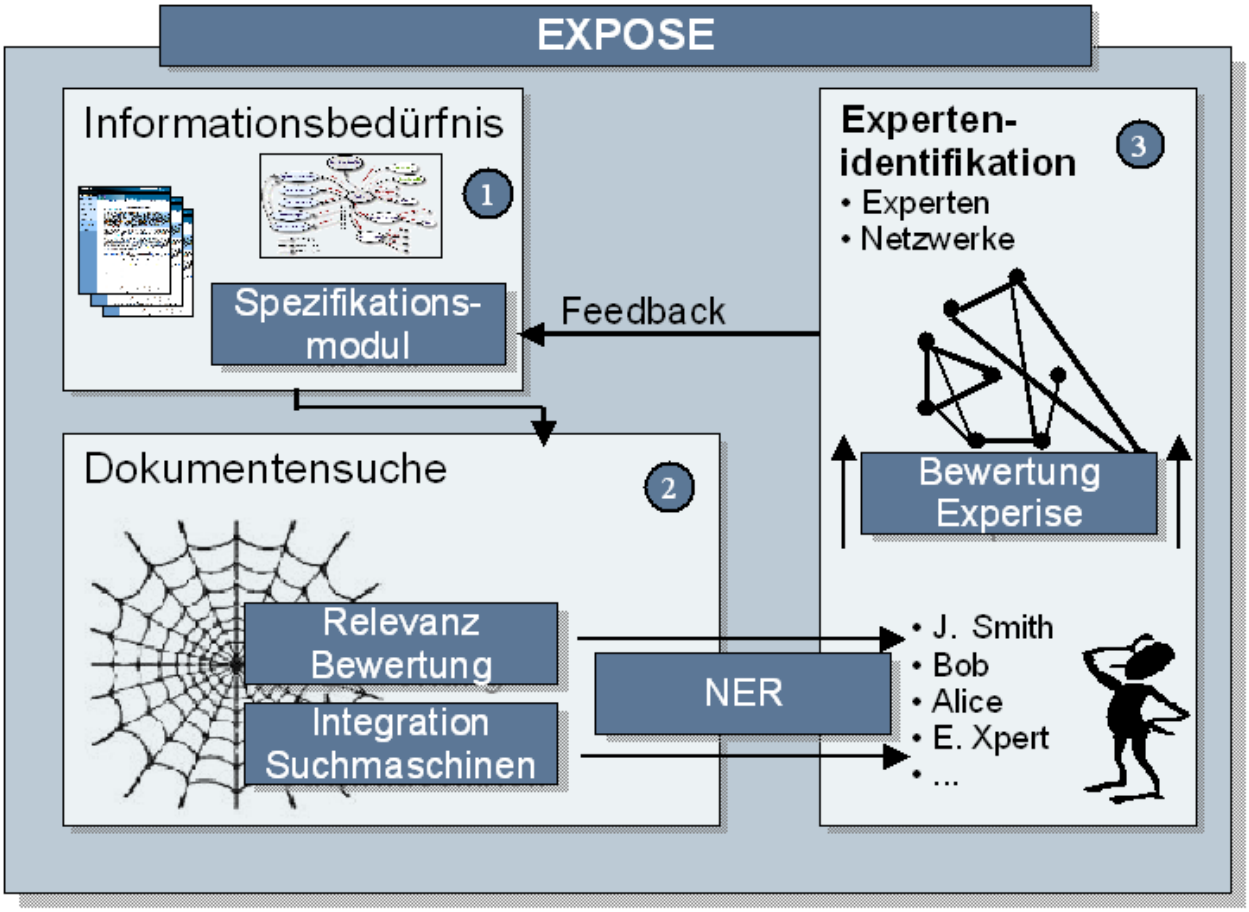

Abbildung 4.2: EXPOSE – EXPert Search Engine

Im Folgenden werden die einzelnen Schritte dieses Prozesses näher erläutert und aufgezeigt, wie sie durch Softwaremodule unterstützt werden können, welche wiederum auf der in Kapitel 4.1 vorgestellten Suchplattform basieren.

4.2.1 Schritt 1 – Spezifikation des Informationsbedürfnisses

Der erste Schritt bei der Suche nach Experten liegt in der genauen Spezifikation des Themengebietes, für das Experten gesucht werden sollen. Die Spezifikation einer Suchmaschinen-Anfrage ist eine Herausforderung, vor die jeder Benutzer herkömmlicher Web-Suchmaschinen gestellt wird. Im Detail besteht diese Herausforderung aus zwei Aspekten:

1. Wahl der richtigen Suchbegriffe
   Unterschiedliche Quellen benutzen ein teilweise unterschiedliches Vokabular um gleiche oder ähnliche Sachverhalte auszudrücken. Herkömmliche Suchmaschinen führen jedoch im Wesentlichen ein syntaktisches Matching zwischen Anfrage und potenziellen Ergebnissen durch. Wenn der Benutzer jedoch die in den Quellen verwendeten Vokabulare nicht explizit berücksichtigt, bspw. durch gezielte Angabe von Synonymen, dann werden im Allgemeinen viele relevante Quellen nicht zurückgemeldet oder bekommen eine schlechte Bewertung, was die Wahrscheinlichkeit erhöht, dass sie vom Benutzer nicht wahrgenommen werden.
2. Einfache Suchanfragen vs. komplexe Fragestellungen
   Wie in [JSP05] angeführt, spezifizieren die meisten Nutzer ihren Informationsbedarf anhand von sehr wenigen Suchbegriffen. Im Fall der Expertensuche kann jedoch von einem sehr speziellen und komplexen Informationsbedarf ausgegangen werden, da ansonsten die Frage nach externen Experten gar nicht gestellt würde. Dabei kann aber nicht erwartet werden, dass eine Suchspezifikation anhand von wenigen Suchbegriffen eine präzise Spezifikation für einen komplexen Sachverhalt darstellt. Auf unpräzise Suchspezifikationen kann jedoch im Umkehrschluss keine präzise Antwort erwartet werden. D. h., die von herkömmlichen Suchmaschinen zurückgemeldeten Ergebnisse sind in der Regel zu unpräzise und verlangen eine manuelle Nachbearbeitung. Aufgrund der geringen Präzision ist jedoch im Allgemeinen der Umfang der Ergebnisliste derart groß, dass eine solche manuelle Nachbearbeitung nicht effizient durchführbar ist.

Aus dieser Motivation heraus wurde im Rahmen der vorliegenden Arbeit ein Ansatz entwickelt, der die Spezifizierung des Informationsbedarfs für den Benutzer intuitiver gestaltet und damit die Komplexität dieses Schrittes senkt: anstatt der Formulierung einer Suchanfrage mittels Schlüsselworten, wird hier eine Beschreibung der gewünschten Suchergebnisse anhand von Beispieltexten eingesetzt [KSJ06] [KSJ07] .

Diese Beispieltexte können dabei prinzipiell aus beliebigen Dokumenten stammen, sollten aber einen starken Bezug zum gesuchten Thema besitzen. Die Reduzierung der Komplexität besteht dann darin, dass aus diesen Beispieltexten automatisch eine Suchspezifikation generiert wird bzw. diese Dokumente selbst als Spezifikation dienen. D. h. der Benutzer wird von der Aufgabe entbunden, eigenhändig von den Inhalten der Dokumente bzw. von seinem Informationsbedürfnis auf eine Menge von Suchbegriffen zu abstrahieren, die von herkömmlichen Suchmaschinen als Eingabe verlangt würden. Das Suchsystem übernimmt diese Aufgabe bzw. bietet Konzepte an, mittels derer die eingegebenen Dokumente direkt, also ohne eine Extraktion von Schlüsselbegriffen, für die Suche genutzt werden können. Für den Benutzer stellt sich das Verhalten des Systems derart dar, dass anhand der von ihm angegebenen Beispieldokumente weitere Dokumente gesucht werden, die eine *inhaltliche Ähnlichkeit* zu den Beispieldokumenten aufweisen. Für eine Diskussion des Begriffs „inhaltliche Ähnlichkeit“ und die Implementierung dieses Konzepts sei an dieser Stelle auf Kapitel 6 verwiesen.

Dass der Benutzer zum Start der Suche dem System Beispieldokumente als Eingabe liefern muss, provoziert offensichtlich ein sog. Henne-Ei-Problem: Es müssen genau solche Dokumente vorab vorhanden sein, wie sie auch in der folgenden Suche gefunden werden sollen. Untersuchungen bei Praxispartnern im Rahmen des Forschungsprojekts *nova-net* haben jedoch gezeigt, dass dies ein gangbarer Weg ist: in der Regel ist ein gewisser Dokumentensatz bereits verfügbar bzw. die Suche nach einigen wenigen themenrelevanten Dokumenten mittels herkömmlicher Suchmaschinen ist mit vertretbarem Aufwand durchführbar. D. h. es müssen zwar Dokumente bereits verfügbar sein, die den gesuchten Dokumenten inhaltlich ähneln, es genügt jedoch eine äußerst geringe Anzahl solcher Beispiele um in der folgenden Suche potenziell eine große Menge verwandter Dokumente zu finden. Dennoch wirft dieser Ansatz weitere Fragen auf:

1. Wie genau ist das bei der direkten Verwendung von Beispieldokumenten eingesetzte Ähnlichkeitsmaß auf natürlichsprachigen Texten definiert?

2. Die Eingabeform der Suchanfrage an herkömmliche Suchmaschinen ist die Form der Schlüsselwortanfrage und nicht wie im vorliegenden Fall benötigt, die Form von Beispieldokumenten. Wie muss eine Suchmaschine konzipiert sein, um mit einer solchen Anfrageform umgehen zu können und wie können existierende Suchmaschinen dennoch in den Suchprozess integriert werden, um deren Wissen über Inhalt und Struktur des WWW bestmöglich zu nutzen?

Problem (1) ist ein im Bereich Information Retrieval viel beachtetes Problem. In Kapitel 6 wird ein Ansatz zur Lösung dieses Problems vorgestellt und im Detail diskutiert. Dort wird auch definiert, was die *Ähnlichkeit* von Dokumenten in diesem Kontext genau bedeutet.

Problem (2) legt die Entwicklung einer eigenständigen Suchmaschine nahe. Diese soll jedoch bestehende Suchmaschinen zumindest teilweise integrieren können, um deren Wissen gewinnbringend weiter zu verwenden. In der folgenden Beschreibung des Schrittes 2 wird auf diese Problematik eingegangen und ein Konzept zu deren Lösung präsentiert.

4.2.2 Schritt 2 – Suche nach relevanten Dokumenten

Wie im vorangegangenen Abschnitt angeführt, sind herkömmliche Suchmaschinen nicht darauf ausgelegt, Suchanfragen in Form von Beispieldokumenten direkt zu verarbeiten. Soll mit diesem Prinzip gearbeitet werden, so müssen entweder die Beispieldokumente in eine Schlüsselwortanfrage und damit in eine für Suchmaschinen verständliche Form transformiert werden. Oder aber die Funktionalität einer solchen Suchmaschine muss teilweise angepasst werden, was aber im Allgemeinen nur durch eine Eigenimplementierung möglich ist. Im Rahmen dieser Arbeit wurden beide Ansätze verfolgt und mittels der im Kapitel 4.1 eingeführten Suchplattform implementiert. Sie werden im Folgenden diskutiert.

4.2.2.1 Transformation der Beispieldokumente in eine Schlüsselwortanfrage

Die Transformation eines oder mehrerer Dokumente in eine Schlüsselwortanfrage bedeutet im Wesentlichen eine Reduzierung der im Text verwendeten Begriffe auf die relevanten, den Text kennzeichnenden Schlüsselworte. Sie ist also vergleichbar mit der Verschlagwortung eines Textes. Die Herausforderung dieser Aufgabe besteht für Mensch wie Maschine gleichermaßen in der Bewertung der Relevanz einzelner Begriffe. Dabei ist diese Bewertung keine statische Abbildung von Begriffen auf Relevanzwerte, sondern dynamisch und abhängig vom jeweiligen (Dokumenten-)Kontext. Dieser Kontextbezug ergibt sich aus der Tatsache, dass ein Begriff in einem Dokument als zentraler Begriff auftreten kann, wohingegen er in einem anderen Dokument lediglich am Rande erwähnt wird. Während dieser Prozess der Identifizierung von relevanten Schlüsselworten beim Menschen häufig intuitiv geschieht, müssen für eine automatisierte Bewertung durch Software klare Berechnungsgrundlagen definiert werden. Hierzu existieren verschiedene Ansätze unterschiedlicher Komplexität und Ergebnisqualität. Im Rahmen von EXPOSE wurde ein verhältnismäßig einfacher Ansatz als Ausgangspunkt gewählt, der dann um intelligente Konzepte erweitert wurde.

Die grundlegende Idee besteht in der Nutzung von $tf \cdot idf$-Informationen (vgl. Kapitel 3.1.3.2) zur Selektion der kennzeichnenden Begriffe. Der $tf \cdot idf$-Wert wird für jeden der n verschiedenen im Text vorkommenden Begriffe bestimmt. Somit werden n Paare $(t, tfidf(t))$ gebildet. Anschließend werden diese Paare absteigend nach $tfidf(t)$ geordnet und die ersten k Begriffe, also die mit dem höchsten $tfidf(t)$, als Ergebnis ausgewählt. Im Falle von mehreren Dokumenten, wie im Szenario der Beispieldokumentenmenge, können die Ergebnismengen der einzelnen Analysen zu einer Gesamtmenge vereinigt werden. Alternativ lässt auch diese Gesamtmenge sich, ggf. nach einer Normierung der $tf \cdot idf$-Werte, erneut ordnen, um wieder die k Begriffe mit der höchsten Bewertung zu selektieren und somit eine obere Schranke für die Anzahl an Suchbegriffen zu garantieren, unabhängig von der Anzahl der Beispieldokumente.

Das Verfahren implementiert einen einfachen Ansatz zur Extraktion der rele-

vanten Begriffe eines Textes oder einer Textsammlung. Es bringt allerdings zwei Schwierigkeiten mit sich:

1. Zur Berechnung der $tf \cdot idf$-Werte muss ein Textkorpus in der dem Text zugrunde liegenden Sprache vorgehalten werden. Andernfalls könnte zwar die tf-Komponente berechnet werden, also die Häufigkeit der Nennung eines Begriffs im Text, nicht jedoch die df-Komponente, die ein Maß für die Kennzeichnungskraft des Begriffs in der zugrunde liegenden Sprache darstellt (vgl. Kapitel 3.1.3.2). Die Erstellung und Pflege eines solchen Korpus ist jedoch ein aufwändiges und mitnichten triviales Unterfangen, da der Korpus für allgemeine Suchen auch thematisch und sprachlich möglichst breit gefächert sein muss, um repräsentative statistische Kennzahlen über die verwendete Sprache zu erhalten.
2. Anhand dieses Verfahrens können nur die im Text vorkommenden Begriffe hinsichtlich ihrer Relevanz bewertet werden. Verwandte Begriffe, die den Text ebenso charakterisieren würden, können nicht identifiziert bzw. bewertet werden. Damit kann dieses automatisierte Verfahren nie die Qualität erreichen, die ein Mensch dank seiner Fähigkeit zur Abstraktion erreichen kann.

Eine Teilfunktionalität des in Kapitel 6 vorgestellte Ansatz zur Berechnung der Ähnlichkeit von Dokumenten kann auch hier eingesetzt werden, um der in Punkt 2 genannten Schwierigkeit zu begegnen. Dabei wird ein Text auf die ihn beschreibenden inhaltlichen Konzepte reduziert, die dann für eine Schlüsselwort-Anfrage genutzt werden können.

Bezug zur Suchplattform

Die Transformation von Dokumenten in eine Darstellung, die aus den kennzeichnenden Begriffen dieser Dokumente besteht, ist eine Funktionalität, die im Rahmen der Suchplattform an verschiedenen Stellen genutzt werden kann. Zum einen kann sie als Einstiegspunkt in die Suche mit herkömmlichen Suchmaschinen dienen, zum anderen aber auch zur Verfeinerung der Suche, wenn bereits erste passende Dokumente gefunden wurde, die dann zur Feinabstim-

mung der Suche herangezogen werden können. Dazu ist es hilfreich, wenn zwischen den einzelnen Darstellungen der aktuell benutzten Suchanfrage hin und her geschaltet werden kann. Im Rahmen der Suchplattform wurde die Transformationskomponente dahingehend umgesetzt, dass sie über die *QueryAPI* eine Koppelung an andere installierte Komponenten, wie bspw. den Schlüsselwortanfrage-Editor besitzt, und sich mit diesem automatisch synchronisiert. Das bedeutet, dass über die Transformation identifizierte Schlüsselworte automatisch in den Anfrage-Editor übernommen werden und dort vom Benutzer ggf. mittels Operatoren zusätzlich verknüpft werden können. Fügt der Benutzer im Anfrage-Editor zusätzliche Begriffe hinzu, so werden diese in der Transformationskomponente zwar nicht reflektiert, das Konzept der kanonischen Anfragerepräsentation (vgl. Kapitel 4.1.2.1) sorgt jedoch dafür, dass beide Komponenten Ihre eigene Sicht auf die Anfrage verwalten können und dort wo es technisch möglich und praktisch auch sinnvoll ist, eine Synchronisation stattfindet.

Ferner erlaubt die Suchplattform die Weitergabe dieser Form der Anfrage an alle im System registrierten Suchmaschinen-Schnittstellen. Diese können dann die jeweils benötigte bzw. verstandene Form der Anfrage nutzen, um eine Suche mit der zugrunde liegenden Suchmaschine zu starten.

Typische Suchen weisen keinen geraden Verlauf sondern eine hohe Verzweigung auf. Sie enden immer wieder in Sackgassen und erfordern eine Rückkehr zu älteren Zwischenständen, um die zu diesen Zwischenständen führenden Suchanfragen zu modifizieren und alternative Suchzweige zu untersuchen. Um diesem Fakt Rechnung zu tragen, werden alle Änderungen an der Suchanfrage in der von der Suchplattform verwalteten Datenbank protokolliert. Das erlaubt es dem Benutzer zu jedem Zeitpunkt zu bereits erreichten Zuständen zurück zu kehren und die Suche in eine andere Richtung zu lenken.

4.2.2.2 Focused Crawler

Aktuelle Suchmaschinen unterstützen die Eingabe der Suchspezifikation anhand von Beispieldokumenten nicht direkt sondern lediglich über den oben vorgestellten Transformationsansatz. Um diese Art der Spezifikation dennoch direkt in einer Suchmaschine verwenden zu können, und um ggf. den Verlauf der Suche besser kontrollieren und steuern zu können, bedarf es daher der Entwicklung einer eigenen Suchmaschinen-Komponente. Die Verwendung der Indexe existierender Suchmaschinen wäre dabei ein wünschenswertes Ziel, das jedoch schwer zu erreichen ist. In der Regel werden die Indexe der existierender Suchmaschinen von den Betreibern nur über die gewohnten Schnittstellen der Suchmaschine, wie bspw. ein Web-Interface, zugänglich gemacht. Für eine direkte Berechnung der Ähnlichkeit von Dokumenten, ist die Mächtigkeit dieser Schnittstellen jedoch zu eingeschränkt. Die Erzeugung eines eigenen Index für signifikante Teile des WWW ist aber in der Praxis auch nicht realisierbar, da hierfür hohe Kapazitäten an Netzwerkbandbreite, Speicherplatz und Rechenleistung nötig wären.

Ein sinnvoller Kompromiss besteht nun darin, zwar mit einem eigenen Crawler das WWW zu durchsuchen, sich dabei jedoch lediglich auf diejenigen Bereiche des WWW zu konzentrieren, die sich mit den in der Suchanfrage spezifizierten Themen beschäftigen. In [CvdBD99] wurde hierzu das Konzept des *Focused Crawlers* entwickelt, der genau dies ermöglicht. Eine Diskussion dieses Konzeptes findet sich in Kapitel 3.1.1.

Die zentralen Merkmale eines solchen Crawlers sind die folgenden:

1. Ein Focused Crawler indexiert das WWW nicht vor der eigentlichen Suchanfrage, sondern erst auf Anweisung des Benutzers. Dabei lädt er nach bestimmten Heuristiken Dokumente aus dem WWW und berechnet deren Relevanz für die Suchanfrage des Benutzers. Der Einsatz solcher Heuristiken ist notwendig, da der Crawler aus einer unbekannten Dokumentenmenge mit möglichst wenigen Zugriffen die größtmögliche Teilmenge aller relevanten Dokumente selektieren muss.

2. Eine solche Suche dauert im Unterschied zu Suchen mit herkömmlichen Suchmaschinen nicht wenige Millisekunden sondern eher Minuten bis Stunden, da nicht auf einem vorbereiteten und optimierten Index, sondern auf vollständigen Dokumenten im WWW gesucht werden muss.
3. Da der Crawler auf den tatsächlichen Dokumenten operiert und nicht auf einen in seinem Informationsgehalt gegenüber diesen Dokumenten reduzierten Index zurück greift, sind detailliertere Analysen möglich. Diese Analysen können speziell auf das jeweilige Suchproblem zugeschnitten sein und damit ungleich mehr Kontextwissen in die Suche einbringen, als dies bei Suchen mit herkömmlichen Suchmaschinen möglich wäre.

Auf diese Art und Weise kann ein im Vergleich zu herkömmlichen Suchmaschinen deutlich langsamerer Focused Crawler dennoch einen Teil dieses Mankos durch Ergebnisse von potenziell höherer Qualität kompensieren.

Hier setzt auch das Konzept der Spezifikation von Suchanfragen anhand von Beispieldokumenten an: Anhand des in Kapitel 6 vorgestellten Ähnlichkeitsmaßes für Dokumente kann bewertet werden, wie gut ein heruntergeladenes Dokument zur Suchanfrage passt. Die Möglichkeit eine Anfrage mittels Suchbegriffen zu spezifizieren, wird dadurch nicht eingeschränkt, sondern um ein mächtiges Konzept erweitert.

Bezug zur Suchplattform

Der Focused Crawler integriert sich dahingehend in die Suchplattform, dass er eine Suchmaschinen-Schnittstelle bereit stellt und somit von allen anderen Komponenten über diese Schnittstelle angesprochen werden kann. Neben der Anfragesyntax für herkömmliche Suchmaschinen kann er zusätzlich die Form der Spezifikation durch Beispieldokumente interpretieren und diese Suchen ausführen. Über die *ResultAPI* liefert der Focused Crawler wie jede andere eingebundene Suchmaschine auch seine Ergebnisse an das System zurück. Diese stehen damit anderen Komponenten, wie bspw. Komponenten zur Auswertung von Hyperlink-Graphen, zur Verfügung.

4.2.2.3 Nutzung herkömmlicher Suchmaschinen

Wie oben ausgeführt, kann ein Focused Crawler eingesetzt werden, um die Suche anhand detaillierter Textanalysen genauer zu steuern. Ein großes Problem für solche Crawler stellt allerdings die Tatsache dar, dass sich im Web zwar oft stark vernetzte Inseln aus thematisch verwandten Dokumenten bilden, diese Inseln jedoch mit anderen solchen Inseln zum selben Thema deutlich schwächer vernetzt sind. Da ein Focused Crawler sich aber dadurch auszeichnet, dass er nur denjenigen Teil des zu einem bestimmten Zeitpunkt noch nicht besuchten Web durchsucht, der direkt an bereits besuchte und für relevant erachtete Ressourcen grenzt, wird er weiter entfernte Themeninseln oftmals nicht erreichen können [DCL+00]. Dieser Schwierigkeit lässt sich durch die Integration herkömmlicher Suchmaschinen in den Lauf des Focused Crawlers begegnen. Im Folgenden werden zwei Ansätze erläutert, die das Wissen herkömmlicher Suchmaschinen über Inhalt und Struktur des WWW für den Focused Crawler nutzbar machen.

Rückwärtsverweise Das Crawling von Dokumenten folgt dem Schema, dass ausgehend von einer Menge an Start-URLs die von diesen URLs bestimmten Dokumente geladen werden. Es werden die darin enthaltenen Verweise extrahiert und ggf. ebenfalls geladen. Dieser Prozess setzt sich rekursiv fort. Das bedeutet aber, dass nur solche Dokumente gefunden werden können, auf die von bereits besuchten Dokumenten verwiesen wurde, andernfalls wäre ihre URL unbekannt und ein Zugriff auf sie nicht möglich.

Problematisch dabei ist, dass sich für einen Crawler die Suche im WWW wie eine Suche auf einem gerichteten Graphen darstellt: Das WWW kann dabei als ein gerichteter Graph interpretiert werden, dessen Knoten Dokumente repräsentieren und dessen Kanten die in den Dokumenten enthaltenen Hyperlinks darstellen. Die Eigenschaft des Graphen, gerichtet zu sein, bedeutet, dass auch wenn von einem Dokument A aus (siehe Abbildung 4.3) ein weiteres Dokument B mittels eines Verweises E_{ab} erreichbar ist, ein Rückwärtsverweis E_{ba} nicht notwendigerweise existieren muss. Das wiederum bedeutet, dass ein von A

kommender Crawler zwar B über E_{ab} erreichen könnte; käme er jedoch von B, so würde er A nicht finden, da in Dokument B kein Verweis auf Dokument A enthalten ist. Für die Suche bzw. den Wunsch des Benutzers nach einer möglichst umfassenden Ergebnismenge, stellt dies offensichtlich ein Problem dar: zwar sind evtl. beide Dokumente A und B relevant und zudem noch miteinander über einen Verweis verknüpft; dennoch wird Dokument A nicht als Ergebnis zurück gemeldet und der Benutzer erfährt nichts über dessen Inhalt. Da der Focused Crawler im Vergleich zu den Crawlern herkömmlicher Suchmaschinen aus Zeitgründen nur einen verhältnismäßig kleinen Teil des Web durchsuchen kann, kann auch nicht generell davon ausgegangen werden, dass er „über Umwege“ irgendwann auf Dokument A stößt.

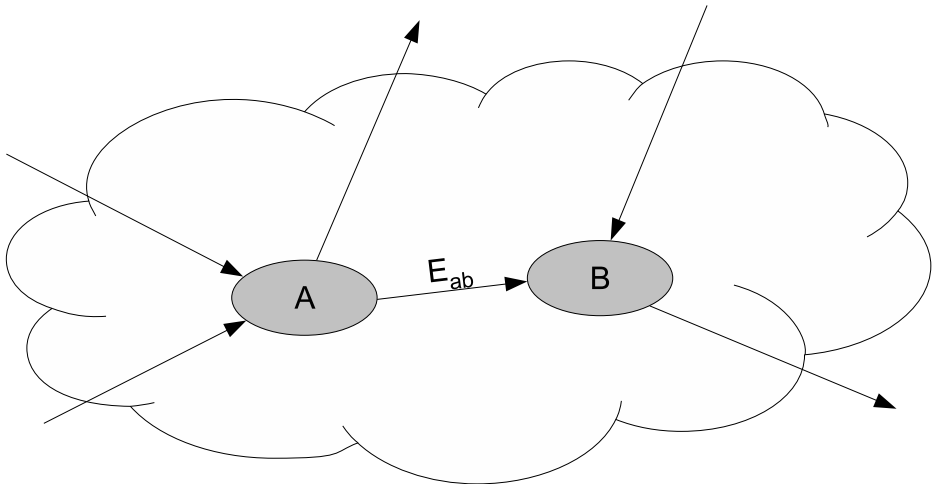

Abbildung 4.3: Verknüpfung von Webressourcen – Rückwärtsverweise

Eine Sicht auf das WWW als ungerichteten Graphen wäre in diesem Zusammenhang daher ein signifikanter Fortschritt, denn dann könnte Dokument A auch von Dokument B aus mit einem Schritt gefunden und analysiert werden. Offensichtlich ist für die Erstellung einer solchen Sicht aber umfangreiches Wissen über das gesamte WWW von Nöten, was dem Prinzip eines Focused Crawlers jedoch entgegen steht.

An diesem Punkt kommen herkömmliche Suchmaschinen ins Spiel. Neben der Indexierung des reinen Dokumenteninhalts speichern diese Systeme auch Informationen zum Hyperlink-Kontext einer Webressource, da diese Informa-

tionen eine signifikante Eingabe für das Ranking von Suchergebnissen liefern. Verschiedene Suchmaschinen machen dem Benutzer Teile dieser Informationen über die Nutzung spezieller Anfragen zugänglich. So können bspw. bei großen Suchmaschinen wie Google oder MSN so genannte *Backlinks* einer Webressource abgefragt werden, was für Dokument B u. a. der Kante E_{ab} entspricht. Damit lassen sich also nicht nur diejenigen Dokumente finden, die aus dem bereits besuchten Teil des WWW heraus verlinkt sind, sondern auch diejenigen, die lediglich auf den bereits besuchten Teil verweisen. Für hochgradig relevante Dokumente empfiehlt sich daher eine Untersuchung der Hyperlink-Nachbarschaft auf diese Art und Weise, um schnell zu weiteren relevanten Dokumenten zu gelangen.

Suche nach ähnlichen Dokumenten Um Dokumente zu finden, die nicht über für die Focused Crawler notwendigerweise kurzen Verweispfade verbunden sind, wie bspw. das Dokument C mit den Dokumenten A oder B in Abbildung 4.4, müssen zusätzliche Informationsquellen erschlossen werden.

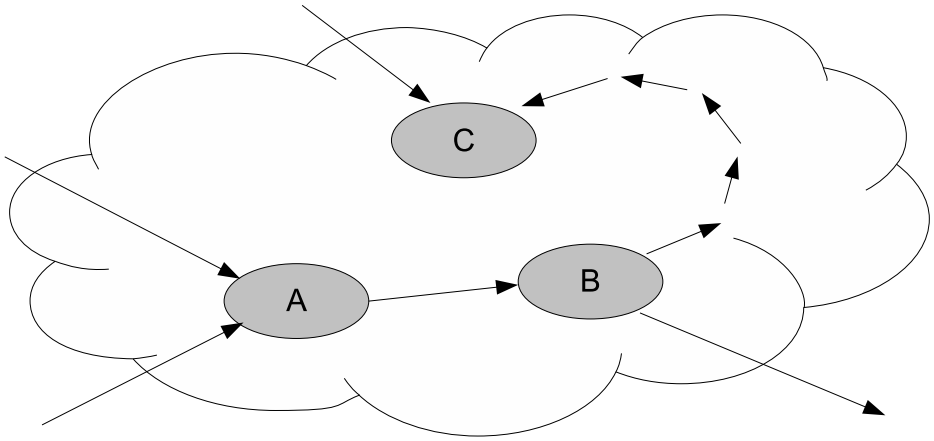

Abbildung 4.4: Verknüpfung von Webressourcen – Ähnliche Dokumente

Herkömmliche Suchmaschinen können als eine solche Quelle genutzt werden. Mittels spezieller Schlüsselworte bieten sie oftmals die Möglichkeit nach Dokumenten zu suchen, die einem bereits gefundenen Dokument ähneln. Dabei können diese Suchmaschinen zur Beantwortung einer solchen Anfrage auf ihren gesamten Index zurück greifen und nicht wie im Falle des Focused Crawler, nur

auf die wenigen bereits besuchten Dokumente. Die Nutzung dieser Funktionalität bietet somit einen ersten Schritt zur Erweiterung der Sicht des Focused Crawlers.

Allerdings liegen im Allgemeinen keine Informationen darüber vor, wie das Kriterium der „Ähnlichkeit“ in der jeweiligen Suchmaschine implementiert wurde. Nahe liegend ist, dass in erster Linie syntaktische Aspekte für diese Berechnung herangezogen werden. Aber auch strukturelle Komponenten werden in der Literatur diskutiert, bspw. in [DH99]. In jedem Fall jedoch kann von außen kein Einfluss auf diese Berechnung genommen werden, was keine eindeutige Definition des Konzeptes der „Ähnlichkeit“ erlaubt und insbesondere keine Möglichkeiten bietet, eigene Berechnungsvorschriften anzuwenden.

Eine größere Einflussnahme erlaubt ein Ansatz, der dem in Kapitel 4.2.2.1 vorgestellten Ansatz zur Schlüsselwort-Identifikation folgt. Dort wurden mittels des $tf \cdot idf$-Konzepts die einen Text kennzeichnenden Begriffe identifiziert. Diese Methode bzw. die aus deren Anwendung resultierenden Schlüsselworte lassen sich auch hier als Anfrage an eine Suchmaschine nutzen, um weitere Dokumente zu finden, die den durch diese Begriffe beschriebenen Sachverhalt behandeln.

Offensichtlich liefert dieses Verfahren jedoch keine perfekten Ergebnisse im Sinne von Dokumenten, die den ursprünglichen Dokumenten „ähneln“. So ist der in Kapitel 4.2.2.1 vorgestellte Ansatz zur Ermittlung kennzeichnender Begriffe anfällig für Fehler und Ungenauigkeiten. Bspw. ist zur Berechnung der idf-Werte ein verhältnismäßig großer Referenzkorpus notwendig, um eine belastbare Aussage über die Häufigkeit der Verwendung einzelner Begriffe im allgemeinen Sprachgebrauch machen zu können. Ist der verwendete Korpus jedoch thematisch nicht hinreichend breit gefächert, so treten hier entsprechende Ungenauigkeiten auf. Auch stellt dieses Verfahren einen rein syntaxbasierten Ansatz dar. Beschreiben zwei Dokumente den selben Sachverhalt mit unterschiedlichen Worten, so kann diese Ähnlichkeit anhand des hier vorgestellten Verfahrens nicht in vollem Umfang ermittelt werden. Dazu bedarf es aufwändigerer Ansätze, wie bspw. des in 6 vorgestellten Verfahrens der Berechnung

von Dokumentenähnlichkeit anhand sog. „konzeptueller Kontexte“. Dennoch stellt dieser verhältnismäßig einfache Ansatz einen ersten Schritt dar, der die Integration herkömmlicher Suchmaschinen in den Focused Crawler ermöglicht und somit über die Nutzung deren Wissens die Erschließung anderweitig nicht auffindbarer Quellen erlaubt.

Bezug zur Suchplattform
Die Integration herkömmlicher Suchmaschinen mit dem Konzept des Focused Crawlers erlaubt die Nutzung deren Wissens über Inhalt und Struktur weiter Teile des WWW. Da der Focused Crawler eine an die Problemstellung angepasste Analysestrategie verfolgen kann, jedoch nicht über die umfassende Sicht der Suchmaschinen auf das WWW verfügt, wird er durch die herkömmlichen Suchmaschinen ideal ergänzt. Die Suchplattform bietet für diese Art der „Zusammenarbeit“ dahingehend Unterstützung, dass entsprechende Schnittstellen für die einzelnen Systeme vorgesehen sind. So existiert eine generische Schnittstelle, über die Suchmaschinen – sowohl herkömmliche als auch der Focused Crawler – angebunden werden können. Sie werden über die *QueryAPI* an die aktuelle Suchanfrage gekoppelt und erhalten darüber die für sie verständliche Form der Anfrage. Über die *ResultAPI* erfolgt die Rückmeldung der Suchergebnisse für die Weiterverarbeitung durch andere Komponenten oder den Benutzer.

4.2.3 Schritt 3 – Identifikation von Experten

Das WWW stellt ein Netz von Dokumenten mit in aller Regel natürlichsprachigen Texten dar und nicht ein Netz aus strukturierten Informationen, die automatisiert verarbeitbar und analysierbar sind. Daher können auch keine direkten Anfragen an das WWW bzw. seine Suchmaschinen gestellt werden im Sinne von bspw. „finde Experten zum Thema Brennstoffzelle“. Statt dessen müssen Dokumente gefunden werden, die das gewünschte Thema behandeln und diese Dokumente dann auf die Nennung von Personen hin untersucht werden, die auf diesem Themengebiet eine vermeintliche Expertise besitzen.

In Abbildung 4.2 wird das in Schritt 3 dargestellt. Die Schwierigkeit bei dieser Aufgabe liegt in den Teilaspekten *Identifikation von Personennennungen* (NER) und der *Bewertung deren Expertise* auf dem jeweiligen Themengebiet.

4.2.3.1 Identifikation von Personennennungen

Das Problem der Identifikation von Personennennungen in natürlichsprachigen Texten soll an dieser Stelle lediglich im Überblick diskutiert werden. Eine detaillierte Analyse des Problems sowie die Ausarbeitung und Analyse eines Lösungsvorschlages erfolgen in Kapitel 7.

Während es dem menschlichen Leser eines Textes in der Regel nicht schwer fällt, die Nennung von Entitäten wie Personen, Firmen, Orten, Telefonnummern etc. in einem natürlichsprachigen Text zu identifizieren, so ist diese Aufgabe für automatisierte Verfahren eine ungleich größere Herausforderung, da für die fehlerfreie Erkennung solcher Nennungen ein umfangreiches Sprachverständnis erforderlich ist. Die Komplexität der Erkennung steigt zudem, wenn auch unbenamte Referenzen auf diese Entitäten berücksichtigt werden, bspw. in Form von Personalpronomen. Letzteres ist insbesondere notwendig wenn Textpassagen bzw. deren Inhalt im Rahmen der weiter unten diskutierten Bewertung mit den identifizierten Personen in Verbindung gebracht werden soll („G. Daimler arbeitete in Stuttgart. Dort entwickelte *er* den Einzylinder-Viertaktmotor“).

Das Forschungsgebiet NLP (Natural Language Processing) und darin insbesondere die NER (Named Entity Recognition) beschäftigt sich mit diesem Thema und bietet vielfältige Konzepte zur Lösung des Problems. Wie in Kapitel 7 dargestellt, sind diese Ansätze jedoch meist nicht mit zufriedenstellender Ergebnisqualität auf allgemeine und a priori unbekannte Dokumente bzw. Dokumententypen anwendbar. Es kann also durchaus sinnvoll sein, die neuesten Erkenntnisse aus dem Bereich der NER-Forschung nicht umzusetzen und statt dessen eher auf weniger ausgefeilte Techniken zu setzen, die jedoch in diesem Szenario stabile Ergebnisse erwarten lassen [PD97] [MMG99]:

- Anhand einer Datenbank mit Vor- und Nachnamen sowie zusätzlichen Regeln für die Bildung von Namensnennungen lassen sich ein Großteil aller direkt benannten Personen erkennen. Beispiele hierfür wären „Gottlieb Daimler“ oder durch Anwendung verschiedener Transformationsregeln auch „Daimler, Gottlieb“ bzw. „G. Daimler“.
- Eine solche Datenbank wird immer unvollständig sein, weil bspw. Namen aus anderen Sprachen übernommen oder neue Schreibweisen eingeführt werden. Auch lassen sich Nennungen lediglich des Vor- oder Nachnamens in der Regel nicht auf diese Art und Weise als Personennennungen identifizieren, da aufgrund von Mehrdeutigkeiten die Fehlerrate stark ansteigen würde. Ein Beispiel hierfür ist „Bei Daimler in Stuttgart . . . “; hier kann aus der Tatsache, dass „Daimler“ in der Namensdatenbank als Nachname geführt wird nicht darauf geschlossen werden, dass auch Herr oder Frau Daimler gemeint sind. Ähnliches gilt für andere mehrdeutige Namen wie „Metzger“ oder „Müller“. Um die Nennung dieser – zumindest teilweise – treffsicher zu erkennen, können verschiedene erweiterte Regeln angewandt werden. Bspw. „<potenzieller Nachname|Vorname> ’sagte’|’meinte’|. . . “ oder „’Herr’|’Frau’|’Dr.’|. . . <potenzieller Nachname>“. Der vermeintliche Name wird also dann als tatsächlicher Name erkannt, wenn die Nennung in einem entsprechenden sprachlichen Kontext erfolgt.

Zusammen mit der genannten Datenbank aus Vor- und Nachnamen lassen sich diese Regeln in einen deterministischen Automaten überführen, der einen Text nach Nennungen von Personen durchsuchen kann. In Kapitel 7 werden weitere Verbesserungen gegenüber diesem einfachen Ansatz näher diskutiert.

4.2.3.2 Bewertung der Expertise

Die bspw. anhand des oben vorgestellten Ansatzes identifizierten Nennungen von Personen oder Institutionen werden nun auf ihre Eignung als Suchergebnis für die Expertensuche hin untersucht. D. h. die genannten Personen/Institutionen werden als potenzielle Experten angesehen und deren vermeintliche

Expertise auf dem betreffenden Themengebiet bewertet. Die Nennung einer Person/Institution in einem Text kann dabei in vier Ausprägungen erfolgen:

1. Die Person ist Autor des Textes
2. Die Person ist aktiv oder passiv an einer vom Text behandelten Diskussion beteiligt, somit ein Spezialfall von 1)
3. Die Person bzw. ihre Arbeit wird im Text referenziert
4. Die Person wird erwähnt, obwohl sie mit dem Thema nichts zu tun hat

Die Fälle 1-3 fördern die Annahme, die Person sei Experte auf dem betrachteten Themengebiet. In Fall 4 jedoch kann davon nicht ausgegangen werden. Das zeigt, dass allein anhand der Nennung einer Person bzw. der Häufigkeit ihrer Nennung im Kontext eines Themas noch keine qualifizierte Aussage über ihre Expertise auf diesem Themengebiet gemacht werden kann.

Exakte Angaben über die Expertise einer Person oder zumindest darüber, was aus den betrachteten Dokumenten über deren Expertise herausgelesen werden kann, erfordert ein umfassendes Sprachverständnis. Dies kann nicht als gegeben angesehen werden, da der maschinellen Sprachverarbeitung auch heute noch enge Grenzen gesetzt sind. Insbesondere im Kontext des WWW mit seinen hochgradig inhomogenen Texten und Texttypen, ist die Analyse von natürlichsprachigen Texten für Computersysteme nicht mit hinreichender Präzision realisierbar. Absolute Aussagen, die nicht mehr überprüft werden müssen, können also nicht getroffen werden.

Im Rahmen der Expertensuchmaschine *EXPOSE* wurden daher verschiedene Kriterien identifiziert, die jede für sich zwar die Expertise einer Person nicht hinreichend genau widerspiegeln, die zusammen betrachtet jedoch einen qualifizierten Hinweis auf die Expertise liefern. Zur Bestimmung der Expertise einer Person werden sie einzeln ausgewertet, normalisiert und zu einem Gesamtindikator zusammengefasst. Im Einzelnen handelt es sich dabei um folgende Kriterien:

1. Häufigkeit der Nennung einer Person
   Als erster Anhaltspunkt wird die Häufigkeit der Nennung einer Person

auf den als relevant erachteten Dokumenten betrachtet. Jede Nennung wird dabei gemäß der Relevanz des nennenden Dokuments gewichtet. Je öfter also eine Person genannt wird und je höher die Relevanz der sie nennenden Dokumente ist, desto stärker wird dieser Indikator eine vorhandene Expertise anzeigen.

2. Kontext der Nennung einer Person
   Alleine aus der Tatsache, dass eine Person im Zusammenhang mit dem gesuchten Thema genannt wird bzw. aus der Anzahl der Nennungen einer Person kann jedoch nicht auf deren Expertise geschlossen werden. Bspw. ist auf den Webauftritten wissenschaftlicher Konferenzen oftmals eine Übersicht aller Autoren, zusammen mit den Abstracts der Veröffentlichungen verfügbar. Wenn nun in einem solchen Fall ein oder mehrere der Abstracts zum Thema der Suche passen so wären damit automatisch alle in der Übersicht genannten Personen über den Häufigkeitsindikator gleich hoch bewertet. Es erhielten also nicht nur diejenigen eine hohe Wertung, die tatsächlich als Autor der relevanten Texte genannte sind.
   Da jedoch die Bedeutung eines Textes ebenso wie die generelle Struktur eines Dokuments nicht allgemein ermittelt werden können, ist es ohne spezielles Kontextwissen nicht möglich, zu erkennen, dass es sich bei der o. g. Übersicht um eben jene Paarung $(Autor(en), Abstract)$ handelt. Genau dieses Kontextwissen ist aber von Nöten, um eine zuverlässige Aussage über diese Zuordnung automatisiert erhalten zu können. Im vorliegenden Fall kann es bspw. über spezielle Konnektoren oder Wrapper eingeführt werden, wie sie in Kapitel 5 diskutiert werden. Sind derartige Konnektoren nicht verfügbar, dann liegt der Einsatz von Heuristiken nahe, die die Struktur des Textes in den meisten Fällen zumindest ansatzweise erkennen helfen. Dabei haben sich folgende Ansätze als zielführend heraus gestellt: i) Personen, die in unmittelbarer textueller Nachbarschaft zu relevanten Schlüsselworten stehen (vgl. auch [SLWM04] sowie oben zum Thema Extraktion von Schlüsselworten), bekommen bessere Bewertungen. ii) Personen, die bezogen auf das Layout eines Dokuments in Randbereichen genannt werden, bekommen eine schlechtere Bewertung

als Personen, die an zentraler Stelle genannt werden. Motiviert wird diese Heuristik durch das im Web weit verbreitete Vorhandensein von Navigtationsstrukturen, Angaben zu Webmastern etc. am Layout-Rand eines Dokuments. Untersuchungen [SMS+08] im Rahmen praktischer Einsätze im Projekt *nova-net* haben gezeigt, dass die an diesen Stellen genannten Personen mit dem gesuchten Thema selten in einem inhaltlichen Bezug stehen und statt dessen durch deren Berücksichtigung in der Bewertung die Qualität der Ergebnisse abnimmt. iii) Insbesondere bei Texten, in denen viele Personennennung identifiziert werden, hat sich eine Aufteilung des Textes in kleinere Einheiten als sinnvoll herausgestellt. In diesen Fällen handelt es sich oftmals um Übersichtsseiten wie die o. g. Publikationsliste einer wissenschaftlichen Konferenz. Wird nun der Text bspw. in einzelne Abschnitte aufgeteilt und werden diese Abschnitte separat betrachtet, dann kann ein bessere Zuordnung von Personen zu relevanten Textabschnitten durchgeführt werden, wodurch letztlich die Qualität der gesamten Bewertungen signifikant gesteigert werden kann. Dieser Ansatz ist ähnlich zu dem in i) genannten, erlaubt jedoch eine erweiterte Betrachtung von Textstrukturen, wie ggf. die Erkennung von Kapiteln der Abschnitten.

3. Vernetzung der identifizierten Person
   Ein weiterer Indikator für die Expertise einer Person ist deren Vernetzung mit anderen Experten. Untersuchungen im Rahmen des *nova-net*-Projekts haben gezeigt, dass die Interaktion oder Kommunikation einer Person mit anderen bereits als Experten identifizierten Personen oftmals einen Rückschluss auf die Expertise eben dieser Person erlaubt [KSJ06]. Erklärt werden kann dies mit mehreren Überlegungen: i) Personen, die sich zu einem Thema austauschen, haben oftmals einen ähnlichen Wissensstand. In diesem Fall kann darauf geschlossen werden, dass wenn eine Person bereits als Experte identifiziert wurde, auch die andere Person eine gewisse Expertise besitzt. ii) Es wird ein Wissenstransfer von einer bereits als Experte identifizierten Person hin zu einer anderen Person unterstellt.
   Diese Überlegungen treffen nicht nur auf direkte Kommunikation zwi-

schen Personen, wie bspw. im Rahmen von Webforen oder Newsgroups zu, sondern können prinzipiell auch auf andere Texte übertragen werden, in denen die Personen lediglich gemeinsam genannt werden. Hinter diesen Überlegungen steht die Idee, dass die bereits als Experte identifizierte Person der aktuell betrachteten Person quasi eine Referenz erstellt. Die Referenz einer hoch bewerteten Person hat dabei für die Weitergabe ein höheres Gewicht als die Referenz einer niedriger bewerteten Person. Dies ist analog zum Ranking von Web-Suchmaschinen-Ergebnissen, bspw. anhand des PageRank-Algorithmus [BP98], wo die Bewertung von Webressourcen dann steigt, wenn sie von anderen, bereits hoch bewerteten Ressourcen aus verlinkt sind.

4. Verknüpfung zwischen Personen und Dokumenten
   Schließlich kann analog zur o. g. Personenverknüpfung auch noch die Verknüpfung der Dokumente als weiterer Indikator zur Bewertung der Expertise einer Person heran gezogen werden. Personen, die auf Dokumenten genannt werden, welche stark mit anderen Dokumenten verknüpft sind – insbesondere mit als relevant erachteten Dokumenten oder mit Dokumenten, die andere hoch bewertete Experten benennen – erhalten ebenfalls eine höhere Bewertung. Motivation hierfür ist, dass, gemeinsam mit dem zuvor genannten Punkt der Verknüpfung von Personen, auf diese Art und Weise Wissensnetzwerke abgebildet werden können.

Zu beachten ist dabei, dass diese Bewertungen von Personen und Dokumenten nicht in einem Schritt erfolgen kann, sondern dass die Ergebnisse einer Bewertung Einfluss auf die Bewertung selbst haben können. Dies wird deutlich am Beispiel einer Person, die auf mehreren Dokumenten identifiziert wurde, u. a. als Autor einer relevanten Veröffentlichung. Damit bekommen automatisch alle anderen Dokumente, die diese Person nennen auch eine höhere Bewertung. In der Folge werden aber auch diejenigen Personen höher bewertet, die auf diesen weiteren Dokumenten genannt werden. Offensichtlich muss die Weitervergabe von Gewichten in jedem weiteren Schritt eingeschränkt werden, da sich diese ansonsten beliebig rekursiv fortsetzen würde. Dennoch wird klar, dass das Ergebnis einer Bewertung Einfluss auf weitere Bewertungen haben

kann und dass sich dies in einem bestimmten Rahmen rekursiv fortsetzt. Dieses Prinzip macht eine Berechnung der Expertise-Bewertungen auf eine iterative Art und Weise notwendig, auch bedingt durch die Tatsache, dass i. d. R. die Präsentation von (Teil-)Ergebnissen schon während des Suchlaufs erfolgen soll und nicht erst nach dessen Abschluss. Ansätze, wie dies erreicht werden kann, finden sich bspw. im PageRank-Algorithmus oder allgemeiner im Algorithmus für Spreading Activation [Cre97].

Als besonders wichtig für die Akzeptanz einer diese Ansätze implementierenden Software durch den Benutzer und damit für deren praktischen Einsatz haben sich die Präsentation der Ergebnisse und die Möglichkeit der Einflussnahme durch den Benutzer herausgestellt (vgl. bspw. Abbildung 4.5). Die o. g. Indikatoren stellen lediglich Anhaltspunkte für die Bewertung der Expertise einer Person dar. Sie unterliegen aufgrund des im Allgemeinen fehlenden Textverständnisses der Software dabei einer relativ großen Unsicherheit. Die Ergebnisse können lediglich eine erste grobe Einschätzung liefern, die jedoch durch den Benutzer verifiziert bzw. verfeinert werden muss. Dazu ist es nötig, die Ergebnisse wo möglich grafisch aufzubereiten und auf Anforderung die der Berechnung zugrunde liegenden Daten zu präsentieren, um die Bewertung für den Benutzer transparent zu machen. Hilfreich ist zudem, dem Benutzer die Möglichkeit zu geben, einzelne Bewertung oder Indikatoren in ihrer Einflussschwere zu begrenzen oder diese stärker zu gewichten.

Bezug zur Suchplattform

Bei der Suche nach Experten handelt es sich um ein spezielles Problem, das die Komplexität einfacher Suchen deutlich übersteigt. Die Suchplattform in ihrer bisher beschriebenen Form bietet zur Unterstützung des Benutzers bei diesem Problem in erster Linie Grundfunktionalität wie die Integration verschiedener Suchmaschinen. Der eigentliche Mehrwert entsteht aber durch die Modularität und Erweiterbarkeit der Plattform. So können speziell auf das Problem der Expertensuche zugeschnittene Analysekomponenten einfach über die jeweiligen APIs integriert werden. Bspw. wurde eine Komponente entwickelt, die basierend auf den Verknüpfungen von Personen bzw. Ressourcen und deren Relevanz-

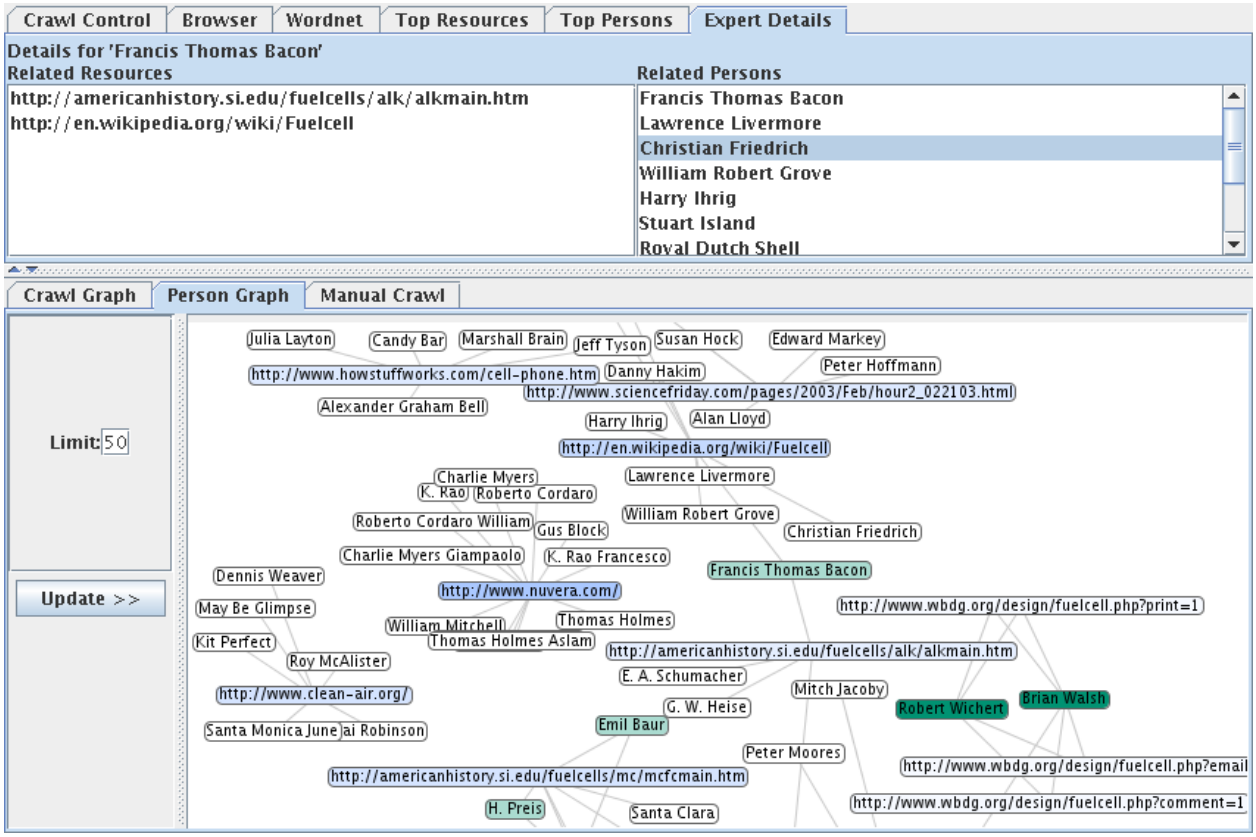

Abbildung 4.5: Teilergebnis einer Suche: Personen und deren Fundstellen im Netz

bewertung anhand eines Spreading-Activation-Algorithmus einen erweiterten Relevanz-Wert berechnet. In die *Präsentationsebene* wurden Darstellungskomponenten integriert, die die Vernetzung von Personen und Dokumenten darstellen und somit dem Benutzer die Steuerung des Prozesses und die Auswahl von weiter zu verfolgenden Suchzweigen erleichtern (vgl. Abbildung 4.5).

4.3 Zusammenfassung

In den zurückliegenden Abschnitten wurde zunächst diskutiert worin die Vorteile einer universellen Suchplattform bestehen. Diese sind zum einen zu sehen in einer durchgängigen Unterstützung des Benutzers bei der Suche, insbesondere wenn diese sich über eine Vielzahl von Einzelschritten erstreckt. Zum anderen

in einem gegenüber der herkömmlichen Entwicklung von Suchkomponenten deutlich reduzierten Aufwand für die Erstellung von problemspezifischen Such- und Analysewerkzeugen, da diese auf bereits vorhandene und gut integrierte Komponenten mit Basisfunktionalität aufsetzen können.

Im Anschluss wurde ein Konzept für die Entwicklung einer solchen Plattform vorgeschlagen und ein grober Überblick über seine Bestandteile, deren Zusammenhänge und Zusammenwirken gegeben. Im darauf folgenden Teil wurde schließlich eine prototypische Implementierung der Expertensuche auf Basis der Suchplattform präsentiert und daran die Stärken der Plattform verdeutlicht. Diese prototypische Implementierung bildet den Rahmen für die in den folgenden drei Kapitel 5-7 diskutierten Komponenten und Speziallösungen.

KAPITEL 5

EIN KONNEKTOR- UND ANALYSEFRAMEWORK

In Kapitel 2 wurde dargestellt, weshalb die alleinige Nutzung herkömmlicher Suchmaschinen in vielen, insbesondere komplexeren Suchszenarien nicht zielführend ist. Zusammenfassend lässt sich an dieser Stelle wiederholen, dass herkömmliche Suchmaschinen in der Regel zu wenig Wissen über den Kontext der Fragestellung oder des Fragestellenden besitzen, um komplexe Fragestellungen direkt beantworten zu können. Zudem stößt der durchschnittliche Benutzer bei der ausreichend präzisen Formulierung seines Informationsbedürfnisses in Form einer Suchanfrage schnell an seine Grenzen. Aus diesem Grund können herkömmliche Suchmaschinen lediglich als ein Teil der Lösungsstrategie für komplexe Suchszenarien angesehen werden, der jedoch um weitere Konzepte und Komponenten ergänzt werden muss.

Die technische Grundlage für solche Ergänzungen wurden in Kapitel 4 anhand der Suchplattform und beispielhaft mittels einer auf ihr bauenden Implementierung der Expertensuche diskutiert. In diesem Zusammenhang wurde an

verschiedenen Stellen auf die im Folgenden behandelten Details zu Konzeption und Umsetzung einzelner Komponenten verwiesen. In den Kapiteln 5-7 sollen nun die drei zentralen Elemente der Suchplattform detailliert diskutiert werden. Zusammen mit der Suchplattform selbst stellen sie das wesentliche Ergebnis dieser Arbeit dar.

Eine zentrale Schwierigkeit für Softwaresysteme zur Analyse von Texten im WWW besteht darin, dass das WWW eine heterogene Sammlung von Inhalten darstellt. Selbst thematisch verwandte Inhalte werden selten in ähnlicher Form präsentiert, es unterscheiden sich Vokabular sowie die verwendete Sprache und zudem oftmals auch die zur Präsentation der Inhalte verwendeten Techniken. Jeder dieser Faktoren bringt Unsicherheiten in die von einer Analysekomponente erzielbaren Ergebnisse und reduziert damit potenziell deren Ergebnisqualität. Im Gegenzug kann daher jedes Wissen über eine Dokumentenquelle, sei es bspw. Wissen über die Struktur einer Webpräsenz oder Wissen über das dort verwendete Vokabular oder die behandelten Themen, eine Steigerung der Ergebnisqualität bewirken – sofern dieses Wissen der Analysekomponente zugänglich gemacht werden kann.

Theoretisch könnte für jede Webpräsenz eine spezielle Analysekomponente implementiert werden. Jegliches Wissen über die Site könnte in diese Komponente integriert werden und damit die Ergebnisqualität positiv beeinflussen. Praktisch ist es aber nicht möglich, für jede Domäne oder Webpräsenz eine spezielle Analysekomponente zu entwickeln, die dieses Wissen umsetzt, da der Entwicklungs- und Wartungsaufwand hierfür nicht zu leisten wäre. Für spezielle Sites und Ressourcen jedoch, die bereits als relevant identifiziert wurden, wird ein solcher Aufwand mit qualitativ hochwertigeren Ergebnissen belohnt. Vor dem Hintergrund der in Kapitel 4.1 vorgestellten Suchplattform stellt sich jedoch die Frage, wie eine solche Spezialisierung generisch und wiederverwendbar gestaltet werden kann, so dass die Integration weiterer Webpräsenzen mit möglichst geringem Aufwand erzielbar ist.

Um diese Frage beantworten zu können, müssen zunächst die zwei wesentlichen Komponenten der Problemstellung näher betrachtet werden. In beiden

Fällen geht es um das Wissen über eine Webpräsenz oder eine Klasse von Webpräsenzen bzw. Webressourcen. Der eine Aspekt betrifft den technischen Zugriff auf die gesuchten Informationen, wohingegen der andere Aspekt die Extraktion der gewünschten Informationen aus den gefundenen Daten behandelt.

Das Problem des Zugriffs auf die gesuchten Informationen lässt sich anhand des Crawler-Beispiels erläutern: Ein Crawler durchforstet das WWW, indem er eine vorgegebene Menge an Start-Dokumenten lädt und rekursiv den darin enthalten Verweisen folgt. Die geladenen Dokumente werden dann von nachgelagerten Komponenten weiter verarbeitet. Zum Beispiel kann daraus ein angepasster Index für eine eigene Suchmaschine erstellt werden oder die o. g. Analyse-Komponenten untersuchen die Dokumente und extrahieren daraus die Eingabe für weitere nachgelagerte und ggf. Webpräsenz-übergreifende Analysekomponenten (vgl. Kapitel 3.1).

Das einfache Prinzip der rekursiven Verfolgung von Hyperlinks ist jedoch nicht immer anwendbar. So liegen große Teile der im Web vorhandenen Informationen nicht in Form statischer Dokumente vor, die über Hyperlinks miteinander verknüpft sind. Der weitaus größte Teil des Web ist im Bereich des so genannten „Deep Web“ zu finden und ist klassischen Crawlern somit zunächst vorenthalten (vgl. Kapitel 2.2).

Die Herausforderung besteht aus Sicht der Crawler-Komponente also darin, einen Weg zu finden, wie Wissen über Schnittstellen solcher Webpräsenzen in den Crawler integriert werden kann, um darüber diese Quellen zu erschließen.

Der zweite oben angeführte Aspekt, die Extraktion der gesuchten Informationen aus den gefundenen Daten, betrifft weniger die Crawling-Komponente als vielmehr die Komponente zur Analyse der gefundenen Ressourcen. Hier kann das Wissen über das Layout eines Textes (bspw. die Lokation der zentralen Informationen), über die Struktur einer Webpräsenz (bspw. im Hinblick auf die Semantik der internen und externen Verknüpfungen) und damit das Wissen über behandelte Themen und handelnde Personen etc. gewinnbringend genutzt werden. Nicht nur in den oben behandelten Fragen des technischen Zugriffs auf diese Daten existieren hier große Unterschiede zwischen Webpräsenzen.

Offensichtlich ist gerade die Frage des Inhalts und dessen Analysierbarkeit eine hochdynamische Problemstellung.

Um nun genau solche Ressourcen von besonderem Interesse im ersten Schritt für das Crawling zugänglich zu machen, und im zweiten Schritt für detaillierte Analysen wie bspw. für die Expertensuche zu erschließen, wurde im Rahmen der Suchplattform ein Zusatzframework, das „Konnektorframework“, entwickelt, welches die Teilaufgaben *Crawling* und *Analyse* unterstützt. Das Framework ist dahingehend flexibel und erweiterbar, dass verschiedene domänenspezifische Konnektoren eingebunden werden können, die den Zugriff auf die gesuchten Ressourcen ermöglichen. Neben verbindungstechnischer Funktionalität wie Authentifizierung und Sessionmanagement wurde zusätzliche Funktionalität vorgesehen, mittels derer Kontextwissen für die Extraktion von Informationen genutzt werden kann. In diesem Rahmen kann das System Wissen über Inhalte und Strukturen einer Webpräsenz nutzen, um bspw. die Relevanz von dort enthaltenen Hyperlinks einzuschätzen, oder um mit hoher Präzision Namen von im Text genannten Personen zu extrahieren, bspw. im Fall von Patentautoren in einer Patentdatenbank. Die Erweiterbarkeit ist dabei eine zentrale Forderung an dieses Konnektorframework, um solche Konnektoren und Analyse-Komponenten mit geringem Aufwand für neue Quellen implementieren zu können.

5.1 Anforderungen und Konzepte der Umsetzung

Im Folgenden werden verschiedene Anforderungen an ein solches Konnektorframework diskutiert und zentrale Konzepte vorgestellt, anhand derer diese Anforderungen im Framework umgesetzt werden.

5.1.1 Asynchrone Verarbeitung

Beim Laden einer einzelnen Ressource aus dem Web sind zwei Partner direkt beteiligt – die herunterladende Anwendung als Client, im vorliegenden Fall der

(Focused) Crawler, und auf Seiten des Inhalteanbieters der Server. Die Verbindung zwischen diesen beiden Partnern erfolgt über ein Netzwerk, das Internet. Da der Server aber in der Regel mehrere Clients bedient, ihm jedoch nur eine beschränkte Netzwerkbandbreite zur Verfügung steht oder an einer anderen Stelle ein Bandbreitenengpass zwischen Client und Server besteht, erhält ein einzelner Client die angeforderten Daten oftmals nicht mit der maximalen dem Client zur Verfügung stehenden Netzwerkbandbreite. Demzufolge nutzt der Client das Betriebsmittel „Netzwerk“ nicht optimal aus. Ferner ist offensichtlich, dass bei den nacheinander ablaufenden Schritten „Herunterladen der Ressource“ und „Verarbeiten der Daten“ jeweils ein einzelnes Betriebsmittel vorrangig genutzt wird: beim Herunterladen ist dies das Netzwerk, bei der Verarbeitung der Daten in erster Linie der Prozessor bzw. der Festspeicher. Für eine Maximierung des Durchsatzes, d. h. um mehr Ressourcen pro Zeit herunterladen und verarbeiten zu können, ist die Parallelisierung dieser Schritte naheliegend. Paralleler Zugriff auf Webressourcen kann verhindern, dass sich einzelne Betriebsmittel im Leerlauf befinden und dass in der Folge Teile des Systems unnötig warten, während z. B. eine Datei aus dem Web heruntergeladen wird. Ähnliches gilt für die Verarbeitung: während der Prozessor mit der Verarbeitung einer Ressource beschäftigt ist, kann parallel der nächste Download gestartet werden, da hierfür kaum Prozessor-Zeit erforderlich ist. Auch lassen sich mittels dieses Konzepts Multi-Core oder Multi-Prozessor-Architekturen besser ausnutzen.

Das Konzept der Parallelität wird im Framework an verschiedenen Stellen eingesetzt, weshalb hier kurz seine softwaretechnische Umsetzung skizziert wird. Das Framework stellt dabei die Infrastruktur zur Ausführung und Überwachung paralleler Aktionen bereit, wobei die Implementierung der jeweiligen vom Framework gesteuerten Komponente für die wechselseitige Isolierung der Abläufe und die damit einhergehende Wahrung der Konsistenz selbst verantwortlich ist.

Eine parallelisierbare Aktion wird in einem „Command“-Objekt (vgl. Entwurfsmuster `Command` in [GHJV95]) gekapselt und zur Ausführung an eine `TaskFactory` übergeben. Diese kapselt die Abarbeitung des Kommandos und erzeugt als Rückgabe an die aufrufende Komponente eine Instanz von `Task<R>`. Der generische Typ `R` beschreibt dabei den konkreten Ergebnistyp

des Kommando-Objekts. Über einen auf diesem `Task` registrierten `Observer` (vgl. Entwurfsmuster `Observer`) kann sich die aufrufende Komponente über Fortschritt und Erfolg des Kommandos informieren lassen. Zwischenergebnisse sowie das abschließende Ergebnis des Kommandos können über die Methode `Task.getResult()` erfragt werden und stehen der aufrufenden Komponente damit zur Weiterverarbeitung zur Verfügung.

Die Konzipierung aller parallelen bzw. asynchronen Systemelemente nach diesem Design hilft, den Implementierungsaufwand sowohl im Bereich des Systemkerns zu reduzieren, als auch den für Konnektoren, die zu einem späteren Zeitpunkt im Rahmen der Anpassung an bspw. eine Web-Datenbankanwendung hinzugefügt werden. Auch trägt die einfachere Implementierung mehrerer Bausteine sowie die Verlagerung eines Teils der Komplexität in eine einmal zu testende Komponente signifikant zur Fehlerreduktion bei.

5.1.2 Container für Quell- und Analysedaten

Das Herunterladen und Analysieren von Daten ist als Prozess zu verstehen, bei dem unstrukturierte Textdaten sukzessive mit neu gewonnenen Informationen wie bspw. Strukturwissen oder Wissen über Inhalte angereichert werden.

Unter Umständen muss in einem Analyseschritt auf Ergebnisse nicht nur des direkt vorangegangenen Schrittes zurückgegriffen werden, sondern auch auf die Ergebnisse eines weiter zurück liegenden Schrittes. Dies legt nahe, Teilergebnisse mit der eigentlichen Ressource zu kombinieren und in einem Containerobjekt durch die einzelnen Verarbeitungsstufen zu propagieren. Mit der Klasse `Document` ist ein solcher Container geschaffen (Abbildung 5.1).

Vom Framework bereits vorgehaltene Strukturen werden im Folgenden aufgelistet. Diese Liste der bereits implementierten Typen kann jedoch für neue Konnektoren beliebig erweitert werden.

PlainSrcData Ursprünglicher Quellcode der heruntergeladenen Ressource, meist HTML.

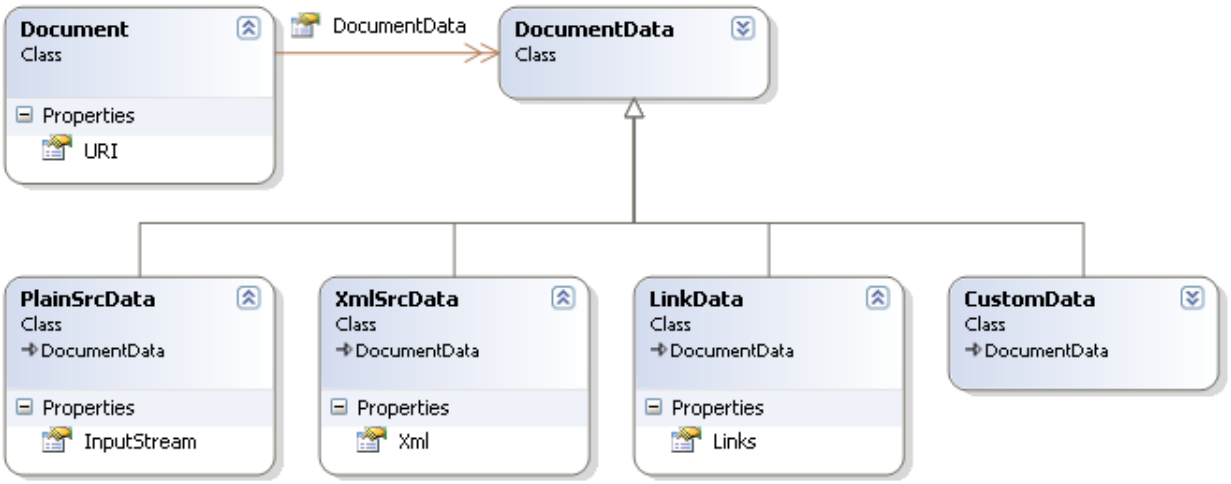

Abbildung 5.1: Repräsentation einer Ressource (Document) und der zugehörigen Inhalte

XmlSrcData XML-Repräsentation der Ressource. Meist der von HTML nach XHTML bzw. XML überführte Quellcode, um die Verarbeitung und Navigation im Dokument zu vereinfachen.

LinkData Fasst Informationen über Hyperlinks zusammen, die auf einer Ressource identifiziert wurden. Dabei kann neben Link-Ziel und Link-Text auch ein Typ-Deskriptor hinterlegt werden, der angibt, welcher Ressourcentyp unter dieser Adresse zu erwarten ist. D. h. ein Konnektor kann hier Kontextwissen verwalten, das in den nachfolgenden Schritten bspw. zur Priorisierung der gefundenen Verweise genutzt werden kann.

CustomData Dient als Basisklasse für domänenspezifische Erweiterungen.

Dieser offene, prozessorientierte Ansatz hat sich als flexibel und erweiterungsfreundlich herausgestellt. Er unterstützt damit wesentlich das Konzept der Erweiterbarkeit des Frameworks und ermöglicht eine durchgängige Sicht auf das Datenmaterial, vom Download über die Analyse bis hin zur Präsentation [KSJ07].

5.1.3 Registrierung

Um das Framework auch mit neu entwickelten, auf ein spezielles Suchproblem abgestimmten Konnektoren einsetzen zu können, wird eine Erweiterungsmöglichkeit benötigt, die mittels eines Plugin-Konzeptes umgesetzt wurde. Die `Registry` stellt bei diesem Ansatz die zentrale Komponente dar. Sie bietet die Möglichkeit, Klassenbibliotheken in Form von Plugins abzulegen und diese dem Framework zugänglich zu machen. Über entsprechende Filtermethoden ermöglicht sie dem Framework und den darin ausgeführten Komponenten den Zugriff auf Klassen dieser Bibliothek. Die Selektion erfolgt dabei über den Typ der benötigten Klasse, d. h. es können wahlweise exakt spezifizierte Klassen gesucht und instantiiert werden, oder aber Listen von Klassen erfragt werden, die eine bestimmte Schnittstelle implementieren und über Properties spezifizierte Eigenschaften besitzen. Auf diese Art und Weise kann das Framework bspw. automatisch den für eine bestimmte Webpräsenz vorgesehenen, weil speziell angepassten, Konnektor identifizieren und für das Crawling dieser Webpräsenz einsetzen.

Ferner lassen sich in der `Registry` XML-Dateien ablegen und anhand von XML-Schemata nach diesen suchen. Dies kann bspw. dazu genutzt werden, Authentifizierungsdaten für bestimmte Webpräsenzen oder Domänen anzugeben, und diese Daten vom System automatisch verarbeiten zu lassen.

5.2 Entwurf und Implementierung

Die Abbildung 5.2 stellt die im Framework vorhandenen Funktionsblöcke dar. Ein zentraler Leitgedanke bei der Konzeption des Frameworks war, die funktionale Trennung von Datenbeschaffung und der Weiterverarbeitung auch in einer architektonischen Trennung widerzuspiegeln. Dadurch können Abhängigkeiten der einzelnen Komponenten untereinander reduziert werden, was die Entwicklung neuer Konnektoren sowie deren Wiederverwendung signifikant vereinfacht.

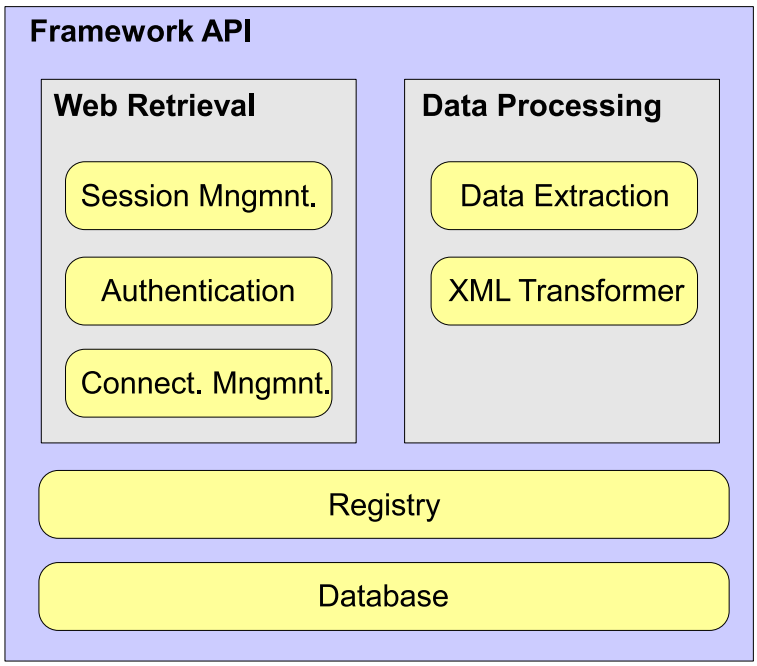

Abbildung 5.2: Funktionsblöcke des Konnektorframeworks

Die Zweiteilung wird anhand der in zwei Paketen umgesetzten Funktionsblöcke deutlich: Linker Hand ist das Paket `WebRetrieval` skizziert. Es umfasst Klassen, die für die Datenbeschaffung zuständig sind. Dies schließt u. a. Aspekte wie Authentifizierung und Sessionmanagement ein. Rechts dagegen ist das Paket `DataProcessing` skizziert, dessen Klassen Funktionalität zur Weiterverarbeitung der heruntergeladenen Daten bereit stellen.

5.2.1 Das Paket `WebRetrieval`

Während die vorherigen Abschnitte die direkte, für externe Komponenten bereit gestellte Schnittstelle des Frameworks diskutierten, so wird im Folgenden nun auf die beiden zentralen in Abbildung 5.2 dargestellten Komponenten `WebRetrieval` und `DataProcessing` eingegangen und deren Funktionalität erläutert.

Das Paket `WebRetrieval` kapselt Funktionalität für das Herunterladen von Web-Ressourcen. Die drei wesentlichen Bausteine sind dabei das Session Management, die Authentifizierung sowie das Connection Management.

5.2.1.1 Session Management

Viele Anfragen eines Clients an einen Webserver sind derart einfacher Natur, dass lediglich für die kurze Zeit der Datenübertragung eine Verbindung zwischen Client und Server aufgebaut werden muss. Der Client spezifiziert die gewünschten Daten und diese werden vom Server geliefert. Danach wird die Verbindung abgebaut und weder Client noch Server müssen Informationen über die getätigte Transaktion speichern, da diese Transaktion vollständig isoliert von anderen ist und keine spätere Transaktion auf einer Vorhergehenden aufbaut.

Der Sachverhalt ist komplizierter wo diese Isolation von Anfragen nicht mehr gegeben ist. Dies ist meist der Fall wenn bspw. auf Seiten des Servers Informationen über das Benutzerverhalten gesammelt werden sollen, also Abfolge von Seitenaufrufen und ggf. die Verweildauer auf einzelnen Seiten protokolliert werden. Auch wenn die Auslieferung oder Darstellung einer Seite von den zuvor besuchten Seiten abhängt, wie dies bspw. bei aufwändigeren Webanwendungen der Fall ist, oder auch zwangsläufig immer dann, wenn der Zugriff auf eine Ressource eine vorangehende Authentifizierung erfordert, müssen auf Seiten des Servers Zustandsinformationen gehalten werden, die dem jeweiligen Client zugeordnet werden. Die serverseitige Verknüpfung von zusammengehörigen Anfragen erfolgt dabei in der Regel über sog. „Sitzungen“ (engl. „Sessions“), die mittels vom Server generierten „SessionIDs“ gekennzeichnet werden.

Das im WWW verwendete Protokoll für den Datenaustausch zwischen Client und Server, das „Hypertext Transfer Protocol“ (HTTP) [RFCa], ist jedoch ein zustandsloses Protokoll. Zustandslosigkeit bedeutet in diesem Zusammenhang, dass das Protokoll keine Möglichkeit vorsieht, den aktuellen Zustand der Client-Anwendung, bspw. „(nicht) authentifiziert“, auf Protokollebene zu verwalten.

Statt dessen müssen Informationen über den aktuellen Zustand des Clients bzw. der Client-Anwendung auf der Applikationsebene verwaltet und außerhalb des Protokolls im Rahmen der Nutzdaten bei jeder Anfrage übertragen werden. Das HTTP-Protokoll sieht dafür zwei Möglichkeiten vor:

1. Die Übertragung von Zusatzdaten in Form eines sog. „Cookies“

2. Die Übertragung von Zusatzdaten als Bestandteil der URL der angefragten Ressource

Da es sich beim Sessionmanagement um grundlegende Funktionalität handelt, die von potenziell vielen Konnektoren genutzt werden soll, liegt eine Implementierung im Rahmen des Konnektorframeworks nahe. Im Folgenden wird daher kurz auf die beiden o. g. Möglichkeiten der Übertragung von Zustandsinformationen eingegangen.

Cookies

Der Internetstandard RFC 2965 [RFCc] spezifiziert eine Möglichkeit zur Übertragung von Zustandsinformationen über das zustandslose HTTP-Protokoll, das sog. „Cookie“. Das Cookie wird im Rahmen der Headerdaten bei Anfrage und Antwort, außerhalb der Nutzdaten übertragen. Hat ein Webserver eine Session angelegt und ihr eine SessionID zugeordnet, so kann er mittels des Antwort-Headers `Set-Cookie` diese SessionID an den Client melden.

Ein RFC 2965-konformer Client wird – sofern der Benutzer keinen Widerspruch einlegt – von diesem Moment an den Cookie in jede weitere Anfrage an den selben Server in den Anfrage-Header einbinden und an den Server übermitteln.

Damit ist der Server in der Lage, über mehrere auf Protokollebene unabhängige Anfragen hinweg, einen Zustand zu verwalten.

GET-Parameter

Als Alternative zur Übertragung von Zustandsinformationen in Form von Cookies können diese Daten auch in die URL der angefragten Ressource integriert werden. In der Praxis wird dieses, aus kosmetischer Sicht weniger ansprechende, Verfahren häufig als Rückfallebene für den Fall verwendet, dass der Client keine Cookies unterstützt bzw. der Benutzer die Annahme des Cookies verweigert.

Eine URL setzt sich aus mehreren Komponenten zusammen (vgl. Abbildung 5.3). Der für die Integration von Zusatzinformationen interessante Teil ist dabei der

Abbildung 5.3: Aufbau einer URL. Quelle: [Gei08]

Parameter-Teil. Durch das Anhängen eines „?“ an den Ressourcen-Pfad kann im URL-String dessen Ende angezeigt werden und eine Folge von Zusatzparametern angegeben werden.

Um nun diese Zustandsinformation über mehrere aufeinander aufbauende Anfragen hinweg zu erhalten, verändert der Server den Quelltext der von ihm an den Client gelieferten Ressourcen dahingehend, dass er allen Hyperlinks der Ressource, die nicht zu fremden Ressourcen führen, den entsprechenden Parameter anhängt, sie also gewissermaßen „verbiegt“. Folgt nun der Client einem dieser modifizierten Verweise, so übermittelt er automatisch die SessionID an den Server.

Umsetzung

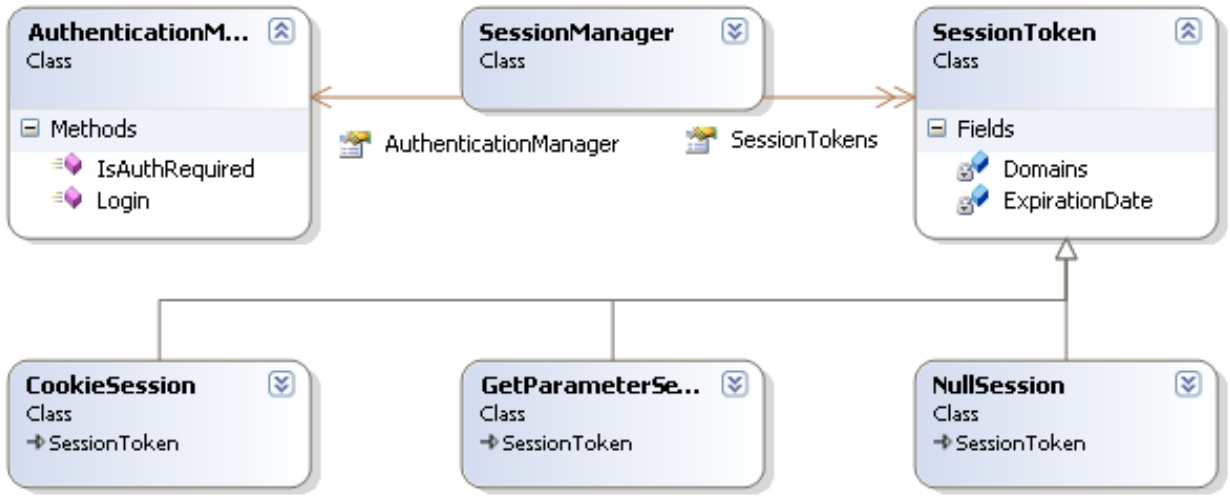

Abbildung 5.4: Unterpaket `SessionManagement`

Abbildung 5.4 zeigt den Aufbau des im Framework umgesetzten Sessionmanagements. Die zentrale Komponente dabei ist der `SessionManager`, der für eine angegebene Webpräsenz bzw. URL die Verwaltung der Session übernimmt. Er er-

zeugt über den im nächsten Abschnitt diskutierten `AuthenticationManager` neue `SessionTokens`, bzw. verwaltet die Tokens bereits existierender Sessions. Für Ressourcen, die kein Sessionmanagement benötigen, wird eine `NullSession`-Instanz erzeugt (vgl. Entwurfsmuster „Null Object“ in [Fow99]), um eine einheitliche Behandlung des `SessionTokens` über das gesamte Framework hinweg zu erreichen.

Das auf diese Art erzeugte und bei folgenden Aufrufen wiederverwendete `SessionToken` dient der Erstellung einer HTTP-Anfrage für das Herunterladen der spezifizierten Ressource. Dazu implementieren die Klassen `CookieSession` und `GetParameterSession` die `SessionToken`-Schnittstelle. Sie setzen damit die beiden zuvor beschriebenen Methoden des Sessionmanagements entsprechend um.

5.2.1.2 Authentifizierung

Verschiedene WWW-Ressourcen sind mit einer Zugriffskontrolle versehen, um bestimmte Inhalte nur für bestimmte Personen oder Systeme zugreifbar zu machen, bspw. weil es sich um kostenpflichtige Inhalte handelt. Um Zugriff auf die gewünschte Ressource zu erhalten, muss der Client dem Server gegenüber nachweisen, dass er die nötige Berechtigung für den Zugriff besitzt. Dazu existieren zwei standardisierte sowie weitere nicht standardisierte Verfahren, die im Folgenden kurz beschrieben werden.

Basic Authentication und Digest Access Authentication

Der Internetstandard RFC 2617 [RFCb] sieht vor, dass ein Webserver die Auslieferung einer passwortgeschützten Ressource verweigern kann, wenn der Client zuvor nicht authentifiziert wurde. Er teilt dies dem Client in der Antwort anhand des HTTP-Statuscodes `401 - Unauthorized` mit.

Für den Client existieren nun zwei Möglichkeiten, die Authentifizierung durch den Server zu erreichen:

1. Basic Authentication
2. Digest Access Authentication

Um dem Client mitzuteilen, dass er die *Basic Authentication*-Methode unterstützt, schickt der Server einen zusätzlichen HTTP-Header in der o. g. Antwort: `WWW-Authenticate: Basic ...`. Der Client seinerseits wiederholt seine zuvor abgelehnte Anfrage und erweitert diese um einen HTTP-Header `Authorization: Basic <basic-credentials>`. Da die Zugangsdaten des Clients bei diesem Verfahren unverschlüsselt übertragen werden, entsteht an dieser Stelle ein gewisses Sicherheitsrisiko.

Um diese Schwachstelle zu eliminieren, wurde die zweite Variante entwickelt, das *Digest Access Authentication*-Verfahren. Dabei sendet der Server zusammen mit der Ablehnung der ersten Anfrage eine zufällige Zeichenfolge, den sog. „nonce value“ an den Client. Dieser kombiniert den *nonce value* mit seinen Zugangsdaten und bildet darüber einen Hash-Wert. Dieser Hash-Wert wird nun anstelle des Passworts zusammen mit dem Benutzername und dem *nonce-value* in der zweiten Anfrage an den Server gesendet. Da die Hashfunktion nicht umkehrbar ist, lässt sich das Passwort somit nicht zurückrechnen, auch wenn die Verbindung abgehört wird. Der *nonce value* stellt dabei sicher, dass ein einmal abgehörtes Triple $(Benutzername, nonce value, Hash)$ nicht für weitere Anfragen missbraucht werden kann, da er vom Server jedes mal neu vergeben wird und somit auch der Hashwert bei jeder Anfrage ein anderer ist.

Authentifizierung mittles HTML-Formularen

In vielen Fällen jedoch greifen Inhalteanbieter zur Benutzerauthentifizierung nicht auf die im HTTP-Protokoll vorgesehenen o. g. Methoden zurück, sondern implementieren die Zugriffskontrolle auf der Applikationsebene. Die Gründe hierfür liegen neben der ggf. höheren Sicherheit auch oftmals in einer einfacheren Benutzerverwaltung sowie in erster Linie in der erhöhten Flexibilität in Sachen optischer Gestaltung des Authentifizierungsvorgangs.

Umgesetzt wird dies i. d. R. anhand von HTML-Formularen, in die der Benutzer vor der erstmaligen Anforderung einer geschützten Ressource seine Zugangsdaten – Benutzername und Passwort – einträgt. Diese werden dann im Rahmen des HTTP-Headers an den Server gesendet, ggf. über eine verschlüsselte Verbindung (HTTPS). Im Gegensatz zu den o. g. Standard-Methoden, erfolgt die Übertragung in diesem Fall nicht bei jeder weiteren Anfrage, sondern beschränkt sich auf dieses eine mal. Der Server überprüft nun die Zugangsdaten auf Gültigkeit und authentifiziert den Client im Erfolgsfall. Die Authentifizierung erfolgt dadurch, dass der Server eine Sitzung erzeugt und dem Client die entsprechende SessionID zurückmeldet (siehe Kapitel 5.2.1.1). Konnte der Benutzer nicht authentifiziert werden, so wird eine solche Sitzung nicht erzeugt. Bei nachfolgenden Anfragen genügt also die Übertragung der SessionID in Form eines Cookies bzw. als URL-Parameter, um dem Server gegenüber den Nachweis zu führen, dass der Client im Vorfeld bereits authentifiziert wurde.

In der Praxis existieren verschiedene Abwandlungen dieses Verfahrens [Gei08]. Sie wurden im Rahmen des Frameworks ebenfalls umgesetzt bzw. vorbereitet, um einen möglichst breite Abdeckung zu erreichen und die Entwicklung neuer Konnektoren zu vereinfachen.

Umsetzung

Die Umsetzung im Framework erfolgte wie in Abbildung 5.5 dargestellt.

Der Ablauf einer Authentifizierung gestaltet sich folgendermaßen: zunächst stellt der `SessionManager` eine Anfrage an den `AuthenticationManager`: `Login()`. Dieser ermittelt über die `Registry` die zu verwendende Authentisierungsmethode, bspw. die `DynamicFormAuthentication`, welche über den im nächsten Abschnitt diskutierten `ConnectionManager` zunächst das vom Server vorgehaltene HMTL-Formular lädt. Aus dem Formular werden ggf. dynamische Parameter extrahiert [Gei08], es werden die Zugangsdaten gesetzt und diese Daten im Rahmen einer zweiten Anfrage über den `ConnectionManager` an

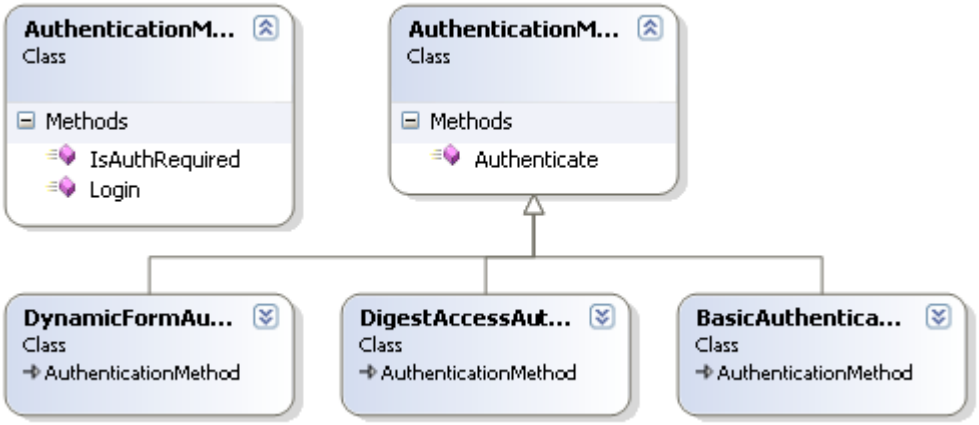

Abbildung 5.5: Unterpaket `Authentication`

den Server gesendet. Bei erfolgreicher Authentifizierung wird aus der Antwort ein `SessionToken` generiert und an den `SessionManager` zurück geliefert.

Ein XML-Schema legt fest, wie Zugangsdaten und Typ des von einer Ressource verwendeten Kontrollmechanismus spezifiziert werden können. Instanzen dieses Schemas können dem Framework über die `Registry` zugänglich gemacht werden.

5.2.1.3 Connection Management

Die beiden zuvor diskutierten Komponenten `Authentifizierung` und `Session Management` implementieren wichtige Details für das automatisierte Herunterladen von Webressourcen. Analog dazu zeichnet die Komponente `Connection-Management` verantwortlich für die Verwaltung der einzelnen Serververbindungen. Sie bedient sich der beiden zuvor genannten Komponenten, um Verbindungen zu Webservern aufzubauen und Ressourcen aus dem Web zu laden. Anhand des `SessionManagers` erfolgt dabei die Zuordnung zu evtl. vorangegangenen Anfragen an den selben Server, sowie ggf. eine Authentifizierung durch diesen. Daraufhin wird die Ressource vom Server angefordert und im Erfolgsfall eine Instanz der `Document`-Klasse erzeugt. Diese wird zunächst mit `PlainSrcData`, also dem Quelltext der Ressource gefüllt.

Das Verbindungsmanagement ist aber nicht nur für die eigentliche Durchführung von Downloads verantwortlich, sondern übernimmt auch die Koordinierung von parallelen Verbindungen. Parallelität ist nötig, da eine rein sequentielle Abarbeitung von Downloads die verfügbare Netzwerk- und Rechenkapazität meist nicht auslasten würde und durch Parallelisierung entsprechende Geschwindigkeitssteigerungen erreicht werden können (vgl. Kapitel 5.1.1). Bei der Koordinierung paralleler Abläufe müssen verschiedene Randbedingungen beachtet werden. Bspw. ist es im Web üblich, einen einzelnen Server nicht durch eine hohe Anzahl Anfragen innerhalb kurzer Zeit unter Last zu setzen und somit unangemessen viele Serverkapazitäten zu beanspruchen. Solche und ähnliche Regeln werden vom `ConnectionManager` implementiert und überwacht, da ein Verstoß gegen sie in der Praxis oftmals dazu führt, dass der Client für weitere Anfragen vom Server ausgeschlossen wird.

5.2.2 Das Paket `DataProcessing`

Nachdem eine Ressource vom Webserver herunter geladen wurde, erfolgt die weitere Verarbeitung. Für verschiedene Arten der Weiterverarbeitung wird im Framework im Rahmen des Pakets `DataProcessing` Funktionalität zur Verfügung gestellt. Die zwei wesentlichen Komponenten sind dabei der `XML Transformer` und die `Data Extraction`.

5.2.2.1 XML Transformer/HTML Fehlerkompensation

Die im Web vorherrschende Sprache zur Beschreibung von Inhalten ist die „Hypertext Markup Language“ (HTML). Sie vereint Konzepte zur Beschreibung von Inhalten und solche zur Beschreibung des Layouts einer Ressource. Beide Arten von Informationen sind je nach Einsatzszenario mehr oder weniger relevant, und können – zumindest teilweise – durch Analyse des HTML-Quellcodes extrahiert werden.

Eben diese Analyse stellt sich in der Praxis jedoch als ein Problem dar. Zwar existiert für HTML eine Spezifikation [HTM], die auch permanent vom W3C weiter entwickelt wird. Allein entlang der Spezifikation kann jedoch kein praktisch einsetzbares Analyse-Werkzeug entwickelt werden, da die meisten HTML-Ressourcen im Web eine Vielzahl an Fehlern im HTML-Quelltext enthalten, d. h. sie entsprechen nicht vollständig der HTML-Spezifikation. Eine Untersuchung [Val05] aus dem Jahr 2005 hat gezeigt, dass lediglich knapp 4% aller deutschsprachigen Webressourcen mit fehlerfreiem HTML codiert sind. Ähnliches hat eine Untersuchung [Wil08] des Browserherstellers Opera aus dem Jahr 2008 gezeigt: lediglich ca. 4 % von 3,5 Mio untersuchten internationalen Webressourcen waren mit korrektem HTML beschrieben.

Das Problem für die Verarbeitung von HTML-Daten ergibt sich nun aus der Tatsache, dass gängige Webbrowser viele dieser Fehler kompensieren können und dadurch die Verbreitung von fehlerhaften HTML eher fördern denn eindämmen. Der Ersteller von Webinhalten bekommt also bei kleineren Fehlern keine Rückmeldung über diese und wird sie daher in den seltensten Fällen korrigieren.

Für selbst entwickelte Analysekomponenten bedeutet dies jedoch, dass auch sie in der Lage sein müssen, diese Fehler zu kompensieren, um nicht in Fehlersituationen zu geraten bzw. um keine falsche Bedeutung in die Daten zu interpretieren. Ferner bedeutet dies, dass entweder jede einzelne Komponente die für sie relevante Fehlerkompensation implementieren muss, oder aber dass das Framework eine umfassende Kompensation selbst vornimmt. Ersteres ist offensichtlich aus Gründen der Redundanzvermeidung, Fehleranfälligkeit, des hohen Entwicklungsaufwandes und der Wartbarkeit als die schlechtere Alternative anzusehen und statt dessen ist die zentrale Kompensation vorzuziehen.

Da die Entwicklung entsprechender Fehlerkompensationstechniken als Teil der Analyse im Rahmen dieser Arbeit zu aufwändig war, wurde ein einfacherer und dennoch praktikabler Ansatz gewählt. Mit der Open Source Software `JTidy` [JTi] existiert eine frei Bibliothek, mittels der ein potenziell fehlerbehaftetes HTML-Dokument in ein wohlgeformtes XML-Dokument überführt werden kann.

Die XML-Struktur besitzt gegenüber der HTML-Struktur den Vorteil, dass sie sich einfacher verarbeiten lässt, da für das Handling von XML-Strukturen entsprechende Softwarelösungen zur Verfügung stehen.

Das bei dieser Transformation von HTML nach XML entstandene XML-Dokument wird als Ergebnis dieses Prozessschrittes in Form eines `XmlSrcData`-Objekts in den `Document`-Container eingefügt und steht somit allen weiteren Komponenten zur Verfügung.

5.2.2.2 Data Extraction

Der letzte Schritt des im Framework definierten Prozesses besteht aus der „Data Extraction“, d. h. aus der Analyse der heruntergeladenen Ressource und der Bereitstellung der extrahierten Informationen in Form von Laufzeit-Datenstrukturen an die aufrufende externe Komponente.

Neben der verbindungstechnischen Erschließung von Webressourcen übernimmt das Framework an dieser Stelle auch die inhaltliche Erschließung. Für das Suchsystem besteht eine Webressource zunächst aus einer Sammlung struktureller und textueller Daten, deren Semantik jedoch im Allgemeinen weitgehend unbekannt ist. Während sich nun mit relativ einfachen Mitteln offensichtliche Informationen extrahieren lassen, bspw. alle Hyperlinks einer Ressource, so ist eine weiter gehende Analyse meist nur durch die Nutzung von Kontextwissen möglich – welche Art Informationen sollen gefunden werden, in welchem Bereich des Dokuments können sie gefunden werden etc.

Um nun die Nutzung solchen Kontextwissens zu ermöglichen, wurde im Framework das Konzept des `Extractor` implementiert (im Folgenden auch deutsch „Extraktor“ genannt). Ein Extraktor ist zugeschnitten auf eine bestimmte Klasse von Webressourcen und kann das Wissen über deren Struktur und Inhalt in die Analyse einfließen lassen, um somit die Qualität des Analyseergebnisses zu verbessern. Dazu operiert der Extraktor auf der aus dem vorherigen Schritt gewonnen XML-Repräsentation der Ressource und extrahiert Informationen,

die er dann in Form von Laufzeitdatenstrukturen den übrigen Komponenten des Systems zur Verfügung stellt.

Die in Kapitel 5.1.3 diskutierte `Registry` verwaltet eine Liste der im System bekannten Extraktoren und bietet sie dem `ExtractionCoordinator` zur Auswahl. Dieser übergibt nun an jeden der vorhandenen Extraktoren die URL und alle bekannten Metadaten der aktuell bearbeiteten Ressource. Er erhält als Antwort eine Wertung, die angibt, wie geeignet der jeweilige Extraktor für die Weiterverarbeitung der spezifizierten Ressource ist. Auf diese Art und Weise kann ein mehrstufiges Extraktoren-System implementiert werden, das stark spezialisierte Extraktoren für besonders relevante Domänen vorhält. Ebenso kann aber auch das Konzept des `FallbackExtractor` greifen, der dann zum Zuge kommt, wenn kein spezialisierter Extraktor existiert. Ein solcher `FallbackExtractor` extrahiert bspw. lediglich alle vorhandenen Hyperlinks aus einem Dokument und nutzt ansonsten keine weiteren Kontextinformationen.

Der auf diese Art und Weise ermittelte, am besten geeignete Extraktor, führt nun die Analyse und Extraktion der Daten aus. Abhängig von der Zielsetzung kann er dabei bspw. lediglich alle Hyperlinks extrahieren und diese ggf. mit Metadaten angereichert im `Document`-Container ablegen. Es sind aber auch Extraktoren möglich, die speziell auf die Dokumente einer Domäne oder Webanwendung zugeschnitten sind und entsprechendes Wissen über Struktur und Inhalte der Ressourcen in den Extraktionsprozess einfließen lassen. An dieser Stelle werden nun die Vorteile der XML-Struktur gegenüber der potenziell fehlerbehafteten HTML-Struktur deutlich: Anhand einfacher Baumtraversierungalgorithmen kann die gesamte Dokumentenstruktur durchsucht werden. Alternativ lassen sich auch Informationen, deren Position innerhalb der Dokumentenstruktur bekannt sind, einfach bspw. mittels XPath-Ausdrücken extrahieren.

Um die extrahierten Informationen zu persistieren und bei einem Neustart des Programms und damit des Frameworks wieder verfügbar zu machen, werden der `Document`-Container zusammen mit den Metadaten in einer Datenbank abgelegt. Dies ist bspw. notwendig für die URL-Warteschlange des Crawlers und

die zu den darin verwalteten URLs gespeicherten Metadaten wie „erwarteter Typ der Ressource“, „Text des Hyperlink“ oder „Quellressource des Hyperlink“. Somit kann bei einem Neustart an der vorherigen Stelle aufgesetzt werden und kein Wissen über bereits besuchte und noch zu besuchende Ressourcen geht verloren.

5.3 Evaluation

Konzeption und Entwicklung des hier beschriebenen Konnektoren-Frameworks haben sich in erster Linie an den Anforderungen der Analyse von Patentdatenbanken orientiert. Hier sollten im Rahmen der Expertensuche u. a. Personen und Firmen in Patentschriften identifiziert werden. Im Fokus der Arbeiten stand dabei das *DEPATISnet*-System des *Deutschen Patent- und Markenamtes* (DPMA) [DEP]. Der in diesem Rahmen entstandene Konnektor [Gei08] stellt ein mächtiges Werkzeug zur Analyse von Patentinformationen dar. Er nutzt die Funktionalität des Frameworks – insbesondere das Authentifizierungs- und Sitzungsmanagement sowie das Konzept der Extraktoren – um im Rahmen der Expertensuche Informationen aus Patentschriften zur Identifikation von Personen und zur Bewertung deren Expertise zu unterstützen. Der Konnektor wurde zur Expertensuche bei Praxispartnern des Forschungsprojekts *nova-net* eingesetzt [GVK08]. Er eignet sich aber nur bedingt zur Evaluation des Frameworks, da die Frameworkentwicklung durch diesen Konnektor motiviert wurde. Eine darauf basierende Evaluierung wäre also nicht neutral und würde zwangsläufig zu positiven Ergebnissen gelangen.

Um dennoch Funktionalität, Flexibilität und die durch das Framework angestrebte Reduzierung von Implementierungsaufwand bewerten zu können, wurde nach Abschluss der Framework-Implementierung ein weiterer Konnektor entwickelt. Der Aufwand für seine Entwicklung dient dabei als Maß für die Erreichung der Frameworkziele. Gegenstand dieses Konnektors war der Webauftritt der Abteilung *Andwendersoftware* (AS) des *Institut für Parallele und Verteilte Systeme* (IPVS) der Universität Stuttgart [IPV]. Anhand der dort vorgehaltenen

Informationen sollte ein Graph erstellt werden, der die Verbindungen zwischen Forschungsprojekten, an den Projekten beteiligten Lehrstuhlmitarbeitern, deren Publikationen und von ihnen angebotene studentischen Arbeiten sowie die an diesen Arbeiten beteiligten Studenten aufzeigt. Die im Rahmen einer solchen Analyse gewonnenen Information ließen sich in einem realen Suchszenario Gewinn bringend bspw. für die Expertensuche (vgl. Kapitel 1.1) einsetzen.

Im Folgenden wird zunächst ein kurzer Überblick über die relevanten Bereiche des Webauftritts der Abteilung Anwendersoftware gegeben. Im Anschluss erfolgt der Entwurf und die Beschreibung des eingesetzten Datenmodells sowie Details zur Implementierung. Der Abschnitt endet mit der Bewertung des Aufwands.

5.3.1 Struktur des IPVS/AS Webauftritts

Der Definitionsbereich des Konnektors beschränkt sich auf den *Anwendersoftware*-relevanten Bereich des IPVS-Webauftritts. Abbildung 5.6 zeigt den Einstieg in diesen Bereich. Die Verweise, die zu den relevanten Inhalten führen, sind `Abteilung|Mitarbeiter`, `Forschung|Projekte`, `Publikationen` sowie `Lehre|Studentische Arbeiten`. Dort finden sich Listen aller Mitarbeiter des Lehrstuhls, deren Publikationen, die von ihnen angebotenen und betreuten studentischen Arbeiten sowie eine Liste aller Forschungsprojekte und der in diesen Projekten aktiven Mitarbeiter.

5.3.2 Datenmodell

Aus dieser Webpräsenz kann folgendes Daten- bzw. Klassenmodell abgeleitet werden (Abbildung 5.7):

- Die Startseite selbst bietet keine in diesem Zusammenhang relevanten Informationen und muss daher nicht modelliert werden.
- Die Mitarbeiterliste wird durch die Klasse `EmployeeList` modelliert und in die Ergebnismenge aufgenommen, indem sie die Framework-Schnittstelle `CustomData` implementiert.

Abbildung 5.6: Startseite der Abteilung Anwendersoftware

- Analog dazu erfolgt die Modellierung der Übersichtsseite für Forschungsprojekte (`ProjectList`), Publikationen (`PublicationList`) und studentische Arbeiten (`StudentWorkList`).
- Die Mitarbeiter, Studenten, Projekte und Publikationen selbst werden durch entsprechende Klassen modelliert (`Employee`, `Student`, `Project` und `Publication`). Auch sie werden in die Ergebnismenge aufgenommen.

5.3.3 Implementierungsdetails

Der eigentliche Konnektor erfordert keinen Entwicklungsaufwand, da der Webauftritt des IPVS weder eine Authentifizierung des Besuchers erfordert, noch Sitzungsinformationen verwaltet. Die Verbindung zu den gewünschten Ressourcen können also komplett mit den vom Framework zur Verfügung gestellten Mitteln durchgeführt werden. Allgemein gilt dies auch für andere Konnektoren-

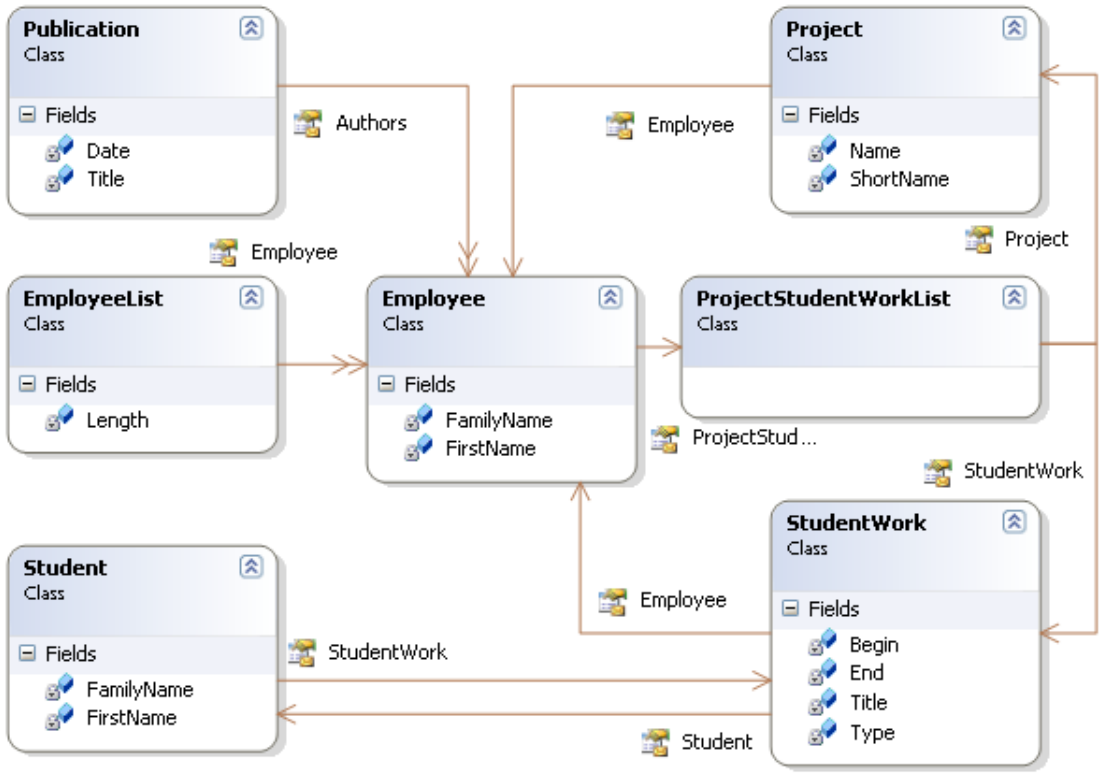

Abbildung 5.7: Datenmodell der Webpräsenz der Abteilung AS (Ausschnitt)

Entwicklungsprojekte. Lediglich in besonderen Fällen, in denen eine Webpräsenz spezielle Authentifizierungsmechanismen verwendet, fällt hier zusätzlicher Entwicklungsaufwand an.

Der Fokus der Konnektoren-Entwicklung lag daher auf der Komponente zur Extraktion relevanter Daten aus den auf der Webpräsenz angebotenen Ressourcen. Prinzipiell wäre aufgrund des einfachen Szenarios eine Umsetzung der Funktionalität in einem einzigen Extraktor möglich gewesen. Aus Gründen der besseren Übersicht wurde jedoch ein eigener Extraktor pro Ressourcentyp entwickelt. Damit sinkt die Komplexität der einzelnen Extraktoren, was diese prinzipiell besser wartbar und an zukünftige Änderungen des IPVS-Webauftritts anpassbar macht.

Das zur Informationsextraktion genutzte Kontextwissen kommt nun bei der Analyse der Ressourcen im Rahmen der einzelnen Extraktoren zum Tragen. Da das Framework automatisch eine Überführung der Quelldaten (HTML) in eine bereinigte XML-Struktur durchführt, kann im Extraktor mit einfachen Mitteln

```
// find the table rows that list an employee
NodeList tableRows = (NodeList)
  xpath.evaluate("//table/tr/td[@bgcolor=\"f0f0f0\"]/..",
                 htmlData.getDOMTree(), XPathConstants.NODESET);

[...]

// for all employees...
for (int i=1; i<tableRows.getLength(); i++) {
  // ... find the hyperlink to the employees homepage
  Node link = (Node)
    xpath.evaluate("./td[1]/font/a/@href",
                   tableRows.item(i),
                   XPathConstants.NODE);

  // ... find the name of the employee
  Node name = (Node)
    xpath.evaluate("./td[1]/font/a/span/text()",
    tableRows.item(i), XPathConstants.NODE);

  // create a result object with URI, name and the Employee-type
  [...] new LinkImpl(
    new URI(link.getNodeValue()),
    name.getNodeValue(), Employee.class);
}
```

Abbildung 5.8: XPath-Ausrücke und Programmcode zur Extraktion von Mitarbeiterinformationen vom IPVS/AS-Webauftritt

auf dieser definierten Struktur operiert werden. Am Beispiel des Extraktors für die Übersichtsseite der Mitarbeiter lässt sich erkennen, dass allein mittels dreier XPath-Ausdrücke die gesamte Mitarbeiterliste generiert werden kann (vgl. Listing 5.8):

1. In einem ersten Schritt werden die Zeilen der Mitarbeitertabelle identifiziert, um die Daten der Mitarbeiter zu ermitteln (`NodeList tableRows = ...`). Über diese Zeilen wird im Folgenden iteriert.
2. Dabei wird zunächst die URL ausgelesen, die zur persönlichen Seite des Mitarbeiters führt (`Node link = ...`).
3. Danach wird der Name des Mitarbeiters ausgelesen (`Node name = ...`).

Diese Informationen werden in Form eines `LinkData`-Objekts an die Ergebnismenge übergeben (`new LinkImpl(...)`).

Abbildung 5.9: Extrahierte Inhalte der Webpräsenz IPVS/AS: Mitarbeiter, Forschungsprojekte und Studentische Arbeiten
(Stand: November 2008)

Analog zur in Listing 5.8 dargestellten Vorgehensweise sind die Extraktoren für die übrigen Ressourcen-Typen (Forschungsprojekte, Publikationen und Studenten) konzipiert. Die Ergebnisse werden von einer einfachen Demonstrations-Anwendung in eine Graph-Datei exportiert. Abbildung 5.9 zeigt einen Ausschnitt aus dem dabei entstanden Graph. Die Publikationsdaten sowie die Namen der Studierenden wurden aus Gründen der Übersichtlichkeit ausgeblendet.

5.4 Aufwand für die Implementierung

Tabelle 5.1 gibt einen Überblick über den Aufwand zur Konzeptionierung und Umsetzung des IPVS/AS-Konnektors. Der Gesamtaufwand lag dabei bei 385 Minuten, also ca. 6,5 Stunden. Allerdings sind in diesem Aufwand auch Tätigkeiten enthalten, die lediglich eine Ausführungsumgebung für den Konnektor bereit gestellt und die Ergebnisse in eine übersichtliche Darstellung in Form eines Graphen überführt haben. Der eigentliche Konnektor konnte in 265 Minuten – also ca. 4,5 Stunden – deutlich schneller realisiert werden.

Tätigkeit	Aufwand
Überblick über den Aufbau der Webpräsenz	20 min
Definition und Implementierung Datentypen	30 min
Extraktor für Liste Mitarbeiter	50 min
Extraktor für Liste Forschungsprojekte/Studentische Arbeiten	20 min
Extraktor für Stammdaten Mitarbeiter und Studenten	35 min
Extraktor für Studentische Arbeit	25 min
Extraktor für Forschungsprojekt	15 min
Extraktor für Publikationen	30 min
Debugging	40 min
Zwischensumme Konnektoren/Extraktoren	265 min
Implementierung Demo-Anwendungsrahmen	60 min
Generierung der Graph-Bitmap	60 min
Summe	385 min

Tabelle 5.1: Zeitliche Übersicht über die Implementierung des IVPS-Konnektors

Die Zahlen für beide Komponenten, den eigentlichen Konnektor/Extraktor

sowie die Demoanwendung sollen im Folgenden diskutiert werden.

Wie bereits oben angeführt, ist für den Zugriff auf den öffentlichen Webauftritt des IPVS/AS keine Authentifizierung nötig. Es findet auch keine anfrageübergreifende Sitzungsverwaltung statt, weshalb keine Cookies o. ä. verwaltet werden müssen. In solchen Fällen muss kein gesonderter Konnektor spezifiziert werden, da das Framework dies als Standard ansieht und entsprechende Konnektorenfunktionalität automatisch einbindet. Im vorliegenden Fall entfiel damit also der Aufwand für die genaue Analyse der Konnektorenparameter und somit auch deren Spezifikation in der XML-Datei für Konnektoren der `Registry`. Wäre dies jedoch nötig gewesen, so hätte sich Dank der bereits im Framework implementierten Funktionalität erneut eine deutliche Aufwandsreduktion gegenüber einer Implementierung ohne Frameworkunterstützung ergeben.

Signifikanter Aufwand floss hingegen in die Extraktor-Entwicklung. Da für diese bisher noch keine Werkzeugunterstützung existiert, musste die Analyse der Dokumente von Hand durchgeführt werden. Die Probleme resultierten in diesem Fall hauptsächlich aus der hohen Anzahl an Fehlern im HTML-Code der Dokumente. Da die externe HTML-Korrekturkomponente (JTidy) diese Fehler nicht vollständig korrekt kompensieren konnte, besaß die den Extraktoren zugrunde liegende XML-Struktur eine geringfügig andere Struktur als die original HTML-Seite. Dies äußerte sich in aufwändigen Suchen und Abgleichen von XML- bzw. HTML-Repräsentation und führte letztlich beim Extraktor für die Mitarbeiterliste zu einem hohen Entwicklungsaufwand von ca. 50 Minuten. Hierauf entfiel auch ein wesentlicher Teil des Debugging-Aufwands. Leistungsfähigere HTML-Korrekturkomponenten können hier jedoch zu einer erheblichen Aufwandsreduktion führen.

Da die übrigen Dokumenttypen nach einem ähnlichen Schema aufgebaut sind, konnten deren Extraktoren entsprechend schneller umgesetzt werden. Nicht zuletzt deshalb, weil ein signifikanter Teil des Codes für den Mitarbeiterlisten-Extraktor mittels Vererbung wiederverwendet werden konnte.

Tabelle 5.1 beziffert die Entwicklungsdauer für die Extraktoren auf knapp 4,5

Stunden. Mit dieser Zahl alleine kann das Potenzial des Frameworks jedoch nicht klar wieder gegeben werden. Bspw. wäre es mit einfachen Mitteln wie Perl/Regulären Ausdrücken sicherlich möglich gewesen, zur reinen Datenextraktion diese vier Stunden zu unterbieten. Allerdings sind dabei zwei Aspekte zu beachten:

- Zum einen musste kein spezieller Konnektor entwickelt werden, da die vom Framework bereit gestellte Basisfunktionalität in diesem Fall ausreichend ist. Wäre jedoch ein Konnektor für die Authentifizierung oder das Sessionmanagement nötig gewesen, dann hätte sich mit dem Framework der Aufwand für die Entwicklung des Konnektors im Wesentlichen auf die Spezifikation der Konnektorenparameter in einer XML-Datei beschränkt, da das Framework umfangreiche Funktionalität für viele Standardfälle bereit hält. Einfache Skripting-Ansätze wären an dieser Stelle gescheitert, da insbesondere die Authentifizierung teilweise komplexe Anfrageabfolgen verlangt und daher entsprechend implementierungsaufwändig ist.
- Der zweite Aspekt betrifft die Demo-Anwendung. Hier konnte gezeigt werden, dass innerhalb von insgesamt nur 6,5 Stunden eine Anwendung entwickelt werden kann, die den gesamten Webauftritt des IPVS/AS nach relevanten Daten durchsucht, diese ggf. persistent zwischen mehreren Suchläufen verwaltet und sie somit sogar einem Monitoring zugänglich machen kann. Letzteres würde eine Entwicklung der Webpräsenz über die Zeit beobachtbar machen und könnte weitere Hinweise auf bspw. die Entwicklung von aktuellen Themenfeldern geben. Von diesen 6,5 Stunden wurde eine Stunde für die grafische Ausgabe der Ergebnisse benötigt, womit die Kernfunktionalität der Sammlung und Auswertung von Daten in lediglich 5,5 Stunden implementiert werden konnte.

Der größte Einzelaufwand wurde, wie oben bereits geschildert, in die Entwicklung der ersten Extraktor-Komponente investiert. An dieser Stelle existiert noch signifikantes Optimierungspotenzial. Da für die Entwicklung eines Extraktors im vorgestellten System noch eine manuelle Analyse von Inhalten und Strukturen der zugrunde liegenden Ressourcen nötig ist, ist dieser Prozessschritt

entsprechend aufwändig. Hier könnte durch eine Werkzeugunterstützung für wiederkehrende Aufgaben, wie bspw. die zeilenweise Extraktion von Daten aus einer Tabelle, viel Aufwand eingespart werden. In [Gei08] findet sich eine Übersicht über aktuelle und mögliche Entwicklungen in diesem Bereich.

5.5 Zusammenfassung

In diesem Abschnitt wurde ein Framework vorgestellt, das die Entwicklung von Softwarekomponenten zur Sammlung und Auswertung von Webressourcen erleichtert. Der Schwerpunkt lag dabei auf den zwei Bereichen *Konnektivität* und *Informationsextraktion*. Anhand eines einfachen Beispielszenarios wurde gezeigt, dass das hier entwickelte Framework einfach einsetz- und erweiterbar ist und die Entwicklung von Suchanwendungen beschleunigen kann. Neben der kürzeren Entwicklungsdauer solcher Suchanwendungen liegt ein weiterer Vorteil des Frameworks in der potenziell höheren Codequalität der Anwendung sowie in deren besserer Wartbarkeit. Dies sind Eigenschaften, die allgemein mit der Nutzungen von Frameworks einhergehen. Sie sind bedingt durch eine klare Strukturierung der Anwendung und eine klare Modularisierung, was beides die Wartbarkeit der Software fördert. Der Aspekt der höheren Codequalität liegt vor allem darin begründet, dass von vielen Komponenten benötigter Code lediglich einmal entwickelt wurde, zentral gewartet wird und dabei vielfach getestet ist [Pre01].

Zusammenfassend lässt sich festhalten, dass dieses Framework auf flexible und erweiterbare Art und Weise die Lücke zwischen der Crawlersteuerung und der eigentlichen Analysekomponente schließt. Es reduziert den Aufwand für die Entwicklung von Komponenten zur Sammlung von Daten und unterstützt deren Auswertung, indem es hilft, Kontextwissen in den Auswertungsprozess einzubringen.

KAPITEL 6

BERECHNUNG VON INHALTLICHER DOKUMENTENÄHNLICHKEIT

In Kapitel 5 stand die Frage nach dem Zugriff auf Dokumente im WWW im Vordergrund. Ferner wurde diskutiert, wie relevante Daten in diesen Dokumenten identifiziert und extrahiert werden können. Aussagen über die Relevanz einer Ressource für eine Suchanfrage wurden in diesem Zusammenhang aber nicht gemacht.

Kapitel 4.2 diskutiert ein Focused-Crawler Szenario, in welchem diese Relevanz anhand der Ähnlichkeit von Dokumenten berechnet wird. Der Crawler durchsucht dabei das WWW und fokussiert sich auf Dokumente, die zu dem Themenbereich gehören, welcher durch die Suchanfrage spezifiziert wurde. Die Ähnlichkeit von Dokumenten kommt nun ins Spiel, wenn der Crawler eine Ressource lädt und auf Ihre Ähnlichkeit zu bereits als relevant klassifizierten Ressourcen prüft. Damit lassen sich bei vertretbarem Initialaufwand für das Zusammenstellen der Referenzdokumente schnell weitere themenrelevante Dokumente finden. Ferner kann mittels eines dabei eingesetzten

Ähnlichkeitsmaßes die URL-Warteschlange des Crawlers dahingehend priorisiert werden, dass zuerst solchen Verweisen gefolgt wird, die auf Dokumenten gefunden wurden, welche als „ähnlich“ zu anderen relevanten Dokumenten gelten. Diese Priorisierung setzt die Idee um, dass Verweise in erster Linie zu verwandten Dokumenten führen und daher relevante Dokumente zunächst in der Hyperlink-Nachbarschaft anderer relevanter Dokumente gesucht werden sollten [DCL+00].

Daneben existieren verschiedene weitere Szenarien, in denen ein Ähnlichkeitsmaß für Dokumente benötigt wird, teilweise relevant im Kontext dieser Arbeit, teilweise außerhalb dieser Arbeit. Beispiele sind die Funktion „Finde ähnliche Dokumente“ vieler Standardsuchmaschinen oder das Clustering von Dokumenten, mit dem Ziel, eine Dokumentenmenge so in Teilmengen aufzuteilen, dass die Ähnlichkeit aller Dokumente innerhalb einer Teilmenge maximal und die Ähnlichkeit mit Dokumenten anderer Teilmengen minimal ist (vergleiche Kapitel 3.2.1).

Viele Vorschläge zur Definition eines solchen Ähnlichkeitsmaßes für Dokumente bzw. Texte wurden bereits gemacht. Frühe Arbeiten beschäftigen sich dabei hautpsächlich mit der Frage der syntaktischen Ähnlichkeit. Neuere Arbeiten hingegen fokussieren stärker auf die inhaltliche Ähnlichkeit von Texten. In Kapitel 3.1.3 werden einige solche Ansätze vorgestellt. Ihnen allen ist gemein, dass sie zusätzliches Hintergrund- oder Kontextwissen nutzen, um im Vergleich zu rein syntaxbasierten Verfahren bessere Ergebnisse zu erzielen und dazu anstatt auf textueller Ebene auf einer abstrakteren Ebene so genannter Konzepte arbeiten. Die Idee hinter der Orientierung an Konzepten, die konkreten (syntaktischen) Termen übergeordnet sind, besteht darin, dass auch das menschliche Verständnis der Ähnlichkeit von Texten weniger auf begrifflicher, als auf konzeptioneller Ebene angesiedelt ist [GM07]. D. h. dass Menschen in der Lage sind, von der Syntax eines Textes auf seinen Inhalt zu abstrahieren und Textvergleiche auf dieser konzeptionellen Ebene durchzuführen. Der Vorteil gegenüber eines syntaxbasierten Vergleichs ist offensichtlich: zwei Texte A und B bleiben inhaltlich vergleichbar, auch wenn sie auf syntaktischer Ebene verhältnismäßig geringe Übereinstimmungen aufweisen – bspw. wenn unterschiedliches Vokabu-

lar eingesetzt wird, wenn die Texte auf jeweils einen anderen Teilaspekt des gesuchten Themas fokussieren oder ihnen eine unterschiedliche Spezialisierung (bspw. „Auto“ in *Text A* vs. „Porsche 911 Turbo“ in *Text B*) zugrunde liegt.

Während sich verschiedene Ansätze mit der Ähnlichkeit von Texten auf semantischer Ebene befassen, war das Ziel im Rahmen dieser Arbeit, herauszufinden, mit welchen Mitteln eine inhaltliche Ähnlichkeit bewertet werden kann, ohne jedoch aufwändige semantische Analysen durchführen zu müssen. Der Unterschied zwischen inhaltlicher Ähnlichkeit und semantischer Ähnlichkeit wird am folgenden Beispiel deutlich:

```
1) Alice spricht mit Bob.
2) Alice spricht nicht mit Bob.
```

Die Semantik beider Sätze ist verschieden bzw. sie beschreiben sogar gegensätzlich Sachverhalte. Auf semantischer Ebene ist die Ähnlichkeit also als eher gering einzuschätzen. Dennoch weisen beide Sätze eine hohe inhaltliche Ähnlichkeit auf: in beiden Fällen geht es um Kommunikation zwischen Alice und Bob. Genau diese inhaltliche Ähnlichkeit soll im Rahmen der vorliegenden Arbeit quantifiziert werden, da auf dieser Ebene relevante Ergebnisse für eine große Klasse von Websuchen zu erwarten sind: In dem in Kapitel 1.1 vorgestellten Szenario geht es bspw. um die Erkundung neuer Technologiefelder. Dabei kommt es offensichtlich weniger darauf an, Dokumente mit identischen Aussagen oder Inhalten zu identifizieren, als vielmehr solche Dokumente, die das selbe Thema oder den selben Sachverhalt behandeln.

Dazu wird im Folgenden ein Ansatz vorgestellt, anhand dessen unter Nutzung von Inhalts- und Strukturinformationen des Wikipedia-Hypertext-Korpus ein Modell eines Textes – der sog. „konzeptionelle Kontext“ – berechnet werden kann. Ferner wird auf diesen Kontexten ein Ähnlichkeitsmaß definiert [KSJ09] welches das oben diskutierte Konzept der „inhaltlichen Ähnlichkeit“ abbildet. Von der Ähnlichkeit dieser Kontexte wird dabei auf die Ähnlichkeit der ihnen zugrunde liegenden Dokumente geschlossen. Nachfolgend wird dieser Ansatz vorgestellt und im Rahmen einer Evaluierung in verschiedenen Szenarien

diskutiert. Für einen Überblick über verwandte Arbeiten sei an dieser Stelle auf Kapitel 3.1.3 verwiesen.

6.1 Nutzung von Wikipedia-Daten

In den letzten Jahren haben sich im Bereich des Information Retrieval viele Arbeiten damit beschäftigt, wie die in großen öffentlichen Datensammlungen wie bspw. der Wikipedia zur Verfügung stehenden Daten und Informationen gewinnbringend für die Lösung von Suchproblemen eingesetzt werden können. Der Reiz solcher Sammlungen besteht darin, dass unzählige Personen dort Informationen zusammen tragen, diese Informationen zumindest rudimentär strukturiert sind und eine breite thematische Fächerung aufweisen. Im Falle der freien Enzyklopädie „Wikipedia“ bedeutet dies konkret:

- Wikipedia bietet eine umfassende Sammlung von Texten zu allen erdenklichen Themen. Kann ein Softwaresystem aus diesen Texten relevante Informationen extrahieren, so kann es zur Lösung oder Unterstützung bei Suchproblemen auf all diesen Themenbereichen eingesetzt werden.
- Neben den rein textuellen Informationen der Wikipedia existieren verschiedene Konzepte der Verknüpfung von Informationen: wer einen Artikel erstellt oder modifiziert, der kann Verweise zu anderen Artikeln einfügen, die er für relevant erachtet. Daneben existiert das System der Kategorien, das eine n:m-Beziehung zwischen Artikeln und diese Artikel enthaltenden Kategorien widerspiegelt. Beide Konzepte können wertvolle Eingaben für die inhaltliche Analyse von Texten liefern, da sie Begriffe oder Texte in eine inhaltliche Beziehung setzen und die daraus gewonnenen Erkenntnisse unter bestimmten Voraussetzungen auf andere Texte angewandt werden können [KSJ09][SP06].
- Wikipedia ist in verschiedenen Sprachen verfügbar. Zudem existieren häufig Verweise zwischen Artikeln, die den selben Gegenstand in unterschiedlichen Sprachen behandeln. Das erlaubt prinzipiell den Einsatz von auf Wikipedia bauenden Systemen über Sprachgrenzen hinweg [MLD+10].

- Wikipedia erlaubt jedem, eigene Inhalte einzufügen und Existierendes zu erweitern bzw. zu modifizieren. Aufgrund dieses Ansatzes steigt die dort abgelegte Informationsmenge beständig an, wobei verschiedene Regeln und eine diese Regeln überwachende Community eine gewisse Informationsqualität sicher stellen.

Die grundlegende Idee des in dieser Arbeit verfolgten Ansatzes ist die, einen Text auf eine Menge von Wikipedia-Artikeln abzubilden, die diesen Text hinreichend beschreiben. Wird dieser Ansatz nun auf weitere Texte angewandt, so können auf den daraus resultierenden charakteristischen Artikel-Mengen verschiedene Berechnungen durchgeführt werden, die schließlich in einem Maß für die Ähnlichkeit der zugrunde liegenden Texte münden. Die Abbildung des Textes auf Wikipedia-Artikel sowie die Bewertung der Ähnlichkeit solcher Abbildungen basiert dabei auf Wikipedias Hyperlink-Struktur. Diese Verweise stellen eine wertvolle Datenbasis dar, die Wissen sichtbar macht: Verweise werden explizit von den Autoren eines Artikels gesetzt und bilden dessen Verständnis von Zusammenhängen zwischen Artikeln bzw. Inhalten ab. Im Folgenden wird gezeigt, wie sich diese Verweise zur Berechnung der Ähnlichkeit von Texten nutzen lassen.

6.2 Konzeptueller Kontext eines Textes

Bevor nun in Kapitel 6.3 auf den eigentlichen Algorithmus eingegangen wird, soll zunächst die generelle Idee von Konzepten und Kontexten erläutert werden.

Ein *Konzept* sei definiert als ein Begriff, der – gegebenenfalls zusammen mit weiteren Begriffen – einen Text charakterisiert. Dies können entweder Begriffe sein, die wörtlich im Text vorkommen, oder aber Begriffe, die zum Text in einem gewissen Bezug stehen.

In der Folge sei daher das *in Bezug stehen* so definiert, dass ein Begriff genau dann mit einem Text in Bezug steht, wenn es im Wikipedia-Hyperlink-Graphen eine Verbindung zwischen dem den Begriff repräsentierenden Wikipedia-Artikel

und einem weiteren Artikel gibt, der den Text gemäß obiger Definition charakterisiert.

Ferner sei der *konzeptionelle Kontext*, oder auch der *Kontext der Konzepte* der folgende Graph

$$V = (N, E, \phi) \tag{6.1}$$

Dabei stellt N die Menge der Konzepte dar, welche den Text direkt repräsentieren oder mit einem der den Text direkt repräsentierenden Konzepte in Bezug stehen. E ist die Menge der gerichteten Kanten $\{(n_i, n_j) | n_i, n_j \in N\}$ und repräsentiert damit die Verweise zwischen den durch die Menge N repräsentierten Konzepten (=Wikipedia-Artikeln).

Offensichtlich hat nicht jedes dieser Konzepte eine gleich große Relevanz für den repräsentierten Text. Daher wird mittels einer Gewichtungsfunktion ϕ jedem $n_i \in N$ ein Gewicht zugeordnet, welches n_is Relevanz für den Text anhand der Relevanz für den Kontext V bestimmt. Dieses Maß ergibt sich aus der Anzahl der Verknüpfungen mit anderen $n \in N$ und dem Verhältnis zwischen eingehenden und ausgehenden Verweisen (siehe Gleichung 6.5 in Kapitel 6.3).

Abbildung 6.1 zeigt den *konzeptionellen Kontext* für den einzelnen Begriff „Brennstoffzelle“ (engl. „Fuel cell“). Jedes weitere Konzept ist Teil des Wikipedia-Hyperlinkgraphen, der um das Konzept „Fuel cell“ berechnet wurde. Färbung und Verknüpfungsdichte spiegeln dabei die Relevanz des jeweiligen Konzepts für den Gesamtgraphen (=Kontext) wider, wobei dunkler gefärbte Konzepte eine höhere Relevanz anzeigen. Um die Darstellung zu vereinfachen, wird in diesem Beispiel der Kontext lediglich eines Begriffes präsentiert. Die Berechnung für einen kompletten Text unterscheidet sich jedoch nicht prinzipiell, lediglich die Darstellung wird komplexer.

Die Idee des Ansatzes ist nun, dass sich aus zwei *ähnlichen Dokumenten* auch zwei *ähnliche Kontextgraphen* V_1 und V_2 ergeben. Um also die Ähnlichkeit

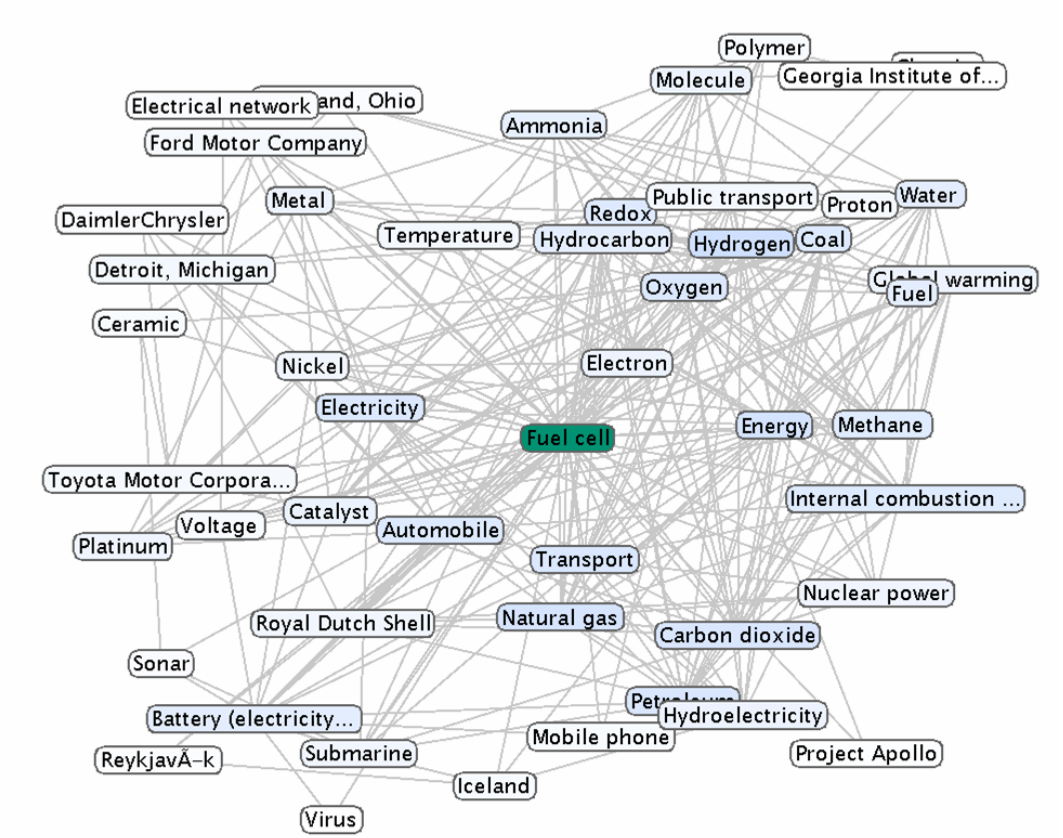

Abbildung 6.1: Konzeptueller Kontext für den Begriff „Brennstoffzelle“ (engl. „Fuel cell“)

zweiter Texte zu quantifizieren, kann dann auch die Ähnlichkeit dieser Graphen bestimmt werden:

$$similarity(d_1, d_2) := similarity(V_1, V_2) \tag{6.2}$$

Im Folgenden wird die genaue Berechnung des Graphen V erläutert und ein Ähnlichkeitsmaß für diese Graphen vorgeschlagen.

6.3 Algorithmus und Ähnlichkeitsmaß

Die Daten der Wikipedia sind in verschiedenen Formen erhältlich. Neben den reinen Artikeltexten können auch strukturelle Daten wie bspw. die Verknüpfung der einzelnen Artikel oder Themenzuordnungen in Form von MySQL-Dumps von der Wikipedia-Site herunter geladen werden[1]. Diese Daten bilden die Grundlage für die nachfolgenden Berechnungen. Auf die Struktur des verwendeten Datenbankschemas wird im Rahmen der Evaluierung näher eingegangen.

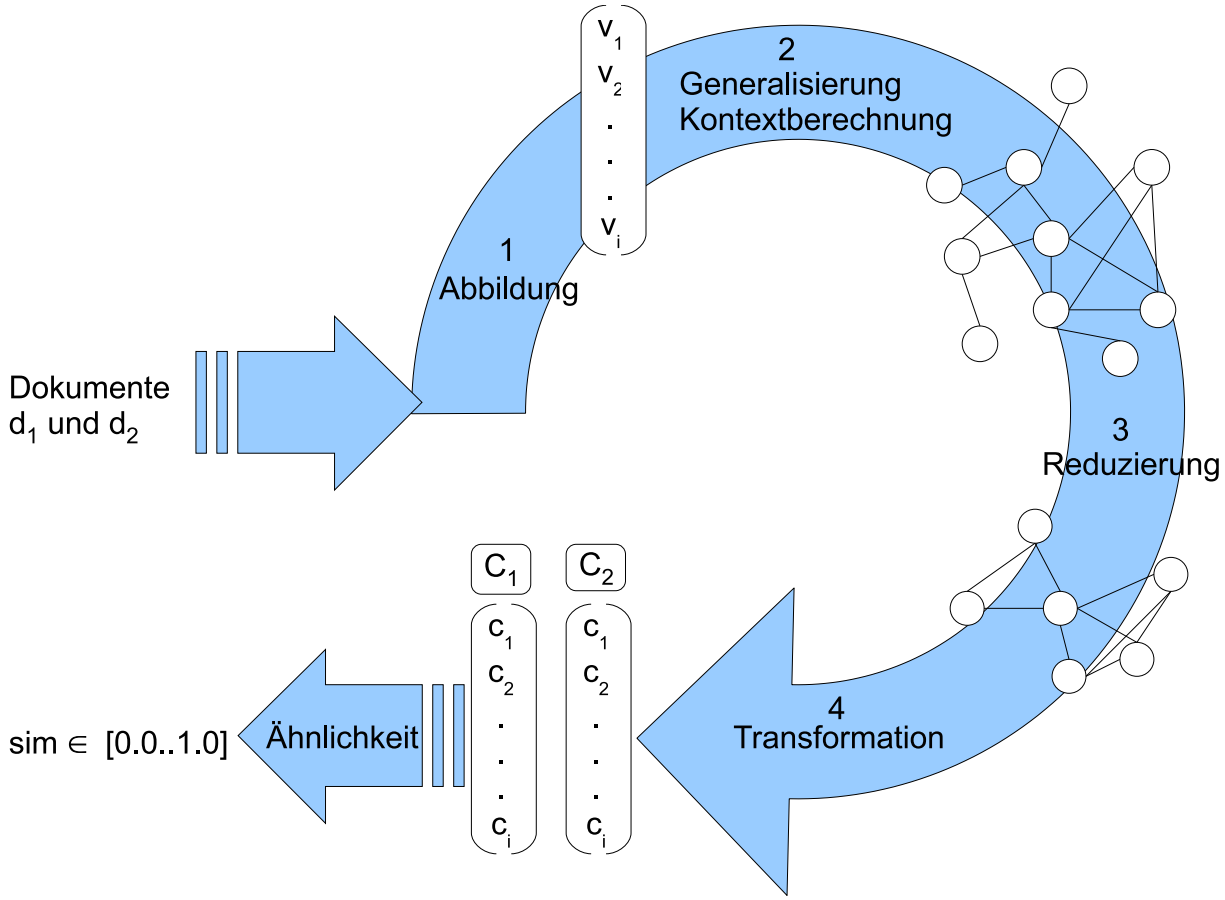

Abbildung 6.2: Schematische Darstellung des Konzept-Kontext Algorithmus

Abbildung 6.2 stellt den Prozess der Abbildung von Texten auf sog. „Konzeptvektoren“ und die Berechnung deren Ähnlichkeit dar. Die einzelnen Schritte werden nachfolgend erläutert.

[1] http://download.wikimedia.org/enwiki/

6.3.1 Schritt 1 – Abbildung auf Konzepte

In einem ersten Schritt wird ein Dokument d auf eine Menge von Wikipedia-Artikeln abgebildet und deren Relevanz für d bestimmt. Bei dieser Abbildung kommen zwei Techniken zum Einsatz: Zunächst werden all jene Wikipedia-Artikel ausgewählt, deren Titel im Text von d vorkommt. Diese werden in einen sog. Feature-Vektor im Vektorraum aller Wikipedia-Artikel-Titel überführt. Die skalaren Werte v_i des Vektors entsprechen dabei der Häufigkeit, mit der der Term i (=Titel des entsprechenden Wikipedia-Artikels) in d erscheint.

Bei manchen Texten, insbesondere bei sehr kurzen, kann es vorkommen, dass diese Art der Abbildung nur eine geringe Anzahl an Wikipedia-Artikeln ergibt und damit für eine sinnvolle weitere Verarbeitung nicht genügend Informationen zur Verfügung stehen. In diesem Fall wird eine alternative Abbildung gewählt. Mittels eines Invertierten Index [GF04] über dem textuellen Inhalt aller Wikipedia-Artikel werden die k Artikel ausgewählt, die auf rein syntaktischer Ebene die stärkste Ähnlichkeit mit d aufweisen. Details zu dieser syntaktischen Ähnlichkeitsberechnung finden sich in Kapitel 3.1.3.2, in [HG04] bzw. unter den Stichworten „$tf \cdot idf$“ und „Cosinus-Ähnlichkeit“ in der einschlägigen Literatur zum Thema Information Retrieval. Die skalaren Werte v_i ergeben sich dabei aus der $tf \cdot idf$-Auswertung auf dem Invertierten Index.

Beide Abbildungen zielen lediglich auf syntaktische Elemente ab und stellen keine Konzeptionalisierung dar. Sie sind daher nur ein erster Schritt und bilden die Grundlage für nachfolgende Berechnungen.

6.3.2 Schritt 2 – Berechnung des Kontexts

Im zweiten Schritt erfolgt die Konzeptionalisierung. D. h. es erfolgt eine Abkehr von der reinen Darstellung des Dokuments d durch Begriffe, die direkt im Text genannt werden. Zusätzlich wird nun der Kontext A berechnet. Dies geschieht indem die in Schritt 1 gefundenen Wikipedia-Artikel als Grundlage verwendet

und um weitere Wikipedia-Artikel ergänzt werden, die mit diesen gemäß obiger Definition *in Bezug stehen*.

Sei A die Menge aller im vorherigen Schritt identifizierter Wikipedia-Artikel a_i. Dann berechnet sich A' also wie folgt:

$$A' = A \cup \{a_n | \exists a_m((a_n, a_m) \in E \vee (a_m, a_n) \in E); a_n \in V, a_m \in A\} \tag{6.3}$$

6.3.3 Schritt 3 – Reduzierung des Kontexts

In Schritt 2 nimmt der Umfang der an der Berechnung beteiligten Daten exponentiell zu. Nach oben abgeschätzt werden kann dies mittels $|A'|^f$, wobei der Exponent f die durchschnittliche Anzahl der ein- und ausgehenden Verweise eines Wikipedia-Artikels ist. Auf dem im Rahmen dieser Arbeit verwendeten Datensatz lag dieser Wert bei $f \approx 20$. In der Praxis ist der Wert von f zwar geringer, da der so generierte Kontext einen Themenbereich repräsentiert und ein Großteil der Verknüpfungen auch innerhalb dieses Themenbereichs besteht, also weniger „neue“ Artikel hinzukommen, als dies zunächst den Anschein hat. Dennoch steigt der Umfang der Daten signifikant und muss für eine sinnvolle Weiterverarbeitung entsprechend reduziert werden. Dies geschieht, indem all die Konzepte bzw. Wikipedia-Artikel entfernt werden, die im Kontext von d nicht relevant sind. Dies ist hier derart definiert, dass genau die Konzepte als nicht relevant erachtet werden, die nicht über mindestens eine ein- und ausgehende Verbindung mit anderen Konzepten aus A' in Beziehung stehen.

$$A'' = \{a_n | \exists a_m \exists a_k((a_n, a_m) \in E \wedge (a_k, a_n) \in E); a_n, a_m, a_k \in A'\} \tag{6.4}$$

A'' wird verwendet, um den Kerngraphen $V'' = (A'', E'', \phi)$ zu definieren, also die kennzeichnende Menge der Konzepte und deren Verbindung. E'' ist die

Teilmenge von E, die aus allen Verbindungen $(a_n, a_m)|a_n, a_m \in A''$ besteht und ϕ bezeichnet wieder die Gewichtungsfunktion.

6.3.4 Schritt 4 – Gewichtung der Konzepte

In Schritt 3 wurden viele Konzepte aus dem Graphen entfernt, die mit einer hohen Wahrscheinlichkeit geringe Relevanz im Kontext von d haben. Dennoch ist auch die Relevanz der verbleibenden Konzepte nicht einheitlich. Daher wird im anschließenden Schritt 4 die Gewichtungsfunktion ϕ angewandt. Diese Funktion basiert auf der im IR-Bereich verbreitet eingesetzten $tf \cdot idf$-Formel und wurde auf die Verwendung im Rahmen der Kontext-Graphen angepasst.

$$\phi(a_i) = \alpha \cdot \frac{gTF_i}{gDF_i} + \beta \cdot \frac{tTF_i}{tDF_i}. \tag{6.5}$$

Sie enthält zwei Komponenten, die jeweils mit einem Faktor α bzw. β gewichtet werden. Der β-Teil entspricht dem klassischen $tf \cdot idf$ und ordnet jedem a_i den $tf \cdot idf$-Wert bezüglich des Dokuments d (tTF) und des Wikipedia-Textkorpus (tDF) zu.

Der α-Teil stellt die Anpassung der $tf \cdot idf$-Formel an den Kontext-Graphen dar. gTF_i sagt dabei aus, wie häufig a_i innerhalb des Graphen referenziert wird und stellt damit in erster Näherung dar, wie relevant a_i in diesem Kontext ist. Das geschieht analog zu „term frequency“ (tf) in klassischem $tf \cdot idf$, wo tf die Häufigkeit der Nennung eines Begriffs in einem Dokument quantifiziert. Wie im klassischen $tf \cdot idf$, so wird auch hier das gTF normalisiert. Dort geschieht das mittels der „document frequency“ (df), die aussagt, wie häufig ein Begriff in einer Dokumentensammlung auftritt, und damit, in wie weit er charakteristisch für ein Dokument sein kann. Hier quantifiziert gDF hingegen, wie häufig ein Konzept in der gesamten Wikipedia referenziert wird. Die Aussage ist dabei aber vergleichbar: je seltener ein Begriff/Konzept verwendet/referenziert wird, desto charakteristischer ist er/es, wenn es tatsächlich einmal genannt/verlinkt

wird. Für den gesamten α-Teil bedeutet das, dass ein Konzept dann gewichtig ist, wenn es innerhalb des Kontextes häufig referenziert wird, jedoch in der gesamten Wikipedia eher selten vorkommt, weil es dann als charakteristisch für genau den vorliegenden Kontext angesehen werden kann. Im Gegenzug wird ein Konzept als nicht sonderlich gewichtig angesehen, wenn es zwar im Kontext verschiedentlich referenziert wird, aber auch in der gesamten Wikipedia viele Verweise auf dieses Konzept existieren. In diesem Fall hat die Existenz des Konzepts im Kontext nur eine geringe Aussagekraft und das Konzept kann daher nicht als charakteristisch für den Kontext angesehen werden.

Das Ergebnis der Anwendung von ϕ ist der Konzept-Vektor
$C = (\phi(a_0), \ldots, \phi(a_k))^T$.

6.3.5 Schritt 5 – Konzeptuelle Ähnlichkeit

Die Schritte 1-4 führen zum Vektor C_1 für Dokument d_1. Sie werden für d_2 wiederholt und führen dabei zu C_2. C_1 und C_2 werden dann anhand der Cosinus-Ähnlichkeit miteinander verglichen (vgl. Kapitel 3.1.3.2). Das aus dieser Berechnung resultierende Ähnlichkeitsmaß für die Kontext-Vektoren dient nun als Maßstab für die Ähnlichkeit der zugrunde liegenden Dokumente.

Bei genauerer Betrachtung der Schritte 1-5 fällt auf, dass Schritt 3 konzeptionell nicht unbedingt nötig ist, da er kaum Einfluss auf die Qualität der Ergebnisse hat. Dies liegt in der Tatsache begründet, dass irrelevante Konzepte in Schritt 4 zwar nicht entfernt werden, jedoch eine schwache Gewichtung bekommen, also kaum in die folgenden Berechnungen einfließen. Dennoch hat sich die Anwendung von Schritt 3 in der Praxis als sinnvoll erwiesen, da durch die Reduzierung der an der weiteren Berechnung beteiligten Konzepte eine signifikante Verkürzung der Berechnungsdauer erzielt werden konnte.

6.4 Evaluation

Im Folgenden werden die Experimente beschrieben, anhand derer der Ansatz evaluiert wurde. Als Testdaten kamen drei Dokumentensammlungen zum Einsatz, die bei Evaluationen im Information Retrieval häufig verwendet werden: NPL (National Physical Laboratory, GB), LISA (Library and Information Science Abstracts)[1] und Reuters-21578[2]. Diese Sammlungen wurden ursprünglich zusammen getragen, um Suchmaschinen bzw. Suchansätze zu evaluieren, sie kommen also aus dem Umfeld der Dokumentensuche. Die Sammlungen NPL und LISA beinhalten zusätzlich zu den eigentlichen Texten weitere Daten in Form von Suchanfragen und entsprechenden Referenz-Ergebnismengen, die von Menschen zusammen gestellt wurden. Die Reuters-21578-Sammlung unterscheidet sich in dieser Hinsicht von LISA/NPL, da sie keine Suchanfragen mit zugehörigen Referenz-Ergebnissen bietet, sondern jedes Dokument einem oder mehreren vordefinierten Themen zuordnet.

6.4.1 Anmerkung zur Wahl der Textsammlungen

Die Textsammlungen NPL und LISA werden seit vielen Jahren für die Evaluation von Softwaresystemen im Bereich des Information Retrieval eingesetzt. Aufgrund deren verhältnismäßig geringen Umfangs gewinnen neuere Sammlungen in der Information Retrieval-Community zwar mehr und mehr Bedeutung und lösen die älteren zunehmend ab. Dennoch existieren gute Gründe, diese Sammlungen im Rahmen der vorliegenden Arbeit einzusetzen, die wichtigsten werden im Folgenden genannt und diskutiert.

1. Ein zentrales Argument für den Einsatz dieser Sammlung ist deren freie Verfügbarkeit. Die meisten neueren Sammlungen sind nur eingeschränkt

[1] NPL, LISA und andere Sammlungen sind verfügbar unter
http://ir.dcs.gla.ac.uk/resources/test_collections/

[2] Die Sammlung Reuters-21578 ist verfügbar unter
http://www.daviddlewis.com/resources/testcollections/reuters21578/

verfügbar – sie werden nur gegen Zahlung einer Gebühr zugänglich gemacht, unterliegen besonderen Nutzungsbedingungen oder sind manchmal sogar nur einer erlesenen Nutzerschaft vorbehalten. Diese Einschränkungen sind zwar in der Regel nachvollziehbar, bspw. handelt es sich um sensible Daten oder deren Zusammenstellung war zu teuer, um sie frei verfügbar zu machen. Dennoch schränkt das die Vergleichbarkeit zwischen verschiedenen Ansätzen stark ein, da nicht alle Systeme von ihren Entwicklern auf den gleichen Daten getestet werden können.

2. Da viele Untersuchungen im IR-Bereich auf die Analyse großer Datenmengen abzielen, ist offensichtlich, dass anhand kleiner Textsammlungen schwerlich Aussagen getroffen werden können über das Verhalten eines Algorithmus auf den heute in der Praxis üblichen Datenmengen. Dennoch existieren auch Szenarien, in denen es auf den Umfang der Sammlung nicht ankommt, wie bspw. im vorliegenden Fall eines Focused Crawlers. Hier werden Dokumente paarweise auf ihre Ähnlichkeit hin untersucht, im Unterschied zu anderen Szenarien, wie bspw. einer herkömmlichen Suchmaschine, wo ein Dokument gegen alle übrigen oder zumindest eine große Untermenge der gesamten Sammlung verglichen wird. Da für eine hier betrachtete Ähnlichkeitsberechnung aber immer nur zwei Dokumente untersucht werden müssen, hat die Größe der Sammlung keinen Einfluss auf die Qualität bzw. auf die Aussagekraft der Ergebnisse.

Auf den ersten Punkt beziehen sich auch folgende Anmerkungen zum *ESA*-Ansatz (vgl. Kapitel 3.1.3.6): *ESA* ist in diesem Zusammenhang der interessanteste Wettbewerber. Jedoch ist die Textsammlung, die zur Evaluierung von *ESA* genutzt wurde, nicht frei zugänglich. Dabei handelt es sich um eine – ebenfalls relativ kleine – Sammlung von 50 Nachrichtentexten der Australian Broadcasting Corporation (ABC), sowie detaillierte Ähnlichkeitsbewertungen, die von menschlichen Beurteilern abgegeben wurden. Frei zugänglich in [Pin04] sind zwar die eigentlichen Texte, die Bewertungen jedoch sind es nicht und keinem der Versuche, sie für eine Auswertung im Rahmen dieser Arbeit zu erhalten, war Erfolg beschieden. Ein direkter Vergleich von *ESA* und dem vorliegenden Algorithmus auf Basis der selben Daten ist daher nicht möglich.

6.4.2 Evaluationsszenario

Die verwendeten Dokument-Sammlungen enthalten keine explizite Information zur Ähnlichkeit der Dokumente. Dennoch können für NPL und LISA zwei Szenarien definiert werden, in deren Rahmen die Sammlungen für die Evaluierung von Algorithmen zur Berechnung von Dokumentenähnlichkeit verwendet werden können.

Szenario A Die Suchanfrage q_j wird mit den zugehörigen Referenzdokumenten d_i verglichen. Dazu werden sowohl die Anfrage, als auch die Referenzdokumente gemäß obiger Beschreibung in einen Konzept-Kontext überführt und auf dieser Ebene verglichen. Idealerweise sollten bei diesem Vergleich signifikant höhere Ähnlichkeitswerte ermittelt werden als beim Vergleich von q_j mit Dokumenten, die nicht zur Referenzmenge für q_j gehören.

Szenario B Die Dokumente der Referenzmenge einer Suchanfrage q_j werden auf der Ebene von Konzept-Kontexten untereinander verglichen. Auch hier sind signifikant höhere Ähnlichkeitswerte zu erwarten als beim Vergleich von Dokumenten, die in keiner Referenzmenge gemeinsam enthalten sind.

Szenario B ist dasjenige, das im Kontext dieser Arbeit die Herausforderungen und Ziele am besten wieder gibt: Aufgrund der Art und Weise, wie die Referenzmengen für solche Textsammlungen generiert werden, liegt Szenario A ein starker Fokus auf syntaktische Ähnlichkeit zugrunde: Menschen bekommen Anfragen und Dokumente zur Bewertung der Ähnlichkeit bzw. der Zugehörigkeit zur Ergebnismenge vorgelegt und führen diese Bewertung durch. Da es sich dabei um umfangreiche Dokumentensammlungen handelt, kann die Prüfung nur verhältnismäßig oberflächlich erfolgen. Ein inhaltlicher Vergleich kann dabei aus Komplexitätsgründen nicht durchgeführt werden. Statt dessen erfolgt in erster Näherung eine Beschränkung auf die Syntax, um ggf. in Grenzfällen auch auf den Inhalt einzugehen. Somit wird die Bewertung großer Anfrage- und Dokumentensammlungen möglich, allerdings auf Kosten der Präzision.

Der im Rahmen dieser Arbeit interessante Aspekt ist jedoch, wie ähnlich der entwickelte Algorithmus diejenigen Dokumente bewerten würde, die von Menschen als Ergebnis zu ein und der selben Suchanfrage klassifiziert wurden. Interessant deshalb, weil immer noch eine gewisse Wahrscheinlichkeit besteht, dass zwei Dokumente d_1 und d_2, die beide als Treffer zu Anfrage q klassifiziert wurden, dennoch nur eine verhältnismäßig geringe syntaktische Überlappung vorweisen. Da aber beide Dokumente zu Anfrage q passen, behandeln beide das selbe Thema und sollten daher eine *hohe Ähnlichkeit* besitzen. Im Gegenzug werden zwei Dokumente dann als *nicht ähnlich* angesehen, wenn sie nicht gemeinsam in der Referenzmenge zu einer beliebigen Anfrage q_i enthalten sind, da sie dann im Allgemeinen auch kein – durch eine Suchanfrage spezifiziertes – gemeinsames Thema behandeln.

Die Sammlung Reuters-21578 ermöglicht noch detailliertere Untersuchungen. Hierfür wurden Untermengen der Dokumente dergestalt gebildet, dass eine solche Untermenge all die Dokumente enthält, denen in der Referenz die selbe Menge an Themen zugeordnet wurde:

$$S = \{d_i, \ldots, d_j | topics(d_i) = \cdots = topics(d_j)\} \tag{6.6}$$

Diese Untermengen wurden anhand der Kardinalität der sie definierenden Themenmengen gruppiert, um die Qualität des Algorithmus hinsichtlich der Ähnlichkeitsberechnung von Dokumenten zu bewerten, welche ein, zwei, drei etc. gleichen Themen zugeordnet wurden.

$$SS_n = \{S_i | n = |topics(S_i)|\} \tag{6.7}$$

Auch hier wurden zwei Dokumente als *nicht ähnlich* angesehen, wenn sie kein gemeinsames Thema in der Referenz besitzen.

Da keine der drei Sammlungen explizit auf einer Ordinalskala angeordnete Ähnlichkeitswerte nennt, wurde eine binäre Klassifikation genutzt: *ähnlich/nicht ähnlich* gemäß der obigen Definition. Der Konzept-Kontext-basierte Ähnlichkeitsvergleich hingegen liefert Werte im Bereich $0.0 - 1.0$. Daher wurde ein Schwellwert t definiert, um die Bewertungen *ähnlich/nicht ähnlich* voneinander zu trennen. Dabei war t der einzige Parameter, der über die Auswertungen hinweg beim Wechsel der Sammlung geändert wurde. D. h. insbesondere, dass keine weiteren Optimierungen oder Anpassungen für eine Sammlung durchgeführt wurden.

Um zu evaluieren, ob der Algorithmus korrekt klassifiziert, wurde für jedes positive (=in der Referenz enthaltene) Dokumentenpaar ein zufälliges negatives Paar (=nicht in der Referenz enthalten) generiert.

Als Nulllinie wurden die Ergebnisse definiert, die von einem auf rein syntaktischer Ebene arbeitenden Algorithmus erzielt wurden. Dazu wurde ein Standard-Ansatz auf Basis von $tf \cdot idf$ mittels der Lucene-Bibliothek implementiert. Wie für die Auswertungen des Konzept-Kontext-Algorithmus, so wurde auch im Fall der Lucene-Auswertungen vor der Erzeugung des Index und bei der Anfrage im Vorfeld eine Reduktion auf die Wortstämme (stemming) durchgeführt.

Die Qualität des Algorithmus wurde dabei anhand dreier Metriken gemessen:

Korrelation mit der Referenz: Es wurde die Korrelation zwischen dem vorgeschlagenen Algorithmus bzw. der Nulllinie und den aus der Referenz abgeleiteten Vorgaben berechnet.

Anteil der korrekt klassifizierten Paare: Um zu quantifizieren, wie viele Dokument-Paare korrekt als *ähnlich/nicht ähnlich* bewertet wurden, wird der Anteil der korrekten Klassifizierungen wie folgt angegeben:

$$frac\_pos := \frac{pos}{ref\_pos} \tag{6.8}$$

$$frac\_neg := \frac{neg}{ref\_neg} \tag{6.9}$$

$$correct_{avg} := \frac{1}{2}(frac\_pos + frac\_neg) \tag{6.10}$$

pos gibt die Anzahl der korrekterweise als „ähnlich“ erkannten Paare an, ref_pos die Anzahl der in der Referenz als „ähnlich“ bewerteten Paare. $frac\_pos$ bezeichnet also den Anteil der vom jeweiligen Algorithmus korrekt erkannten „ähnlichen“ Paare. Analog dazu bezeichnet $frac\_neg$ mit neg und ref_neg den Anteil der vom jeweiligen Algorithmus korrekt erkannten „nicht ähnlichen“ Paare. $correct_{avg}$ gibt somit den über positive und negative Paare gemittelten Anteil der korrekten Ergebnisse des Algorithmus an.

Harmonisches Mittel der korrekten Klassifikationen: Zusätzlich wird auch das harmonische Mittel angegeben, um zu evaluieren, ob der Algorithmus auf *ähnlichen* oder *nicht ähnlichen* Dokumenten besser arbeite, oder ob die Ergebnisse auf beiden Domänen vergleichbar sind.

$$correct_{h\_mean} := \frac{2 \cdot frac\_pos \cdot frac\_neg}{frac\_pos + frac\_neg} \tag{6.11}$$

Letzteres ist Fall wenn der Wert von $correct_{avg}$ nahe bei $correct_{h\_mean}$ liegt, da dann keine Ausreißer in signifikantem Umfang enthalten wären, was eine Gleichmäßigkeit bestätigen würde.

6.4.3 Ergebnisse und Diskussion

Im Folgenden werden die Ergebnisse der Untersuchung in den beschriebenen Szenarien und Textsammlungen präsentiert und diskutiert.

6.4.3.1 LISA/NPL

Tabelle 6.1: Evaluationsergebnisse für LISA & NPL

		Szenario A			Szenario B		
	Alg.	Korrel.	∅	Harm.	Korrel.	∅	Harm.
LISA	Nulllinie	0.73	0.87	0.87	0.16	0.59	0.52
LISA	K-Kontext	0.44	0.71	0.70	0.25	0.64	0.60
NPL	Nulllinie	0.73	0.89	0.89	0.48	0.74	0.72
NPL	K-Kontext	0.64	0.82	0.82	0.59	0.80	0.78

Tabelle 6.1 zeigt die Ergebnisse der Evaluation anhand der Textsammlungen LISA und NPL. Die folgenden Beobachtungen werden diskutiert:

1. Der Konzept-Kontext-Algorithmus ist dem Nulllinien-Algorithmus in Szenario A unterlegen. In Szenario B jedoch liefert er bessere Ergebnisse.
2. Absolut gesehen ist die Qualität beider Algorithmen, des Konzept-Kontext-Algorithmus wie die des Nulllinien-Algorithmus, in Szenario A besser als in Szenario B.
3. Beide Algorithmen liefern auf der NPL-Sammlung bessere Ergebnisse als auf der LISA-Sammlung.

Wie aus 1) ersichtlich ist, hängt die Sinnhaftigkeit des Einsatzes des vorgeschlagenen Algorithmus stark davon ab, was der Benutzer erwartet. Der Algorithmus liefert dann schlechtere Ergebnisse als der Lucene-Ansatz, wenn er im Rahmen von Szenario A eingesetzt wird. Andererseits liefert er im Rahmen von Szenario B deutlich bessere Ergebnisse als der Nulllinien-Algorithmus. Verantwortlich dafür ist die Nutzung von abstrakten Konzepten gegenüber der Nutzung konkreter Begriffe, welche lediglich eine syntaktische Ähnlichkeit aufdecken würden.

Auch wenn zwei Dokumente unterschiedliche Aspekte ein und des selben Themas behandeln, ein unterschiedliches Vokabular benutzen etc., so findet der hier vorgeschlagene Algorithmus dennoch Übereinstimmungen auf konzeptioneller Ebene. Abhängig von der Dokumentensammlung kann dies die Ergebnisqualität deutlich verbessern.

Punkt 2) bestätigt die Erwartungen an den Algorithmus. Jedes Referenzdokument d_i zur Anfrage q_j fokussiert möglicherweise auf einen anderen Teilaspekt von q_j. Bspw. behandelt d_1 den Aspekt 1 der Anfrage q, d_2 behandelt Aspekt 2 und kein weiteres Dokument ist relevant im Kontext von q. Dann würden d_1 und d_2 in Szenario A hohe Wertungen bekommen. In Szenario B hingegen wären diese Wertungen im Allgemeinen niedriger, da beide Dokumente jeweils auf einen eigenen Teilaspekt von q fokussieren. Die Herausforderung besteht jedoch darin, zu erkennen, dass sie überhaupt ein gemeinsames – evtl. übergeordnetes – Thema haben. Offensichtlich wird damit ein gewisses Maß an Unsicherheit eingeführt, was sich auch in niedrigeren Korrelationswerten in Szenario B zeigt. Das bedeutet, dass – insbesondere in Szenario B – der Algorithmus keinesfalls perfekte Ergebnisse liefert. Dennoch sind die Ergebnisse bei Weitem besser als die des rein syntaxbasierten Nulllinien-Algorithmus.

Die Beobachtung 3), dass beide Algorithmen auf der NPL-Sammlung bessere Ergebnisse liefern als auf der LISA-Sammlung, lässt sich folgendermaßen erklären: Viele der Anfragen in LISA behandeln mehrere Themen im Stile von „In meiner Dissertation beschäftige ich mich mit A. Das beinhaltet B, C und D. Auch E ist wichtig. Gib mir eine Liste der relevanten Veröffentlichungen“. Daher ist sowohl in Szenario A, als auch in Szenario B die Referenzmenge sehr heterogen im Sinne der syntaktischen oder semantischen Fokussiertheit. Die Anfragen in NPL hingegen besitzen nicht diese Komplexität und haben daher eine homogenere Referenzmenge, die von beiden Algorithmen leichter reproduziert werden kann. Das Problem der Ähnlichkeitsberechnung ist also auf NPL einfacher als auf LISA, weshalb dort auch naturgemäß die Ergebnisse besser sind.

Aus den unterschiedlichen Ergebnissen auf NPL und LISA lässt sich ableiten,

dass die Ergebnisqualität maßgeblich von der Dokumentensammlung abhängt. Daher ist der Vergleich verschiedener Algorithmen auf unterschiedlichen Dokumentensammlungen problematisch und birgt wenig Aussagekraft. Wie oben bereits angeführt, gelang es im Rahmen dieser Arbeit nicht, an die Daten zu gelangen, die zur Evaluierung des interessantesten Wettbewerbers *ESA* verwendet wurden. Daher lassen sich auch die Ergebnisse nicht direkt miteinander vergleichen. Gabrilovich und Markovitch [GM07] geben Korrelationswerte von ~ 0.7 für *ESA* an. Ihre Dokumentensammlung besteht aus 50 Nachrichtenartikeln zu verschiedenen Themen. Es kann davon ausgegangen werden, dass der hier vorgestellte Algorithmus auf dieser Sammlung deutlich bessere Ergebnisse liefert als auf NPL, da NPL im Wesentlichen aus Dokumenten zum Thema Physik und Elektrotechnik besteht. D. h. die Dokumente in NPL besitzen von sich aus eine gewisse Grundähnlichkeit, was die Bewertung verkompliziert. Im Falle der *ESA*-Sammlung sind die Themen der Dokumente deutlich breiter gefächert, was offensichtlich Komplexität aus der Bewertung nimmt und somit bessere Ergebnisse erwarten lässt. Dennoch soll an dieser Stelle angemerkt werden, dass es sich bei dieser Aussage um eine theoretische Annahme handelt, die nicht ohne Zugriff auf die gesamten Daten der *ESA*-Sammlung und die Bewertungen belegt werden kann.

6.4.3.2 Reuters-21578

Tabelle 6.2: Evaluationsergebnisse für Reuters-21578

	Nulllinie			K-Kontext		
Themen	Korrel.	∅	Harm.	Korrel.	∅	Harm.
1	0.18	0.60	0.59	0.53	0.74	0.73
2	0.25	0.63	0.63	0.56	0.78	0.77
3	0.43	0.73	0.72	0.63	0.81	0.80
4	0.46	0.74	0.72	0.74	0.87	0.87
5	0.50	0.76	0.73	0.74	0.86	0.85
(6)	0.65	0.81	0.77	0.82	0.91	0.90
(7)	1.00	1.00	1.00	1.00	1.00	1.00

Tabelle 6.2 zeigt die Ergebnisse der Evaluation auf der Dokumentensammlung Reuters-21578. Sowohl der hier vorgeschlagene Algorithmus, als auch der Nulllinien-Algorithmus entsprechen den Erwartungen, bessere Ergebnisse auf den Dokumentenklassen zu erzielen, die sich durch mehrere Themenannotationen definieren. Das ist darin begründet, dass es für einen Algorithmus dann tendenziell einfacher ist, die Charakteristik eines Textes zu bestimmen, wenn auch ein menschlicher Beurteiler eine präzise Charakterisierung vornehmen kann. Im Falle der Reuters-21578-Sammlung also dann, wenn menschliche Beurteiler den Text präzise anhand der in der Sammlung vorgegebenen Schlagworte klassifiziert haben. Wenn jedoch ein solcher Beurteiler einem Text lediglich ein oder zwei generelle Themen/Schlagworte zuordnen kann, dann wird auch ein Algorithmus, der das menschliche Verständnis von Dokumentenähnlichkeit modelliert, Schwierigkeiten mit dieser Aufgabe haben. Auch auf dieser Sammlung wird anhand der Evaluationsergebnisse deutlich, dass hier der Konzept-Kontext-Algorithmus deutlich bessere Ergebnisse liefert, als der rein syntaxbasierte Nulllinien-Algorithmus, da er das menschliche Verständnis von Dokumentenähnlichkeit besser widerspiegelt.

Anmerkung: Die Werte für die Klassen SS_6 und SS_7 werden als irrelevant angesehen, da die Anzahl der Dokumente, die genau sechs bzw. sieben Themen gemeinsam haben zu klein ist und die Ergebnisse somit nicht signifikant sind.

6.4.4 Aussagen zur Performanz

Als Ergänzung zu den oben genannten qualitativen Untersuchungen soll im Folgenden eine Diskussion von Performanz-Aspekten erfolgen. Neben den qualitativen Aspekten stellen diese ein weiteres Kriterium für den erfolgreichen Einsatz des Konzeptes in der Praxis dar. Zunächst wird die Testumgebung beschrieben. Im Anschluss erfolgt die Diskussion von Messergebnissen.

6.4.4.1 Testumgebung

Als Testsystem kam die folgende Kombination aus Hard- und Software zum Einsatz:

- 2x Intel Xeon 3,2GHz mit 2 MB Cache, 3 GB Hauptspeicher
- Linux Kernel 2.6.9
- MySQL-Datenbankserver 4.1.22, MyISAM-Engine. Abweichend von den Standard-Einstellungen: 500 MB Index-Cache (Default: 8 MB)
- DB-Dump der englischsprachigen Wikipedia vom 26.09.2006 und vom 24.05.2008

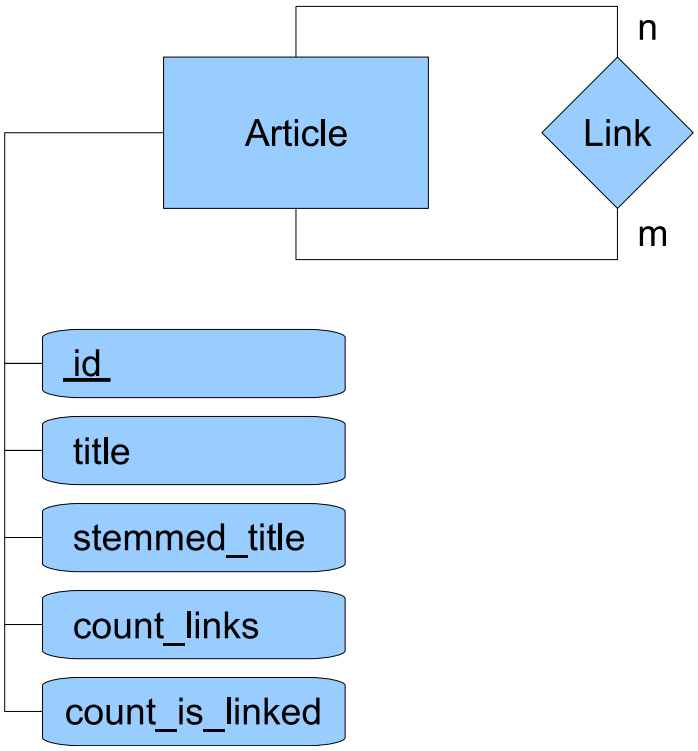

Abbildung 6.3: Datenbankschema

Abbildung 6.3 zeigt das den Berechnungen zugrunde liegende Datenbankschema. Tabelle `Article` listet alle Wikipedia-Artikel. Tabelle `Link` verwaltet die gerichteten Verweise zwischen den in `Article` beschriebenen Wikipedia-Artikeln.

Die Tabelle `Article` besteht aus 5,2 Mio. Einträgen, wobei gegenüber den ursprünglichen Wikipedia-Inhalten eine Reduzierung vorgenommen wurde: es wurden alle Einträge zu Jahreszahlen oder Tagen entfernt. Dieser Schritt beschleunigt die Berechnung, da Verknüpfungen von zwei Artikeln über derartige Brückenartikel häufig sind und somit in der Berechnung auch entsprechend häufig Berücksichtigung finden. Werden sie ignoriert, so führt dies zu keinen messbaren Qualitätseinbußen, da die meisten dieser Verknüpfungen eine Brücke über in zwei Artikeln genannten Jahreszahlen darstellen und die Semantik derartiger Verknüpfungen im Kontext der Bestimmung der inhaltlichen Ähnlichkeit zweier Texte meist irrelevant ist.

Die Tabelle `Link` ist mit über 100 Mio. Einträgen um ein Vielfaches größer. Diese Einträge repräsentieren die Verknüpfungen der 5,2 Mio. `Article`-Einträge. Gegenüber den ursprünglichen Wikipedia-Daten wurden jedoch Redirects aufgelöst. Redirects sind Einträge in der Wikipedia, denen kein Text zugeordnet ist, sondern die lediglich einen Verweis auf einen anderen Eintrag darstellen. Diese Technik wird u. a. zur Beschreibung und Verwaltung von Synonymen eingesetzt. Um die Berechnung zu beschleunigen, erfolgte eine Auflösung der Redirects dahingehend, dass bei Verweisen auf ein Redirect als neues Ziel der Inhalt des Redirects gesetzt wurde. Zur weiteren Beschleunigung der Netz- bzw. Vektor-Berechnungen wurden in der Tabelle `Article` die Werte der Attribute `count_links` und `count_is_linked` vorab berechnet. Sie geben für jeden Eintrag an, wie viele Verweise von diesem Eintrag aus auf weitere Einträge existieren bzw. wie viele andere Einträge auf diesen verweisen.

Um die Extraktion von Features aus einem Text zu beschleunigen, wurde in der Tabelle `Article` zudem ein weiteres Attribut vorab berechnet: `stemmed_title` enthält den auf den Wortstamm reduzierten Artikel-Titel (`title`). Besteht der Titel aus mehreren Begriffen, so erfolgt die Wortstammreduktion für jeden einzelnen Begriff und `stemmed_title` ergibt sich aus der Konkatenierung der einzelnen Wortstämme. Damit können im Text anhand von Datenbanksuchen sowohl Einwort- als auch Mehrwort-Terme identifiziert werden.

Um diese Datenbanksuche effizient durchführen zu können, wurde neben dem Primärschlüssel `id` zudem ein Index auf `stemmed_title` erstellt. Für die effiziente Berechnung von Nachbarschaften wurden zusätzliche Indexe auf der Tabelle `Link` erstellt: `from(to)` sowie `to(from)`.

6.4.4.2 Untersuchungen

Die folgenden beiden Untersuchungen zur Geschwindigkeit und Parallelisierbarkeit des Algorithmus werden diskutiert:

1. Abhängigkeit des Berechnungsdauer von Größe und weiteren Eingeschaften des zu analysierenden Textes
2. Parallelisierbarkeit und die Nutzung von Multiprozessorsystemen

Untersuchung 1 – Größe des Textes/der initialen Feature-Menge

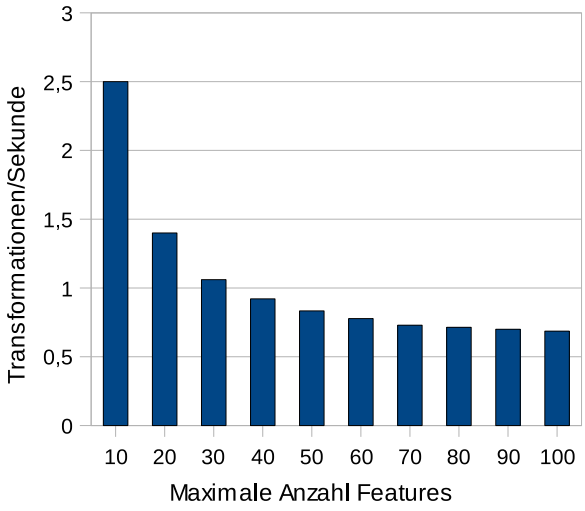

Abbildung 6.4: Durchsatz in Abhängigkeit der Anzahl erkannter Features im Text

Die Länge eines Textes bzw. dessen Informationsgehalt, hier quantifiziert durch die Anzahl der extrahierbaren Features, hat einen wesentlichen Einfluss auf die Geschwindigkeit der Berechnung des zugehörigen Vektors. Die Textgröße selbst stellt in diesem Zusammenhang keinen geeigneten Eingabeparameter dar, da

der größte Teil des Aufwandes bei der Berechnung der Beziehungen zwischen im Text enthaltenen Features und deren Wikipedia-Nachbarschaft entsteht. Die Anzahl der aus einem Text extrahierten Features hängt aber nicht allein von der Länge des Textes ab, sondern zudem von Art und Stil des Textes.

Aus diesem Grund wird die Länge des Textes in dieser Untersuchung nicht betrachtet und statt dessen direkt die Anzahl der extrahierten Features als Eingabeparameter verwendet. Aus Abbildung 6.4 wird ersichtlich, dass erwartungsgemäß eine kleine Anzahl Features eine schnelle Berechnung ermöglicht, bei Zunahme der Featureanzahl die Dauer der Berechnung jedoch stark ansteigt. Dies erklärt sich aus der in erster Näherung exponentiellen Zunahme der in der weiteren Berechnung zu berücksichtigenden Menge an Knoten, die mit den Knoten der initialen Featuremenge verlinkt sind (vgl. 6.3, Schritt 2). Letztlich wird dieser exponentielle Anstieg jedoch gedämpft, da die durchschnittliche Anzahl Knoten, die ein weiteres Feature als Nachbarn in die Berechnung neu einbringt mit zunehmender Anzahl Features sinkt. Diese Tatsache begründet sich darin, dass alle Features eines Textes zusammen das Thema des Textes repräsentieren. Wenige Features geben bereits den groben Kontext wieder und zusätzliche Features helfen, diesen Kontext zu präzisieren. Zwangsläufig bringen sie aber Redundanzen in Form von gemeinsamen verlinkten Knoten mit sich. Diese Redundanzen werden in der weiteren Berechnung erkannt und berücksichtigt, was sich positiv auf die Dauer der Berechnung auswirkt.

Für die Anwendbarkeit des Ansatzes auf praxisrelevanten Texten stellt dies eine gewisse Einschränkung dar. Sollen auch längere Texte bzw. Texte mit hohem Informationsgehalt, d.h. Texte, aus denen sich viele Features extrahieren lassen, verarbeitet werden, so müssen zusätzliche Anstrengungen unternommen werden, um die Verarbeitungsdauer zu reduzieren. Zwei Ansätze haben sich als zielführend heraus gestellt:

- Die aus einem Text extrahierten Features tragen alle in unterschiedlichem Maße zur Beschreibung des Textes bei. Aus diesem Grund wird bei der Berechnung des Vektors eine Gewichtungsfunktion eingesetzt, um diesem Sachverhalt Rechnung zu tragen. Diese Eigenschaft der Features bzw.

des Textes kann sich aber auch dafür zu Nutze gemacht werden, im Vorfeld der eigentlichen Netz- bzw. Vektorberechnung eine Selektion der Features durchzuführen. Da das Ziel eine Reduzierung der Featuremenge ist, besteht ein Ansatz darin, diejenigen Features zu selektieren, die den Text am besten charakterisieren. Ein Maß für diese Eigenschaft kann in Anlehnung an $tf \cdot idf$ bzw. die ϕ-Funktion aus Kapitel 6.3 definiert werden: Die tf-Komponente spiegelt die Häufigkeit wider, mit der ein Feature im Text vorkommt. Die df-Komponente hingegen stellt ein Maß dafür dar, wie relevant das Feature im Kontext der gesamten Wikipedia ist und damit, wie hoch seine Kennzeichnungskraft ist. Diese Berechnung kann mit minimalem Aufwand erfolgen, da sie neben der einfachen Häufigkeitszählung lediglich einen Datenbank-Lookup und eine einfache Multiplikation für jedes Feature erfordern (die df-Werte aller Features sind vorberechnet). Werden die Features nun nach ihrer auf diese Art berechneten Relevanz sortiert, so können die n Relevantesten für die weitere Berechnung ausgewählt werden.

  Es bleibt dennoch festzuhalten, dass diese Reduzierung offensichtlich nicht beliebig einsetzbar ist. Jede Reduzierung geht mit einem Informationsverlust einher und hat daher Einfluss auf die Qualität des Ergebnisses.
- Um diesen Nachteil zu kompensieren, kann je nach Einsatzszenario alternativ ein zweistufiger Ansatz angewandt werden. In einem ersten Schritt werden die Features der Texte extrahiert und die Vektoren anhand einer kleinen Anzahl dieser Features berechnet. Aufgrund der wenigen Features kann dieser Berechnungsschritt schnell im Vergleich zur vollen Berechnung durchgeführt werden. Da aber, wie oben angeführt, mit dieser Reduzierung ein Informationsverlust und damit eine verminderte Ergebnisqualität einhergehen, wird dieses Ergebnis lediglich zur Vorfilterung für eine weitere Berechnungsrunde verwendet. Die Evaluierung hat gezeigt, dass selbst bei Verwendung von lediglich zehn bis 20 Features eine sehr gute Unterscheidung zwischen “nicht ähnlich” und “ähnlich” erreicht wird. Lediglich der Grad der Ähnlichkeit kann mit einer höheren Anzahl Features genauer bestimmt werden. Aus diesen Beobachtungen ergibt

sich der Ansatz, lediglich für diejenigen Texte eine detaillierte Berechnung durchzuführen, deren Relevanzwerte nach der ersten Berechnungsstufe über einem bestimmten Grenzwert liegen.
Eine weitere Steigerung der Berechnungsgeschwindigkeit kann erreicht werden, indem der erste Schritt nicht anhand des hier diskutierten Algorithmus berechnet wird, sondern auf rein syntaxbasierten Verfahren wie dem klassischen $tf \cdot idf$ in Kombination mit der Cosinusähnlichkeit baut. Damit kommen zwar die oben diskutierten Vorteile dieses Ansatzes nicht in vollem Umfang zum Zuge, für eine einfache Vorfilterung sind jedoch die Ergebnisse syntaxbasierter Verfahren in vielen Suchszenarien hinreichend gut. Texte, für die sich in dieser Stufe eine über einem bestimmten Grenzwert liegende Ähnlichkeit ergeben hat, werden dann in der nachfolgenden Stufe wieder mit dem aufwändigeren aber auch detaillierteren hier diskutierten Verfahren analysiert.

- Wird das zweistufige Verfahren aus Punkt 2 mit dem Algorithmus der Konzeptuellen Kontexte in beiden Stufen eingesetzt, dann ist offensichtlich eine Kombination mit der Feature-Selektion aus Punkt 1 sinnvoll, um die Ergebnisqualität des Vorfilters zu steigern.

Der relevante Aufwand fällt alleine bei der Berechnung der die Dokumente repräsentierenden Vekoren an. Die anschließende Berechnung der Ähnlichkeit anhand der Cosinus-Ähnlichkeit der Vektoren erfordert nur wenige Millisekunden und fällt somit nicht ins Gewicht. Sie wird daher an dieser Stelle nicht gesondert betrachtet.

Untersuchung 2 – Parallelisierbarkeit

Abbildung 6.5 zeigt die Anzahl der Transformationen $d \rightarrow C_d$ pro Sekunde in Abhängigkeit der Anzahl parallel gestarteter Prozesse. Da es sich bei der Testumgebung um ein Doppelprozesser-System handelt, entspricht die Steigerung des Durchsatzes bei Nutzung von zwei Prozessen (1,13 Transformationen/Sekunde) gegenüber lediglich eines Prozesses (0,65 Transformationen/Sekunde)

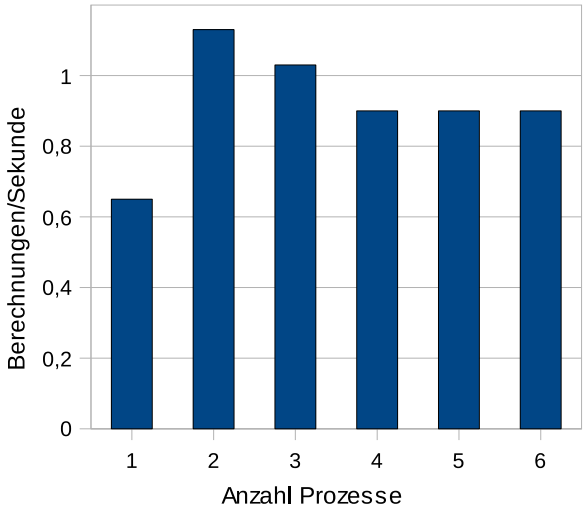

Abbildung 6.5: Durchsatz in Abhängigkeit der Anzahl paralleler Prozesse

den Erwartungen. Die Steigerung des Durchsatzes um 74% liegt unter der Idealmarke von 100% und erklärt sich aus dem höheren IO-Aufwand bei mehreren parallelen Prozessen, wodurch der Festspeicher zur Performanz bestimmenden Komponente wird. Werden weitere Prozesse parallel gestartet, so sinkt der Durchsatz wieder auf einen konstanten Wert von 0,9 Transformationen pro Sekunde. Dieser Rückgang erklärt sich durch höhere Kosten für das Prozessmanagement sowie aus der Tatsache, dass der hier verwendete Datenbankserver MySQL keinen konfigurierbaren Datencache verwaltet und statt dessen auf den Dateisystem-Cache des Betriebssystems zurückgreift. Bei der Nutzung mehrerer paralleler Prozesse werden damit aber auch größere Datenvolumina prozessiert, was zu vermehrten Cache-Misses führt und somit zusätzlichen IO-Aufwand produziert. Der dritte Prozess nimmt jedoch eine Sonderrolle ein. Hier fällt die Reduktion des Durchsatzes geringer aus im Vergleich zu vier und mehr Prozessen. Der Grund dafür ist in der Tatsache zu sehen, dass der dritte Prozess CPU-Zeit nutzen kann, die bei lediglich zwei Prozessen aufgrund von IO-Wartezeiten ungenutzt bliebe. Zwar tritt auch hier die Cache-Problematik zu Tage, jedoch wirkt sie sich nicht so stark aus wie bei einer weiteren Steigerung der Parallelität.

6.4.4.3 Zusammenfassung der Performanz-Messungen

Die Ergebnisse der Performanz-Messungen lassen sich wie folgt zusammenfassen:

- Der hier vorgestellte Ansatz eignet sich insbesondere für den Einsatz in Szenarien wie den in Kapitel 4.2.2.2 vorgestellten Focused Crawling Szenarien. Diese Art von Suche findet typischerweise unter starker Begrenzung der verfügbaren Bandbreite statt, was in einer geringen Anzahl von Texten resultiert, die in einem bestimmten Zeitraum verarbeitet werden müssen. Beim motivierenden Szenario handelt es sich um kleine und mittelständige Unternehmen, die lediglich einen Teil der Bandbreite der Internetanbindung für langfristig angelegte Suchen wie einen Focused-Crawler-Lauf reservieren können. Vor diesem Hintergrund betrachtet, ist ein Durchsatz von zwei Dokumenten pro Sekunde akzeptabel. Insbesondere vor dem Hintergrund der zunehmenden Verbreitung von Mehrprozessor- und Mehrkern-Architekturen sind die notwendigen Berechnungen auch auf im typischen Büroumfeld eingesetzten PCs effizient durchführbar.
- Im Rahmen von herkömmlichen Suchmaschinen hingegen erscheint der Ansatz nicht praktikabel. Selbst bei massiver Parallelisierung der Netz- bzw. Vektorberechnung können die heute von typischen Suchmaschinen indexierten Dokumentenmengen nicht in akzeptabler Zeit verarbeitet werden.
- Dort wo eine Steigerung des Durchsatzes nötig ist oder umfangreiche Texte verarbeitet werden, können mehrstufige Lösungen zum Einsatz kommen, die den durchschnittlichen Durchsatz steigern, ohne in nennenswerten Qualitätseinbußen zu resultieren.

6.5 Zusammenfassung

In den vorangegangenen Abschnitten wurde ein Ansatz zur Berechnung der Ähnlichkeit von Texten vorgestellt. Dieser Ansatz geht über die Berechnung der rein syntaktischen Ähnlichkeit hinaus und liefert ein Maß für die inhaltliche Ähnlichkeit. Diese inhaltliche Ähnlichkeit bestimmt sich dabei aus der Ähnlichkeit der über den Texten berechneten Wikipedia-Artikel-Graphen. Sie bietet jedoch insbesondere kein Maß für die semantische Ähnlichkeit der Texte, was aber für den hier vorgestellten Anwendungsfall nicht nur keine Einschränkung, sondern vielmehr ein gewünschtes Verhalten darstellt.

Neben einer detaillierten Beschreibung des Algorithmus zur Berechnung dieser Graphen und des zur Bestimmung der Ähnlichkeit von Texten eingesetzten Maßes, wurden qualitative und quantitative Messungen präsentiert, die die Nutzbarkeit dieses Ansatzes bestätigen bzw. ihre Grenzen aufzeigen.

KAPITEL 7

ERKENNUNG VON NAMENSNENNUNGEN IN TEXTEN

In den beiden vorangegangenen Kapiteln 5 und 6 wurde diskutiert, wie Informationsquellen im Web aus technischer Sicht erschlossen werden können und wie eine Filterung der relevanten Dokumente erfolgen kann. Diese beiden Schritte sind aus Benutzersicht jedoch lediglich als Vorbereitung für den eigentlichen Schritt der Informationsextraktion zu sehen. Dieses Problem der Extraktion strukturierter Informationen aus unstrukturierten Texten wird im Folgenden diskutiert und rundet somit das Konzept der Softwareplattform für komplexe Suchprobleme ab.

Die Extraktion von Information aus Texten oder Dokumenten ist eine zentrale Aufgabe im Bereich Information Retrieval. Die in einem Text enthaltenen Informationen sind dabei unterschiedlicher Natur, und nicht alle diese Informationen lassen sich Stand heute mittels automatisierter Verfahren auch vollständig extrahieren – bspw. ob eine Person einen Sachverhalt gutheißt oder nicht (Sentiment Detection, vgl. Kapitel 3.2). In anderen Bereichen, insbesondere wo es um die

Erkennung einfacher Fakten geht, funktioniert dies hingegen besser. Ein solcher Bereich ist die Named Entity Recognition (NER). Dabei geht es darum, die Nennung von Entitäten wie Personen-, Firmen- oder Ortsnamen, Telefonnummern, Adressen etc. in einem Text zu identifizieren. Dem menschlichen Leser fällt diese Aufgabe im Allgemeinen nicht schwer, da er anhand des Kontextes oder mittels Methoden der Mustererkennung schnell und einfach die Nennung von Namen, Orten etc. in einem Text erkennt. Doch auch auf diese Art und Weise nicht eindeutig identifizierbare Nennungen stellen für den menschlichen Leser eines Textes kein Problem dar. Aufgrund seines Wissens über die verwendete Sprache kann der Mensch den Inhalt eines Textes verstehen und einzelnen Begriffen eine Bedeutung zuordnen – bspw. *Ort*, *Person* oder *Telefonnummer*. Software hat dieses Sprachverständnis im Allgemeinen nicht in dem hierfür benötigten Umfang, weshalb eine derartige Analyse meist nicht die Ergebnisqualität erreicht, die ein Mensch erreichen würde. Andererseits ist die Verarbeitungsgeschwindigkeit eines solchen Softwaresystems ungleich höher als die eines Menschen, weshalb oft zu Gunsten des höheren Durchsatzes auf solche Systeme zurück gegriffen wird und somit Kompromisse zwischen Qualität und Quantität eingegangen werden.

Prinzipiell lassen sich die bei der Erkennung solcher Entitäten von Software bzw. auch von Menschen gemachte Fehler in zwei Klassen teilen:

1. Es werden nicht alle Nennungen von Entitäten in einem Text erkannt
2. Wörter oder Wortsequenzen werden als Entitäten identifiziert, obwohl sie eigentlich keine solche repräsentieren

Die Reduzierung der Erkennungsfehler einer Klasse führt dabei in der Regel zu einer Zunahme der Fehler in der anderen Klasse, da es sich um gegenläufige Optimierungsziele handelt: wenn mehr Entitätennennungen erkannt werden sollen, so steigt auch die Gefahr, fälschlicherweise Begriffe als Entitäten zu identifizieren, die keine sind. Anders herum werden manche Begriffe nicht als Entitäten erkannt wenn darauf Wert gelegt wird, dass das System nur diejenigen Begriffe zurück meldet, bei denen es ganz sicher ist, dass es sich um Entitätennennungen handelt.

Bewerten lassen sich solche Systeme durch den Vergleich ihrer Ergebnisse mit einer Referenz. Diese Referenz setzt sich aus Texten zusammen, die manuell oder halbautomatisch mit manueller Nacharbeit auf Entitätennennungen hin untersucht und um diese Informationen in Form von zusätzlichen Metadaten ergänzt wurden. Die Erkennungsleistung eines solchen Systems kann nun mit der im Information Retrieval weit verbreiteten Metriken *Precision* und *Recall* bewertet werden:

Precision Die Precision gibt an, wie verlässlich die Ergebnisse eines Systems im Hinblick auf fälschlicherweise erkannte Textbereiche sind. Sie setzt dabei den Anteil der vom System korrekt identifizierten Nennungen ins Verhältnis zu der Gesamtzahl der vom System gefundenen vermeintlichen Nennungen.

$$P = \frac{TP}{TP + FP} \tag{7.1}$$

Die Anzahl der korrekt identifizierten Nennungen wird als *korrekt positiv* bzw. *True Positive (TP)* bezeichnet. Die Gesamtzahl der vom System gefundenen vermeintlichen Nennungen hingegen setzt sich aus den *korrekt positiv* und den *falsch positiv* (*False Positive (FP)*) gewerteten Nennungen zusammen. Letztere bezeichnen die Anzahl der vom System fälschlicherweise als Entitäten-Nennungen identifizierten Textstellen. Eine hohe *Precision* bedeutet somit, dass es unter allen identifizierten vermeintlichen Nennungen nur wenige gibt, die tatsächlich keine Nennungen sind. Sie stellt damit ein Maß für die Genauigkeit des Systems dar.

Recall Der Recall hingegen setzt den Anteil der vom System korrekt identifizierten Nennungen ins Verhältnis zur Anzahl der in der Referenz hinterlegten Nennungen.

$$R = \frac{TP}{TP + FN} \tag{7.2}$$

Neben den für die Berechnung der *Precision* bereits diskutierten Maßen *TP* und *FP* wird hierzu ein weiteres Maß *FN* eingeführt. Es bezeichnet die Anzahl der in der Referenz hinterlegten Nennungen, die jedoch vom betrachteten System fälschlicherweise nicht erkannt wurden. Sie werden als *falsch negativ* bzw. *False Negative (FN)* bezeichnet. Der *Recall* ist damit ein Maß dafür, welcher Anteil aller in einem Text vorhandenen Nennungen von einem System erkannt werden.

Wie oben bereits angedeutet, stellen Precision und Recall zwei gegenläufige Optimierungsziele dar. Um einen hohen Recall zu erhalten, könnte man alle Wörter eines Textes als Namensnennungen „erkennen“. Jedoch würde dies zu einer extrem schlechten Precision führen, da das Ergebnis von Gleichung 7.1 aufgrund der vielen *False Positives* stark gegen Null tendieren würde. Entsprechend würde der Ansatz, nichts bzw. nur absolut sichere Nennungen zu „erkennen“, zur Maximierung der Precision P führen. Es würden die *False Negatives* in Gleichung 7.1 an Einfluss verlieren und P wäre immer 1.0 bzw. 100%. Dann jedoch würde FN in Gleichung 7.2 bestimmend und der Recall R würde gegen Null gehen.

Da nun unterschiedliche Systeme unterschiedlich stark auf Precision bzw. Recall hin optimiert sind, ist ein direkter Vergleich oftmals schwer. Die beiden Maße werden daher meist zusätzlich in ein kombiniertes Maß, den F-Score, überführt:

$$F_\beta = \frac{(\beta^2 + 1) \cdot P \cdot R}{P + \beta^2 \cdot R} \tag{7.3}$$

Anhand des Faktors β kann das Gewicht von Recall bzw. Precision im kombinierten Maß beeinflusst werden. Für gewöhnlich wird $\beta = 1$ gewählt, wodurch sich dann der F_1-Score zum harmonische Mittel ergibt.

$$F_1 = \frac{2 \cdot P \cdot R}{P + R} \tag{7.4}$$

Das Ergebnis ist ein Wert im Intervall zwischen 0.0 und 1.0 bzw. zwischen 0% und 100%, wobei ein Wert von 1.0 ein in Bezug auf Precision und Recall perfektes System repräsentieren würde.

7.1 Probleme herkömmlicher Ansätze

Die Diskussion existierender und möglicher Verfahren beschränkt sich im Folgenden auf die Erkennung von Personennamen. Die selben oder zumindest ähnliche Verfahren können jedoch im Allgemeinen auch für die Erkennung anderer Typen von Entitäten verwendet werden. Diese Fokussierung dient daher lediglich der Vereinfachung der Diskussion und stellt keine funktionale Einschränkung dar.

Die Technologie der NER ist mittlerweile so weit fortgeschritten, dass aktuelle Verfahren Precision- und Recall-Werte von über 90% erreichen [MP01] [TKSDM03]. Allerdings werden solche Werte derzeit nur unter Laborbedingungen erreicht, d. h. auf den Texten, die zur Entwicklung der jeweiligen NER-Systeme verwendet wurden bzw. auf Texten, die diesen Trainings-/ Entwicklungstexten in Stil und oftmals auch Inhalt ähneln [Law06]. Bspw. wurde und wird auf wissenschaftlichen Veranstaltungen wie den MUC- (Message Understanding Conference) und CoNLL- (Conference on Computational Natural Language Learning) Konferenzen, eine Sammlung von Nachrichtentexten sowohl als Entwicklungs- als auch als Evaluationskorpus [Chi01] verwendet. Dazu wird die Textsammlung in zwei Untermengen aufgeteilt, so dass die Entwicklung bzw. das Training von NER-Systemen auf der einen Teilmenge vorgenommen wird. Die Evaluation dieser Systeme erfolgt dann auf der anderen Teilmenge. Motivation für dieses Vorgehen ist dabei, die Systeme anhand der ersten Teilmenge auf die Eigenarten und Besonderheiten zu trainieren bzw. darauf zuzuschneiden. Mit dem Testlauf auf den übrigen Texten, die nicht für die Entwicklung der Systeme genutzt wurden und daher noch als unbekannt gelten, kann nun ermittelt werden, ob das jeweilige System für alle Texte der Sammlung ähnlich gute Ergebnisse erzielt.

Problematisch an diesem Ansatz bzw. am Streben nach immer höheren Precision- und Recall-Werten ist jedoch, dass die publizierten Systeme hochgradig auf die jeweiligen Texte spezialisiert sind. Diese Spezialisierung äußert sich darin, dass sowohl die Textsorte und das Genre Einfluss auf die Erkennungsleistung haben und oftmals sogar auch das Thema der Texte. In [Gri03] wird ein Beispiel angeführt, in dem es um NER in Nachrichtentexten geht. Die Erkennungsleistung des Systems ging dabei um 3% zurück, als Nachrichten zu einem bisher unbekannten, d. h. nicht trainierten Thema analysiert werden sollten. Bei einem anderen Beispiel aus der Bioinformatik ist der Rückgang noch deutlicher: In einer Textsammlung sollten Gennamen erkannt werden. In [LH05] wurde dabei ein Rückgang von 13% beobachtet, wenn Texte einer bisher unbekannten Sammlung zur Evaluation genutzt wurden, d. h. das System auf einen anderen Korpus angewandt wurde. Bezüglich des Textgenres kann festgestellt werden, dass diese Systeme auf Nachrichtentexten die besten Ergebnisse in absoluten Zahlen erzielen. Insbesondere bei Texten, die nicht professionell verfasst wurden, wie bspw. persönliche E-Mails, E-Mails in Newsgruppen oder Diskussionsforen, liegen die Erkennungsraten mit F-Scores um 68% deutlich niedriger [MWC05]. Eine Erklärung für dieses Phänomen ist die im Vergleich zu journalistischen (Nachrichten-)Texten mangelnde Sorgfalt, mit der die im Beispiel genannten Texte erstellt werden. Es wird weniger auf eine saubere Grammatik geachtet, Rechtschreibung hat zumindest im privaten E-Mail-Verkehr oftmals nicht den Stellenwert wie im Bereich der Print-/Online-Medien oder im geschäftlichen Umfeld und nicht zuletzt werden Bezüge auf für das Textverständnis relevante zusätzliche Informationen oftmals nicht explizit niedergeschrieben, da sie den Kommunizierenden aus dem Kontext bekannt sind [MTU+01][MWC05][Law06].

7.2 Bessere Erkennungsraten durch Kombination mehrerer Systeme

Während die Spezialisierung der oben angeführten Systeme auf spezielle Texte in vielen Bereichen nicht hinderlich oder sogar gewollt ist, so stellt sie doch bei der Anwendung solcher Systeme auf das allgemeine WWW den zentralen Schwachpunkt dar. Das WWW besteht aus einer Sammlung hochgradig heterogener Dokumente, die Genre-übergreifend miteinander verknüpft sind. Nachrichtentexte verweisen auf Enzyklopädie-Einträge, diese wiederum auf Newsgroup-Beiträge, auf Webforen oder private Homepages etc. Das oben geschilderte Sonderszenario, in dem ein NER-System Texte verarbeitet die nicht zu dem Genre passen auf dessen Texten es trainiert wurde, stellt somit im Kontext des WWW den Normalfall dar, mit den entsprechenden Konsequenzen für die Erkennungsleistung.

Ein Ansatz zur Lösung des Problems wäre das Trainieren der Systeme auf einem thematisch breiteren Korpus, der sich aus Texten unterschiedlicher Medien zusammensetzt. Dies würde jedoch zum einen die manuelle Annotation großer Textmengen erfordern. Zum anderen kann aber der Korpus aus praktischen Gründen nicht so umfassend sein, als dass man tatsächlich ein von der Domäne unabhängiges System erhalten würde.

Statt dessen wurden im Rahmen dieser Arbeit andere Wege gesucht, wie ausgehend von bestehenden Systemen und Sprachmodellen die Erkennungsleistung verbessert werden kann. Zwei Ansätze wurden dazu betrachtet [Law06][KSJ07]:

- Die Kombination mehrerer existierender Systeme.
- Die Nutzung von Informationen über bereits erkannte Namen.

7.2.1 Kombination mehrerer existierender Systeme

Für die im Folgenden beschriebenen Analysen wurden drei frei verfügbare NER-Systeme eingesetzt: *OpenNLP* [Ope] als Vertreter der Maximum-Entropy-

Modelle, *LingPipe* [Lina] mit einem Hidden-Markov-Modell sowie der *Stanford Named Entity Recognizer* [FGM05] auf Basis von Conditional Random Fields. Für eine grundlegende Erläuterung dieser Modelle und Techniken sei an dieser Stelle auf die Fachliteratur verwiesen: [Bor99] [BSW99] [FGM05] [MS99].

Die Idee, mehrere Systeme zu kombinieren, wird motiviert durch die Beobachtung, dass unterschiedliche Systeme ihre Fehler an unterschiedlichen Stellen im Text machen [FIJZ03]. Kombiniert man jedoch die Ergebnisse dieser unterschiedlichen Systeme, so können diese Fehler, zumindest teilweise, verhindert werden. Die Kombination kann mittels verschiedener Strategien erfolgen: eine UND-Verknüpfung der Ergebnisse würde genau diejenigen Ergebnisse liefern, die alle beteiligten Systeme übereinstimmend erzielt haben. In erster Linie würde das die Precision erhöhen, wohingegen der Recall als obere Schranke den Recall-Wert des „schlechtesten“ Systems bekäme. Alternativ zur UND-Verknüpfung kann eine ODER-Verknüpfung angewandt werden. Damit würde der Recall optimiert, für die Precision jedoch müsste mit Einbußen gerechnet werden, da auch mehr falsch-positive Ergebnisse geliefert würden. Um diese Fokussierung der UND-/ODER-Verknüpfung auf Precision bzw. Recall aufzuheben, wird im Folgenden eine Voting-Strategie [HZD01] diskutiert. Dabei wird eine Textstelle nur dann in die Ergebnismenge übernommen, wenn eine Mehrheit der Systeme sie als Entität erkannt hat. Sind am Voting-Verfahren bspw. drei Systeme beteiligt und wird als Quorum die einfache Mehrheit angesetzt, so kann jeder Fehler (falsch-positiv sowie falsch-negativ) behoben bzw. vermieden werden, der nur in einem der Systeme auftritt. Soll dennoch in bestimmten Szenarien entweder die Precision oder der Recall im Vordergrund stehen, so kann das Voting-Verfahren auch die UND- bzw. ODER-Verknüfung simulieren, indem das Quorum hin zu „alle Systeme“ bzw. „lediglich ein System“ modifiziert wird. Auch kann der Einfluss eines Systems auf das Gesamtergebnis angepasst werden, indem beim Voting gewichtete Stimmen verwendet werden.

7.2.2 Nutzung von Informationen über bereits erkannte Namen

Alle hier untersuchten NER-Systeme behandeln jedes Vorkommen eines Namens isoliert. Sofern lediglich erkannt werden soll, welche Entitäten in einem Text genannt werden, ist dies auch oftmals ausreichend. Sollen jedoch weitere Analysen durchgeführt werden, wie bspw. im dieser Arbeit zugrunde liegenden Szenario der Bestimmung der Expertise einer Person, so ist auch von Interesse, an welchen Stellen bzw. wie oft eine Person genannt wird. Um diese Information korrekt bestimmen zu können, muss jedoch berücksichtigt werden, dass die Nennung einer Person auf verschiedene Arten erfolgen kann. In den Sätzen „Gottlieb Daimler arbeitete in Stuttgart. Dort entwickelte *er* den Einzylinder-Viertaktmotor“ beziehen sich die Nennungen „Gottlieb Daimler“ und „er“ auf die selbe Person, sie werden als *koreferent* bezeichnet. Koreferenzen beschränken sich aber nicht nur auf die Behandlung solcher Pronomen [Mit01], sondern schließen auch unterschiedliche Bezeichner einer Entität ein. Bspw. wird für Personen in den meisten Texten bei ihrer ersten Nennung der volle Namen verwendet, wohingegen sich folgende Nennungen oftmals lediglich auf den Vor- oder Nachnamen beschränken [VC01]: „Gottlieb Daimler arbeitete in Stuttgart. [...] Daimler entwickelte gemeinsam mit Maybach das erste benzinbetriebene Motorrad“. In vielen Fällen wird die Nennung im zweiten Satz von NER-Systemen nicht eindeutig erkannt, da es sich um keinen vollständigen Namen handelt. In solchen nicht-eindeutigen Kontexten ist daher das Wissen darüber, dass „Daimler“ bereits als Teil eines Namens identifiziert wurde, eine hilfreiche Information zur Erkennung dieser Nennung [KM06].

Implementiert wurde ein solches System anhand von Äquivalenzklassen. Dabei bildet jede erstmalig identifizierte Nennung den ersten Eintrag einer eigenen Klasse. Diese wird dann mit Namen erweitert, die sich von dem ersten Eintrag ableiten lassen, was durch ein einfaches Regelwerk erreicht werden kann. Bspw. ergibt sich aus der Nennung „Dr. Michael Maier“ die Äquivalenzklasse (Dr. Michael Maier; Dr. M. Maier; Dr. Maier; Michael Maier; M. Maier; Michael; Maier). Mittels einfacher Verfahren, bspw. anhand von Regulären Ausdrücken, können dann weitere Nennungen der Person im Text erkannt werden.

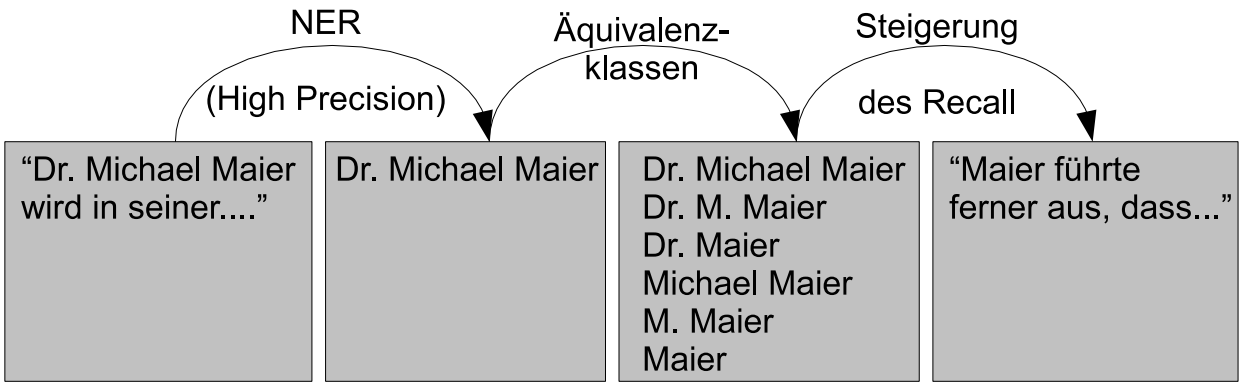

Abbildung 7.1: Verwendung von Äquivalenzklassen (nach [Law06])

Der Einsatz solcher Äquivalenzklassen zur Erhöhung der Ergebnisqualität kann somit als dreistufiger Prozess dargestellt werden (Abbildung 7.1):

1. Annotation des Textes mit einem NER-System bzw. einem kombinierten System, wobei der Schwerpunkt auf hoher Precision liegt. Der Recall ist dabei verhältnismäßig niedrig, da potenziell viele Nennungen übersehen werden.
2. Erzeugung von Äquivalenzklassen aus in Schritt 1 erkannten Nennungen.
3. Suche nach weiteren Nennungen aus den Äquivalenzklassen, um den Recall zu erhöhen.

Die Precision kann mit diesem Ansatz nicht erhöht werden, da falsche Klassifizierungen der NER-Systeme aus Schritt 1 nicht rückgängig gemacht werden. Im Gegenteil, die Precision der Systeme aus Schritt 1 stellt eine obere Schranke dar, die jedoch im Regelfall nicht erreicht werden kann, da in Schritt 3 ggf. weitere falsch-positive Nennungen in die Ergebnismenge einfließen.

7.3 Evaluation

Wie oben bereits angeführt, werden NER-Systeme gewöhnlich entweder auf Korpora getestet, die sich aus Nachrichtentexten zusammen setzen, oder aus solchen, die der Bio-Informatik entstammen. Aus diesen Bereichen kommen auch

die Testdaten. Diese Korpora sind jedoch sehr homogen in Bezug auf Genre und oftmals sogar in Bezug auf das von den einzelnen Texten behandelte Thema. Da in vielen Szenarien – wie bspw. bei der Expertensuche – aber eine Beschränkung auf Nachrichtentexte nicht sinnvoll ist, musste das NER-System mit der Heterogenität der Texte des gesamten WWW umgehen können. Der Nachweis, dass dies gelingt, kann aber anhand eines homogenen Nachrichtenkorpus nicht geliefert werden.

Aus diesem Grund wurde für die folgende Evaluation ein auf der Online-Enzyklopädie *Wikipedia* basierender Korpus gewählt [Law06]. Zwar ist auch hier keine Unabhängigkeit vom Textgenre gegeben – es handelt sich bei allen Texten um Lexikoneinträge – dennoch besitzt dieser Korpus im hier betrachteten Kontext zwei wesentliche Vorteile gegenüber Standardkorpora aus der NER-Evaluationsdomäne:

1. Er setzt sich aus Texten zu unterschiedlichsten Themen zusammen. Dies wurde durch eine zufällige Auswahl von Artikeln aus der gesamten Wikipedia erreicht.
2. Art und Stil der Texte sind deutlich heterogener als in einem Nachrichtenkorpus. Dies begründet sich in der Tatsache, dass die Autoren der Texte meist keine journalistische Ausbildung haben, ihre Schreibstile daher nicht einer Norm folgen. Zudem weisen die Texte einen sehr unterschiedlichen Detaillierungsgrad auf.

Die im Rahmen der Evaluierung eingesetzten Systeme wurden mit ihren standardmäßig mitgelieferten Modellen auf diesen Korpus angewandt und insbesondere nicht neu trainiert. Aufgrund der breiten inhaltlichen Fächerung des hier erzeugten Evaluationskorpus und der Tatsache, dass die Standardmodelle durch Training auf Nachrichtenkorpora generiert wurden, kann somit also dieses Szenario als Einsatz auf trainingsfremden Texten angesehen werden.

Tabelle 7.1 fasst die Ergebnisse dieser Evaluation zusammen. Der erste Eintrag „Listenbasiert“ stellt die Ergebnisse des in Kapitel 4.2.3.1 vorgestellten einfachen Ansatzes auf Basis von Namensdatenbanken und Regulären Ausdrücken

Tabelle 7.1: Evaluationsergebnisse für verschiedene NER-Systeme

NER-System	Precision	Recall	F_1
Listenbasiert	0,65	0,20	0,31
OpenNLP	0,50	0,37	0,43
LingPipe	0,39	0,55	0,45
Stanford	0,64	0,68	0,66
Voting	**0,74**	**0,55**	**0,63**
Voting+Koreferenz	**0,69**	**0,66**	**0,68**

dar. Während die Precision mit 0,65 mit den übrigen Systemen durchaus auf ähnlichem Niveau liegt, so ist doch auffällig, dass das System in Bezug auf Recall (0,20) nicht konkurrenzfähig ist. In erster Linie liegt das daran, dass die Regulären Ausdrücke so definiert wurden, dass die Nennung alleine eines Vor- oder Nachnamens nicht erkannt wird, um die Anzahl der falsch-positiven Ergebnisse im Falle von Namen wie „Müller“ (Nachname und Berufsbezeichnung), „König“ (Nachname und Amt) oder „April“ (Monat und engl. Vorname) zu reduzieren. In dieser Beziehung sind statistische NER-Systeme dem Listenbasierten deutlich überlegen, da sie ein Sprachmodell besitzen, anhand dessen in vielen Fällen erkannt werden kann, ob es sich um eine Namensnennung oder die Nennung des jeweiligen Begriffs in einer anderen Bedeutung handelt. Aufgrund des sehr niedrigen Recalls und des daraus resultierenden niedrigen F_1-Wertes (0,31), wurde dieses NER-System von den weiteren Untersuchungen ausgenommen.

Im Rahmen dieser Arbeit wurde für den Parameter β für die Berechnung des F-Score stets der Wert 1.0 gewählt, also Precision und Recall gleichermaßen stark gewichtet. Dies entspricht dem Standardvorgehen in der Literatur. Für manche Einsatzzwecke kann βdavon abweichende Werte annehmen. Sollen bspw. die Namensnennungen in einem Textkorpus anonymisiert werden, so ist ein hoher Recall wichtiger als eine hohe Präzision – besser es werden alle Namensnennungen gefunden und einige weitere Textpassagen fälschlicherweise unkenntlich gemacht, als dass doch ein Name sichtbar bleibt. In [MWC05] bspw. wurden für diesen Anwendungsfall auch F_2-Werte mit $\beta = 2$ untersucht.

Die Zeilen zwei bis vier in Tabelle 7.1 zeigen die Ergebnisse für die o. g. frei verfügbaren NER-Systeme OpenNLP, LingPipe und den Stanford-NER. OpenNLP und LingPipe besitzen ihre Stärken jeweils im Bereich der verhältnismäßig hohen Precision (OpenNLP) bzw. des Recalls (LingPipe). Ihre F_1-Werte unterscheiden sich hingegen kaum. Demgegenüber zeigt das Standford-System deutlich konsistentere Leistungen in Bezug auf Precision und Recall, was sich auch in einem F_1-Wert auf gleichem Niveau spiegelt. Da das listenbasierte System von den weiteren Untersuchungen ausgenommen wurde, stellen diese drei Systeme die Nulllinie dar, gegenüber der die hier vorgestellten Verbesserungsvorschläge getestet wurden.

Der erste Verbesserungsvorschlag war die Einführung von Mehrheitsentscheidungen, um die Fehler (falsch-positiv sowie falsch-negativ) zu kompensieren, die von lediglich einem System begangen werden. Dazu wurden die drei zuvor genannten Systeme in ein kombiniertes System zusammengefasst und das Quorum für den Entscheid auf die einfache Mehrheit festgesetzt. Die Ergebnisse finden sich in Zeile fünf unter „Voting“. Auffallend ist der herausragende Precision-Wert von 0,74, dem jedoch ein lediglich durchschnittlicher Recall-Wert von 0,55 gegenüber steht.

Der gute Precision-Wert lässt sich dadurch erklären, dass die drei Systeme auf unterschiedlichen Modellen basieren (Maximum-Entropy, Hidden-Markov und Conditional Random Fields) und ihre Fehler daher in der Regel an unterschiedlichen Stellen machen. Der Mehrheitsentscheid führt nun dazu, dass diejenigen Textstellen, die mehrheitlich als Namensnennung erkannt wurden, nun auch mit einer sehr geringen Wahrscheinlichkeit falsch-positiv sind, weil im Gegenzug die Wahrscheinlichkeit gering ist, dass drei unabhängige Systeme an der selben Textstelle einen Fehler machen.

Dies hat allerdings offensichtliche Auswirkungen auf den Recall. Textstellen, die von lediglich einem System erkannt werden, die also im Sinne des kombinierten Modells eine große Unsicherheit aufweisen, fallen bei diesem Ansatz heraus. Während dadurch falsch-positive Nennungen reduziert werden und die Precision steigt (siehe oben), so sinkt demgegenüber klar der Recall, da

Tabelle 7.2: Beste NER-Systeme für Precision bzw. Recall bzw. F_1

NER-System	Precision	Recall	F_1
Stanford + OpenNLP	**0,84**	0,35	0,48
Voting + Koreferenz	0,69	**0,66**	**0,68**

manche Konstrukte lediglich von einem System erkannt werden und somit den Mehrheitsentscheid nicht passieren. Zusammengefasst bedeutet dies, dass bei Anwendung dieses Ansatzes davon ausgegangen werden kann, dass diejenigen Nennungen, die identifiziert wurden, auch tatsächlich Namensnennungen sind. Allerdings werden bei weitem nicht alle Nennungen auch identifiziert. Liegt der Fokus also auf einer hohen Precision, so ist das Voting-System den drei Einzelsystemen eindeutig vorzuziehen, und auch in Bezug auf den F_1-Wert ist es mit 0,63 gegenüber 0,66 nur wenig schlechter als das beste Einzelsystem, der Stanford-NER.

Die letzte Zeile zeigt die Ergebnisse wenn zusätzlich zum Voting-System auch Informationen über Koreferenzen genutzt werden. Zwar sinkt die Precision, da auch zu falsch-positive Nennungen Äquivalenzklassen gebildet werden und in der Folge weitere Fehler auftreten. Dennoch kann das Ziel, den Recall zu steigern bei verhältnismäßig geringen Einbußen der Precision, als erfüllt angesehen werden. Einer Reduzierung der Precision von 0,74 auf 0,69 steht eine Erhöhung des Recalls von 0,55 auf 0,66 gegenüber. Diese positive Bilanz zeigt sich auch im signifikant höheren F_1-Wert (0,68 gegenüber 0,63).

Der Vollständigkeit halber zeigt Tabelle 7.2 auch noch die Systemkombinationen, die bei alleiniger Betrachtung von Precision/Recall/F_1 die besten Ergebnisse erzielen. Zeile eins zeigt den Ansatz mit den besten Werten für Precision, den kombinierten Voting-Ansatz. Allerdings sind in diesem Fall lediglich die beiden Systeme Stanford-NER und OpenNLP beteiligt, was faktisch eine UND-Verknüpfung bedeutet. Dass diese Kombination einen hohen Wert für die Precision liefert, war aus den Ergebnissen in Tabelle 7.1 vorhersehbar. Auch dort hatte das kombinierte System die besten Precision-Werte erzielt, die jedoch durch das Recall-optimierte LingPipe signifikant reduziert wurden. Bedingt durch die

UND-Verknüpfung konnte jedoch bei der hier betrachteten Zweierkombination mit keinem hohen Recall gerechnet werden, da dieser durch den Recall des schlechtesten Systems beschränkt ist (OpenNLP mit R=0,37).

Zeile zwei zeigt die Ergebnisse des kombinierten Voting-Ansatzes (OpenNLP + LingPipe + Stanford-NER), erweitert um die Koreferenz-Technik. Diese Kombination liefert sowohl die besten Werte für den Recall, als auch insgesamt den besten F_1-Wert. Diese Werte finden sich auch in Tabelle 7.1. Die Hintergründe wurden dort bereits erläutert.

7.4 Zusammenfassung

Der in diesem Abschnitt vorgestellte Ansatz zur Identifizierung von Namensnennungen in natürlichsprachigen Texten hat das Ziel, die Schwächen einzelner NER-Systeme auf trainingsfremden Texten zu kompensieren. Dies wurde durch die Kombination mehrerer einzelner NER-Systeme sowie durch die Nutzung von Koreferenz-Informationen erreicht. Während der Kombinationsansatz in erster Linie der Steigerung der Precision dient, so dient der Koreferenz-Ansatz der Steigerung des Recalls. Auch wenn keiner der hier erarbeiteten Ansätze zu Erkennungsraten führt, wie sie für die Einzelsysteme publiziert sind (Precision und Recall teilweise bis zu 0,90), so sind die Ergebnisse dennoch als Erfolg zu werten, da die Untersuchungen auf Textdaten durchgeführt wurden, die nicht den Trainingsdaten der Einzelsysteme entsprechen. An den signifikant besseren Ergebnissen im Vergleich zu den Einzelsystemen wird dieser Erfolg deutlich.

Im praktischen Einsatz wird man allerdings nicht alleine auf dieses Konzept setzen. Viele Dokumente lassen sich eindeutig bestimmten Kategorien zuordnen, wobei das Wissen über diese Kategorien zur Namenserkennung genutzt werden kann. Bspw. können wissenschaftliche Veröffentlichung relativ leicht erkannt werden und Autorennamen sowie Literaturreferenzen extrahiert werden, da diese meist sehr einfachen Regeln folgen [LGB99] [PM04]. Ein weiteres Beispiel sind Patentdatenbanken im Web. Ist hier die Struktur sowohl der gesamten Webpräsenz als auch der jeweiligen Dokumenttypen bekannt, so kann dieses

Wissen zur Identifikation von Namensnennungen genutzt werden, da derartige datenbankbasierte Anwendungen stets Dokumente erstellen, die dem selben Schema folgen und daher die Nennungen der Erfinder bzw. Autoren auch immer an der selben Stelle im Dokument erfolgt. Lässt der praktische Einsatz eines solchen Systems also erwarten, dass bestimmte Quellen – zwar nicht ausschließlich, aber doch häufig – benutzt werden, so kann eine Anpassung auf deren Textstruktur sinnvoll sein. Kapitel 5 beschäftigt sich mit dem Konzept der Webpräsenz-Konnektoren und einem dazu passenden Integrationsframework, das eine solche Anpassung effizient ermöglicht.

Ein weiterer Ansatz zur Verbesserung der Erkennungsraten besteht im dokumentenübergreifenden Einsatz der Koreferenz-Bestimmung. Bspw. können zusätzlich zur dokumenteninternen Berechnung auch diejenigen Dokumente mit berücksichtigt werden, die über direkte Verweise mit dem untersuchten Dokument verknüpft sind. Auf diese Art und Weise lassen sich zusätzlich strukturelle Informationen des WWW für die Erkennung nutzen.

KAPITEL 8

ZUSAMMENFASSUNG UND AUSBLICK

Dieses Kapitel schließt die vorliegende Arbeit ab und fasst die Ergebnisse zusammen. Dazu wird zunächst das Gesamtkonzept anhand des in Kapitel 1.1 vorgestellten beispielhaften Szenarios diskutiert. Anschließend werden die Ergebnisse den Anforderungen gegenüber gestellt und das Erreichte sowie offene Punkte bewertet.

Die im Rahmen dieser Arbeit entwickelten Konzepte und Softwarelösungen formen ein Werkzeug, das zur Unterstützung und Teilautomatisierung von Webrecherchen gedacht ist. Da sich reale Recherchen, insbesondere im Umfeld komplexer Problemstellungen, selten gleichen und immer individuellen Charakter haben, sind auch die einzusetzenden oder einsetzbaren Mittel und Werkzeuge nicht zwangsläufig die selben und ein statisches Werkzeug alleine würde dem Anspruch der Universalität nicht gerecht werden.

In dieser Arbeit wurden daher zwei Schwerpunkte gesetzt. Zum einen die Entwicklung einer Suchplattform (vgl. Kapitel 4.1), die sowohl grundlegende

Suchwerkzeuge bereits implementiert, als auch entsprechende Schnittstellen für Erweiterungen bzw. die Integration externer Komponenten vorsieht. Darüber hinaus lag der zweite Schwerpunkt auf der Optimierung von drei dieser Plattformkomponenten (vgl. Kapitel 5-7).

Die Suchplattform an sich wurde in Kapitel 4.2 anhand einer prototypischen Implementierung der Expertensuche evaluiert. Auch eine Bewertung der drei Detail-Ansätze wurde in den Kapiteln 5-7 bereits durchgeführt. Letztere Ansätze bauen auf der Suchplattform auf und liefern somit ebenfalls einen Beitrag zu deren Evaluation.

An dieser Stelle soll daher nun eine zusammenfassende Betrachtung erfolgen, die das Gesamtkonzept vor dem Hintergrund eines realen Einsatzszenarios beleuchtet und dessen Tauglichkeit für reale Einsatzzwecke und Fragestellungen bewertet.

8.1 Beispielhafte Anwendung

In Kapitel 1.1 wurde das an eine reale Problemstellung angelehnte Szenario der alternativen Energiequellen in Form von Brennstoffzellen für Outdoor-Stirnlampen eingeführt. Im Folgenden wird nun ein möglicher Verlauf einer solchen Suche skizziert, wobei über den gesamten Suchprozess hinweg die Suchplattform mit ihren diversen Komponenten zum Einsatz kommt.

Die Suche wird, wie bei Suchen mit herkömmlichen Suchmaschinen gewohnt, über eine einfache Schlüsselwortanfrage „fuel cell“ (engl. für Brennstoffzelle) gestartet. Die an die Suchplattform angeschlossenen externen Suchmaschinen liefern jedoch erwartungsgemäß eine unspezifische und thematisch breit gefächerte Trefferliste, was auch durch den implizit aktivierten Clustermechanismus in Form von tendenziell unabhängigen Clustern bestätigt wird (vgl. Abbildung 8.1)

Im nächsten Schritt muss also zwangsläufig eine Konkretisierung der Suchanfrage erfolgen. Dazu bietet sich eine Selektion der relevanten Cluster sowie eine

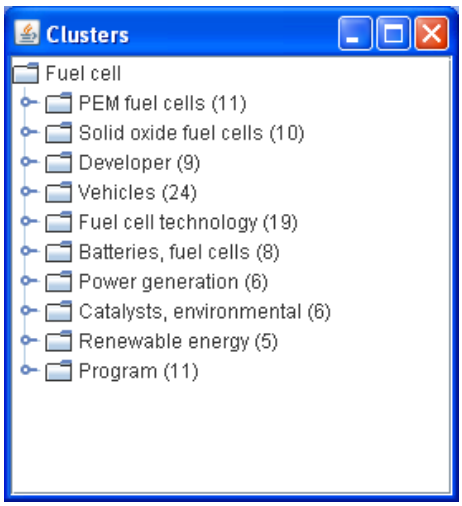

Abbildung 8.1: Cluster der zur Anfrage „Fuel cell“ gefundenen Dokumente

nachgelagerte Schlüsselwortextraktion aus in diesen enthaltenen Dokumenten an. Alternativ kann über die Darstellung des konzeptuellen Kontexts von „fuel cell“ (vgl. Abbildung 8.2) mittels der Begriffsnetz-Komponente, die auf dem in Kapitel 6 vorgestellten Algorithmus zur Berechnung der Ähnlichkeit von Dokumenten basiert, eine weitere Darstellung der verwandten Themen und eine interaktive Fokussierung durch die Selektion relevanter Bereiche erzielt werden.

In diesem Fall soll eine Fokussierung auf physikalische und chemische Grundlagen erfolgen. Die modifizierte Schlüsselwortanfrage ergibt sich zu „fuel cell AND (hydrogen OR catalyst OR . . . OR energy)“.

Über eine oder mehrere Iterationen solcher Anpassungen lässt sich die Suchanfrage schnell in die gewünschte Richtung lenken und beliebig selektiv gestalten. Neben der genannten Begriffsnetz-Komponente sind auf dieser Stufe auch der Synonym-Dienst sowie automatisierte Übersetzungen von Schlüsselwörtern in andere Sprachen, vornehmlich zwischen Deutsch und Englisch, hilfreiche Instrumente.

Die erstellte Suchanfrage kann nun entweder direkt an die angeschlossenen Suchmaschinen übergeben, oder aber zur manuellen Suche von Beispieldokumenten für den Focused Crawler genutzt werden. Letzterer sucht dann in der Folge nach inhaltlich ähnlichen Dokumenten, wobei sich beide Konzepte auch

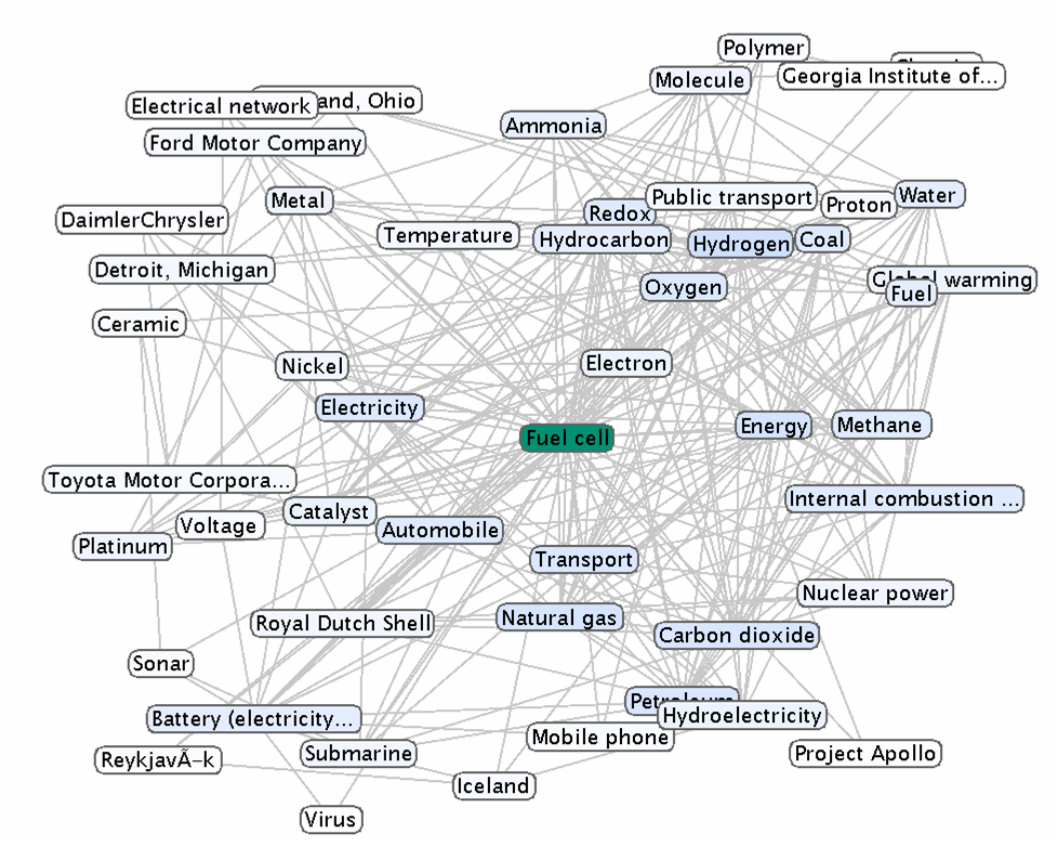

Abbildung 8.2: Konzeptueller Kontext für den Begriff „Fuel cell“

kombinieren lassen. Wird bspw. der Focused Crawler gestartet, so kann anhand des Crawler-Logs und der Linkstruktur-Komponente verfolgt werden, welchen Weg der Crawler durch das WWW nimmt, wo er thematische Inseln identifiziert und wie diese zusammen hängen (vgl. Abbildung 8.3).

Um weiteren Einfluss auf das Ergebnis des Crawlers und dessen Bewertung zu nehmen, kann der Benutzer die Bewertung für seiner Einschätzung nach „gute“ bzw. „schlechte“ Quellen manuell herauf oder herab setzen. In der Folge wird der Crawler dann Verweisen von dort gefundenen Dokumenten mit höherer oder niedrigerer Priorität folgen und die Bewertung entsprechend anpassen. Bspw. könnte so der Quelle *www.fuelcells.org* (allgemeine Informationen und Verweise zum Thema Brennstoffzelle) eine höhere Wertung zugedacht werden, wohingegen *consumerguideauto.howstuffworks.com* (Werbemanagement für die Domain howstuffworks.com) herabgewertet würde (vgl. Abbildung 8.3).

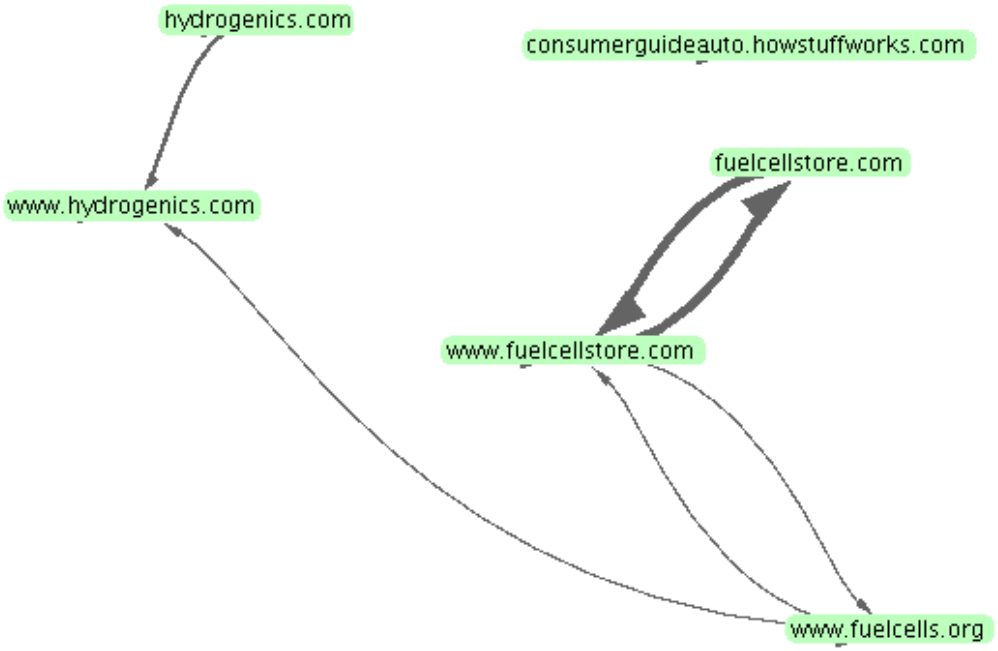

Abbildung 8.3: Besuchte Domains zu Beginn der Suche zum Thema Brennstoffzellen

In Kapitel 4.2 wurden weitere allgemeine Komponenten beschrieben, die bei vielen Suchen Gewinn bringend eingesetzt werden können. An dieser Stelle wird auf eine vollständige Darstellung im beispielhaften Suchprozess verzichtet und statt dessen auf die problemspezifischen Werkzeuge zur Unterstützung der Expertensuche abgezielt:

Bei jedem Download einer Webressource wird im Hintergrund eine Namenserkennung gestartet. Dabei werden Personen- wie Firmennamen im Text identifiziert und das Dokument entsprechend annotiert und persistiert. Eine speziell für die Expertensuche entwickelte Analysekomponente kombiniert nun die Dokumentenlisten der angeschlossenen Suchmaschine(n), die Crawl-Verlaufs- und Rankinginformationen, Netzstrukturinformationen und eben die Ergebnisse der Namenserkennung. Aus diesen Daten werden verschiedene Kennzahlen für die identifizierten Personen berechnet, ihre Verknüpfungen analysiert und damit letztlich eine Aussage über ihre vermutete Expertise auf dem betrachteten Themengebiet gemacht (vgl. Kapitel 4.2.3.2). Dabei kann der Benutzer nach Be-

lieben das Gewicht der einzelnen Kriterien modifizieren und die Auswirkungen auf das Ranking der identifizierten Experten beobachten.

Für eine manuelle Prüfung der Ergebnisliste und als Startpunkt für weitere Suchen, bspw. nach Kontaktdaten, werden zu jedem identifizierten Experten entsprechend die Fundstellen im Web, deren Ranking sowie weitere Hintergrundinformationen, wie Dichte und Kontext des von den Experten und Ressourcen gebildeten Netzwerks, dargestellt (vgl. Abbildung 8.4).

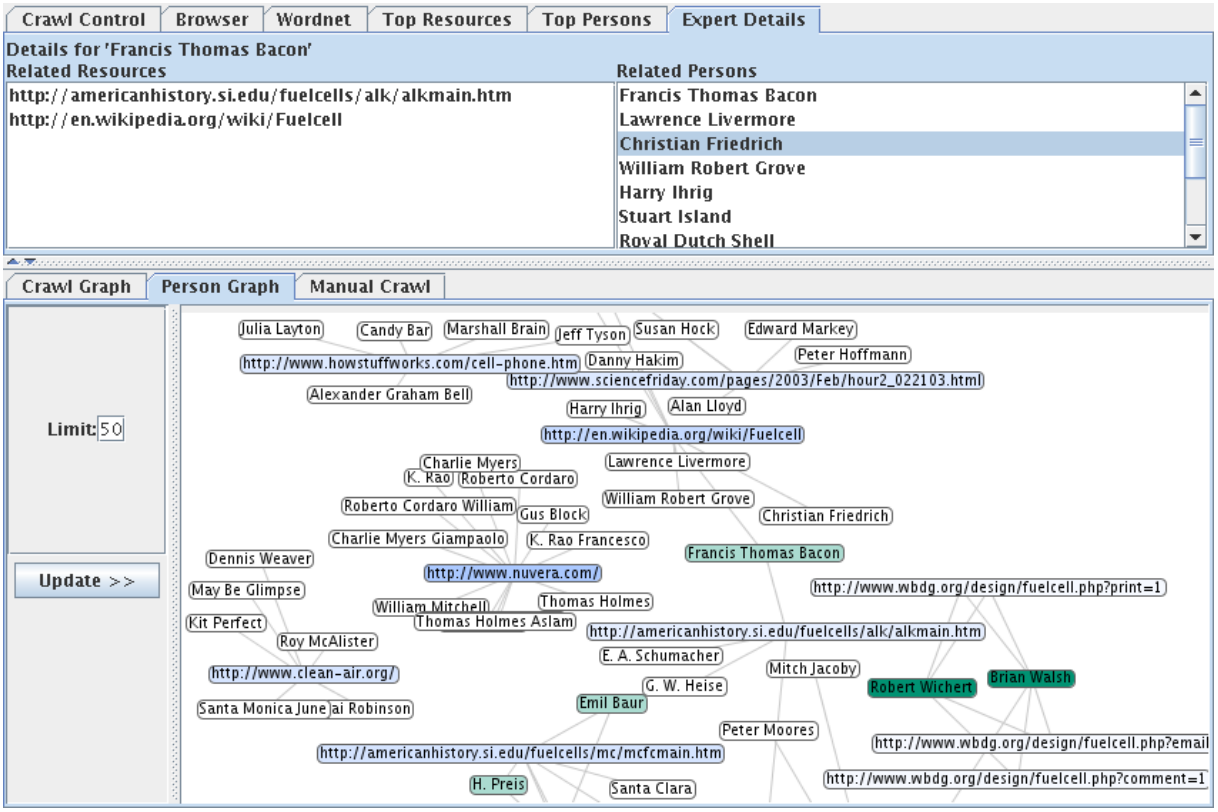

Abbildung 8.4: Potenzielle Experten und deren Kontext

Insbesondere im Zusammenhang mit einem Focused Crawler spielt die Persistierung der gefundenen Information eine große Rolle. So kann eine langlaufende Suche zu jedem Zeitpunkt unterbrochen und beliebig später fortgesetzt werden. Die Suchanfrage(n) lassen sich jederzeit modifizieren, bspw. durch die Erweiterung mit Namen von Personen oder Firmen, die in früheren Suchschritten bereits als relevant identifiziert wurden.

Neben der Möglichkeit des Wiederaufsetzens von unterbrochenen Suchen

schafft diese Persistierung auch die Grundlagen für ein – bisher noch nicht implementiertes – Monitoring. Es können bspw. zusätzliche Komponenten entwickelt werden, die die gleiche Suche in regelmäßigen zeitlichen Abständen wiederholen und entsprechende Unterschiede aufzeigen. Somit können Entwicklungen von Themen oder Personennetzwerken analysiert werden, die allein durch eine einmalige Suche nicht entdeckt würden. Wertvolle Dienste würde in diesem Zusammenhang das in Kapitel 5 beschriebene Konnektoren-Framework leisten, da es die Erschließung von einzelnen Webpräsenzen besonders einfach gestaltet.

8.2 Bewertung der Ergebnisse und Ausblick

Die vorliegende Arbeit motiviert sich in erster Linie aus den Anforderungen und Ergebnissen des Forschungsprojekts *nova-net – Innovation in der Internetökonomie*. Hier sollten in verschiedenen Industriekooperationen Mittel und Wege gefunden werden, wie das Internet und insbesondere das WWW zur Informationsgewinnung für kleine und mittelständische Unternehmen genutzt werden kann – vor dem Hintergrund dass die beteiligten Akteuere keine professionellen Rechercheure sind. Die Durchführung einer Suche in diesem Kontext wurde als komplexes Suchproblem und die folgenden drei Aspekte als die wesentlichen Herausforderungen dabei identifiziert: die Integration des Benutzers in den Suchprozess (vgl. Kapitel 2.1), die Struktur des WWW mit seinen heterogenen, für herkömmliche Webcrawler nur teilweise zugänglichen Ressourcen (vgl. Kapitel 2.2), sowie die Analyse der im WWW vorherrschenden unstrukturierten bzw. schwach strukturierten Ressourcen, hauptsächlich natürlichsprachige Texte (vgl. Kapitel 2.3).

Aufgrund der Komplexität des gesamten Themenfeldes der Websuche, erfolgte nach der Analyse der Problemstellung eine Fokussierung auf vier Themen, um diese Herausforderungen anzugehen: Suchplattform, Webpräsenz-Konnektoren, Bestimmung der inhaltlichen Ähnlichkeit von Texten sowie Personen- bzw. Expertenidentifikation. Die Wahl fiel auf eben diese vier Themen, weil sie es

erlaubten, einen Kompromiss zwischen der Entwicklung von theoretischen Konzepten und praktisch einsetzbaren Werkzeugen zu finden bzw. weil dort die Synergien am größten erschienen.

Mit der in Kapitel 4 vorgestellten Suchplattform wurde die Grundlage für ein effizientes und einzelne Schritte des Suchprozesses integrierendes Werkzeug geschaffen. Das Konzept der Plattform an sich richtet sich dabei weniger an den Endbenutzer selbst, als viel mehr an die Entwickler von Suchwerkzeugen. Der Benutzer profitiert jedoch über den intuitiven Zugriff auf verschiedenste integrierte Werkzeuge. Die Plattform erlaubt es dabei, externe problemspezifische Komponenten in einen Suchprozess zu integrieren und die gewonnenen Daten und Informationen zwischen beteiligten Komponenten und dem Benutzer auf einfache Art und Weise auszutauschen.

Im Zuge der Entwicklung der Suchplattform wurden verschiedene Werkzeugkomponenten entwickelt, die auch bereits im Projekt *nova-net* sowohl im industriellen als auch im universitären Umfeld erfolgreich eingesetzt werden konnten. Besonders hervorzuheben ist in diesem Zusammenhang das Framework für die Entwicklung von Webpräsenz-spezifischen Konnektoren (vgl. Kapitel 5). Durch dieses Framework ist es möglich, schnell und einfach strukturierte Daten aus nicht oder nur schwach strukturierten Webressourcen zu extrahieren, indem Wissen über die jeweiligen Ressourcen in die Extraktions- und Analysekomponente integriert wird. Demonstriert wurde die Praxistauglichkeit unter anderem am Beispiel der Webdatenbank des Deutschen Patent- und Markenamtes und am Webauftritt eines Universitätsinstituts.

Im Laufe des praktischen Einsatzes dieser Suchplattform wurde an verschiedenen Stellen Optimierungspotenzial identifiziert. Insbesondere der Wunsch der Benutzer nach Unterstützung bei der Spezifikation und Konkretisierung von Suchanfragen hat den Einsatz des Konzepts der „Spezifikation anhand von Beispieldokumenten und Begriffsnetzen“ motiviert. In diesem Zusammenhang konnte gegenüber herkömmlichen rein syntaxbasierten Verfahren zur Berechnung der inhaltlichen Dokumentenähnlichkeit ein qualitativer Fortschritt durch den Einsatz der in Kapitel 6 beschriebenen konzeptuellen Kontexte er-

zielt werden. Auch die in Kapitel 7 diskutierten Konzepte zur Identifizierung von Namensnennungen in natürlichsprachigen Texten sind motiviert durch die Schwächen aktueller Systeme. Zwar liefern diese gute bis sehr gute Ergebnisse auf Texten, für die sie entwickelt oder zumindest trainiert wurden. Auf allgemein unbekannten Texten jedoch können diese hohen Erkennungsraten meist nicht erreicht werden. Der Ansatz zur Nutzung von Koreferenzinformationen und der Kombination mehrerer Systeme führt auch hier zu Verbesserungen der Ergebnisqualität.

Dennoch bleiben auch Fragen offen und es existiert weiteres Optimierungspotenzial in allen geschilderten Bereichen. Die Suchplattform bietet zwar eine solide Grundlage für die Eigenentwicklung spezieller Problemlösungsstrategien und -komponenten. Dennoch erhebt sie keinen Anspruch auf völlige Universalität – es wird immer Probleme geben, für dessen Lösung die Plattform nicht geeignet ist oder zumindest nicht die optimale Unterstützung bietet. Auch das Konnektoren-Framework besitzt Einschränkungen: so bauen bspw. Webanwendungen zunehmend auf Technologien wie AJAX (Asynchronous JavaScript and XML), was eine Browser-basierte Webanwendung in der Bedienung mit herkömmlichen interaktiven Desktop-Anwendungen vergleichbar macht – allerdings die Interpretation und automatisierte Verarbeitung durch Analysesysteme signifikant erschweren kann. Hier bietet das Framework bisher keine umfassenden Antworten.

Der Schwachpunkt der konzeptuellen Kontexte liegt nicht in der Qualität der Ergebnisse sondern vielmehr in ihrer aufwändigen Berechnung. Außerhalb des hier beschriebenen Focused-Crawler-Szenarios mit seinem verhältnismäßig geringen Datenaufkommen lassen sich die vorgeschlagenen Algorithmen auf großen Datenmengen kaum effizient einsetzen. Hier sind weitere Konzepte zur Beschleunigung der Berechnung gefragt, bspw. angepasste Indexstrukturen um eine teilweise Vorabberechnung der Kontexte zu ermöglichen, oder aber eine stärkere Parallelisierung, bspw. mittels Cloud Computing.

Die Namenserkennung schließlich bietet auf den Texten des hier beschriebenen Szenarios bessere Erkennungsraten als herkömmliche bzw. einzelne Systeme.

Dennoch kann es an die Leistungen moderner Systeme auf den Texten, für die sie trainiert wurden, nicht anknüpfen.

Zusammenfassend lässt sich festhalten, dass die im Rahmen dieser Arbeit vorgeschlagenen und implementierten Konzepte den Benutzer bei der Bearbeitung komplexer Suchfragestellungen im WWW deutlich besser unterstützen können, als dies mit herkömmlichen Werkzeugen möglich wäre. Illustriert und bestätigt wurde dies anhand des in Kapitel 1.1 vorgestellten beispielhaften Suchprozesses und in verschiedenen Praxiseinsätzen im Rahmen von Industriekooperationen im Forschungsprojekt *nova-net*. Wo die Unterstützung des Benutzers noch nicht ausreichend ist, werden in dieser Arbeit Konzepte vorgeschlagen, die eine individuelle, problemorientierte Entwicklung von Werkzeugen erleichtern und für einen entsprechend geringeren Aufwand bei deren Implementierung sorgen.

Abbildungsverzeichnis

TABELLENVERZEICHNIS

LITERATURVERZEICHNIS

[BBC+11] Alessandro Bozzon, Marco Brambilla, Stefano Ceri, Piero Fraternali, und Salvatore Vadacca. Exploratory search in multi-domain information spaces with liquid query. In *Proceedings of the 20th international conference companion on World wide web*, WWW '11, Seiten 189–192, New York, NY, USA, 2011. ACM.

[BBCF10] Alessandro Bozzon, Marco Brambilla, Stefano Ceri, und Piero Fraternali. Liquid query: multi-domain exploratory search on the web. In Michael Rappa, Paul Jones, Juliana Freire, und Soumen Chakrabarti, Editoren, *WWW*, Seiten 161–170. ACM, 2010.

[BC87] Nicholas J. Belkin und W. Bruce Croft. Retrieval techniques. Seiten 109–145, 1987.

[BdVCS07] Peter Bailey, Arjen P. de Vries, Nick Craswell, und Ian Soboroff. Overview of the TREC 2007 Enterprise Track. In Ellen M. Voorhees und Lori P. Buckland, Editoren, *TREC*, Volume Special Publication 500-274. National Institute of Standards and Technology (NIST), 2007.

[BdVS+10] K. Balog, A. P. de Vries, P. Serdyukov, P. Thomas, und T. Westerveld. Overview of the TREC 2009 Entity Track. In *Proceedings of the Eighteenth Text REtrieval Conference (TREC 2009)*. NIST, February 2010.

[Ber00] Michael K. Bergman. The Deep Web: Surfacing Hidden Value, 2000. Whitepaper.

[Biz] BizWiz. http://bizwiz.com.

[BL89] Tim Berners-Lee. Information Management: A Proposal, 1989.

[Bor99] Andrew Eliot Borthwick. *A maximum entropy approach to named entity recognition*. Dissertation, New York, NY, USA, 1999. Adviser-Grishman, Ralph.

[BP98] Sergey Brin und Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. In *WWW7: Proceedings of the seventh international conference on World Wide Web 7*, Seiten 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[BP03] Satanjeev Banerjee und Ted Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, Seiten 805–810, 2003.

[BST+08] Krisztian Balog, Ian Soboroff, Paul Thomas, Peter Bailey, Nick Craswell, und Arjen P. de Vries. Overview of the TREC 2008 Enterprise Track. In *The Seventeenth Text Retrieval Conference (TREC) Proceedings*, Gaithersburg, USA, 2008.

[BSW99] Daniel M. Bikel, Richard M. Schwartz, und Ralph M. Weischedel. An Algorithm that Learns What's in a Name. *Machine Learning*, 34(1-3):211–231, 1999.

[BYRN99] Ricardo A. Baeza-Yates und Berthier A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[CdVS05] Nick Craswell, Arjen P. de Vries, und Ian Soboroff. Overview of the TREC 2005 Enterprise Track. In Ellen M. Voorhees und Lori P. Buckland, Editoren, *TREC*, Volume Special Publication 500-266. National Institute of Standards and Technology (NIST), 2005.

[Chi01] Nancy A. Chinchor. Overview of MUC-7. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, 2001.

[Chr08] Michael G. Christel. Supporting video library exploratory search: when storyboards are not enough. In *Proceedings of the 2008 international conference on Content-based image and video retrieval*, CIVR '08, Seiten 447–456, New York, NY, USA, 2008. ACM.

[CM08] Robert G. Capra und Gary Marchionini. The relation browser tool for faceted exploratory search. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '08, Seiten 420–420, New York, NY, USA, 2008. ACM.

[CNCO10] Karen Church, Joachim Neumann, Mauro Cherubini, und Nuria Oliver. SocialSearchBrowser: a novel mobile search and information discovery tool. In *Proceeding of the 14th international conference on Intelligent user interfaces*, IUI '10, Seiten 101–110, New York, NY, USA, 2010. ACM.

[Cre97] Fabio Crestani. Application of Spreading Activation Techniques in Information Retrieval. *Artificial Intelligence Review*, Seiten 453–482, December 1997.

[CvdBD99] Soumen Chakrabarti, Martin van den Berg, und Byron Dom. Focused crawling: a new approach to topic-specific Web resource discovery. *Comput. Networks*, 31(11-16):1623–1640, 1999.

[DCL+00] Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, Lee C. Giles, und Marco Gori. Focused Crawling using Context Graphs. In *26th International Conference on Very Large Databases, VLDB 2000*, Seiten 527–534, Cairo, Egypt, 2000.

[DDL+90] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, und Richard A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[DEP] DEPATIS. http://depatisnet.dpma.de.

[DFJ+04] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, R. Scott Cost, Yun Peng, Pavan Reddivari, Vishal C Doshi, und Joel Sachs. Swoogle: A Search and Metadata Engine for the Semantic Web. In *Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management*. ACM Press, November 2004.

[DH99] Jeffrey Dean und Monika R. Henzinger. Finding related pages in the World Wide Web. In *WWW '99: Proceedings of the eighth international conference on World Wide Web*, Seiten 1467–1479, New York, NY, USA, 1999. Elsevier North-Holland, Inc.

[Expa] Experteer. http://experteer.de.

[Expb] Expert System, COGITO. http://www.expertsystem.net/.

[Fel98] Christiane Fellbaum, Editor. *WordNet. An Electronic Lexical Database*. MIT Press, Cambridge, MA, USA, 1998.

[Fer03] Reginald Ferber. *Information Retrieval: Suchmodelle und Data-Mining-Verfahren für Textsammlungen und das Web*. dpunkt Verlag, Heidelberg, 2003.

[FGM05] Jenny Rose Finkel, Trond Grenager, und Christopher Manning. Incorporating non-local information into information extraction systems by Gibbs sampling. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, Seiten 363–370, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[FIJZ03] Radu Florian, Abe Ittycheriah, Hongyan Jing, und Tong Zhang. Named entity recognition through classifier combination. In *Proceedings of CoNLL-2003*, Seiten 168–171, 2003.

[FKH09] Jan Finzen, Thomas Krepp, und Daniel Heubach. Web Searching in Early Innovation Phases: a Survey among German Companies. In *2nd ISPIM Innovation Symposium*, New York, 12/2009 2009.

[FKK10] Jan Finzen, Harriet Kasper, und Maximilian Kintz. *Innovation Mining - Ein Leitfaden für die effektive Recherche unternehmensstrategisch relevanter Informationen im Internet*. Fraunhofer IRB, Stuttgart, 2010.

[Fow99] Martin Fowler. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley, 1999.

[Gei08] Benjamin Geißelmeier. Entwicklung eines Konnektorenframeworks für das Webcrawling. Diplomarbeit, Universität Stuttgart, Fakultät Informatik, Elektrotechnik und Informationstechnik, Germany, Dezember 2008.

[GF04] David A. Grossman und Ophir Frieder. *Information Retrieval. Algorithms and Heuristics*, Volume 15 of *The Information Retrieval Series*. Springer, 2nd edition, 2004.

[GHJV95] Erich Gamma, Richard Helm, Ralph Johnson, und John Vlissides. *Design Patterns*. Addison-Wesley Professional, January 1995.

[GKC+10] Carsten Görg, Jaeyeon Kihm, Jaegul Choo, Zhicheng Liu, Sivasailam Muthiah, Haesun Park, und John Stasko. Combining Computational Analyses and Interactive Visualization to Enhance Information Retrieval. In *HCIR 2010: Proceedings of the Fourth Workshop on Human-Computer Interaction and Information Retrieval*, 2010.

[GM07] Evgeniy Gabrilovich und Shaul Markovitch. Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of The Twentieth International Joint Conference for Artificial Intelligence*, Seiten 1606–1611, 2007.

[Gri03] Ralf Grishman. Information extraction. In R. Mitkov, Editor, *Oxford Handbook of Computational Linguistics*, Kapitel 30. Oxford University Press, Oxford, 2003.

[Gri08] Seth Grimes. Text Technologies in the Mainstream - Text Analytics Solutions, Applications, and Trends. A white paper prepared for Text Analytics Summit 2008. 2008.

[GVK08] Nova-net - Innovation in der Internetökonomie; Abschlussbericht, 2008.

[Ham50] Richard Hamming. Error-detecting and error-correcting codes. In *Bell System Technical Journal*, Volume 29(2), Seiten 147–160, 1950.

[HG04] Erik Hatcher und Otis Gospodnetic. *Lucene in Action (In Action series)*. Manning Publications Co., Greenwich, CT, USA, 2004.

[HS02] F. Heeren und W. Sihn. Xpertfinder - message analysis for the recommendation of contact persons within defined topics. *Africon Conference in Africa, 2002. IEEE AFRICON. 6th*, 1:41–46, Oct. 2002.

[HTM] W3C HTML Working Group. http://www.w3.org/html/wg/.

[HZD01] Hans van Halteren, Jakub Zavrel, und Walter Daelemans. Improving accuracy in word class tagging through the combination of machine learning systems. *Computational Linguistics*, 27:199–230, 2001.

[IBM] IBM Content Analyzer. http://www-01.ibm.com/software/data/content-management/content-analyzer/.

[IPV] Institut für Parallele und Verteilte Systeme (IPVS). http://www.ipvs.uni-stuttgart.de/abteilungen/as/start.

[JKS05] Mihály Jakob, Fabian Kaiser, und Holger Schwarz. SEMAFOR: a framework for an extensible scenario management system. *Engineering Management Conference, 2005. Proceedings. 2005 IEEE International*, 1:354–358, Sept. 11-13, 2005.

[JKS06] Mihály Jakob, Fabian Kaiser, und Holger Schwarz. *Technologie-Roadmap*. Fraunhofer IRB Verlag, April 2006. ISBN: 3-8167-7047-9.

[JKS+07] Mihály Jakob, Dierk-Oliver Kiehne, Holger Schwarz, Fabian Kaiser, und Severin Beucker, Editoren. *Delphigestütztes Szenario-Management und -Monitoring*. Fraunhofer IRB Verlag, September 2007.

[JS03] Mario Jarmasz und Stan Szpakowicz. Roget's thesaurus and semantic similarity. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP-03)*, Seiten 212–219, Borovets, Bulgaria, 2003.

[JS06] Bernard J. Jansen und Amanda Spink. How are we searching the World Wide Web? A comparison of nine search engine transaction logs. *Information Processing & Management*, 42(1):248–263, 2006. Formal Methods for Information Retrieval.

[JSKM06] Mihály Jakob, Holger Schwarz, Fabian Kaiser, und Bernhard Mitschang. Modeling and generating application logic for data-intensive web applications. In *Proceedings of the 6th international conference on Web engineering*, ICWE '06, Seiten 77–84, New York, NY, USA, 2006. ACM.

[JSP05] Bernard J. Jansen, Amanda Spink, und Jan Pedersen. A temporal comparison of AltaVista Web searching: Research Articles. *J. Am. Soc. Inf. Sci. Technol.*, 56(6):559–570, 2005.

[JTi] JTidy. http://jtidy.sourceforge.net/.

[Kar] Kartoo. http://www.kartoo.com (nicht mehr verfügbar seit Januar 2010).

[KB11] Steffen Koch und Harald Bosch. From Static Textual Display of Patents to Graphical Interactions. In Mihai Lupu, Katja Mayer, John Tait, und Anthony J. Trippe, Editoren, *Current Challenges*

in Patent Information Retrieval, Volume 29 of *The Information Retrieval Series*, Seiten 217–235. Springer Berlin Heidelberg, 2011.

[KDH11] Ralf Krestel, Gianluca Demartini, und Eelco Herder. Visual Interfaces for Stimulating Exploratory Search. In *Proceedings of the 2011 Joint International Conference on Digital Libraries (JCDL 2011)*. ACM, June 13–17 2011.

[KJWS07] Fabian Kaiser, Mihály Jakob, Sebastian Wiedersheim, und Holger Schwarz. Framework-Unterstützung für aufwendige Websuche. *Datenbank-Spektrum*, Seiten 13–20, November 2007.

[KM01] Marja-Riitta Koivunen und Eric Miller. W3C Semantic Web Activity. In *Proceedings: Semantic Web Kick-Off in Finland*, 2001.

[KM06] Vijay Krishnan und Christopher D. Manning. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Seiten 1121–1128, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[KSJ06] Fabian Kaiser, Holger Schwarz, und Mihály Jakob. Finding Experts on the Web. In *Proceedings of the Second International Conference on Web Information Systems and Technologies, Setúbal, Portugal, April 11-13, 2006*, Seiten 363–368. INSTICC, April 2006.

[KSJ07] Fabian Kaiser, Holger Schwarz, und Mihály Jakob. EXPOSE: searching the web for expertise. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, Seiten 906–906, New York, NY, USA, 2007. ACM.

[KSJ09] Fabian Kaiser, Holger Schwarz, und Mihály Jakob. Using Wikipedia-based conceptual contexts to calculate document simi-

larity. In *ICDS2009: Proceedings of the 3rd International Conference on Digital Society*, Seiten 322–327. IEEE Computer Society, 2009.

[KSS+07] Fabian Kaiser, Sven Schimpf, Holger Schwarz, Mihály Jakob, und Severin Beucker, Editoren. *Internetgestützte Expertenidentifikation zur Unterstützung der frühen Innovationsphasen*. Fraunhofer IRB Verlag, September 2007. ISBN: 978-3-8167-7448-8.

[KTSW08] Dominik Kuropka, Peter Tröger, Steffen Staab, und Mathias Weske, Editoren. *Semantic Service Provisioning*. Springer, Berlin, 2008.

[Law06] Florian Laws. Entwicklung einer UIMA-basierten Softwarekomponente zur Identifizierung von Personennennungen in natürlichsprachigen Texten. Diplomarbeit, Universität Stuttgart, Fakultät Informatik, Elektrotechnik und Informationstechnik, Germany, Dezember 2006.

[Lev66] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, 1966.

[LGB99] Steve Lawrence, C. Lee Giles, und Kurt Bollacker. Digital Libraries and Autonomous Citation Indexing. *IEEE Computer*, 32(6):67–71, 1999.

[LH05] Ulf Leser und Jörg Hakenberg. What makes a gene name? Named entity recognition in the biomedical literature. *Briefings in Bioinformatics*, 6(4):357–369, December 2005.

[Lina] LingPipe. http://alias-i.com/lingpipe/.

[Linb] LinkedIn. http://linkedin.com.

[MA00] David W. Mcdonald und Mark S. Ackerman. Expertise recommender: a flexible recommendation system and architecture. Seiten 231–240, 2000.

[Mar06] Gary Marchionini. Exploratory search: from finding to understanding. *Commun. ACM*, 49:41–46, April 2006.

[Mit01] Ruslan Mitkov. Outstanding Issues in Anaphora Resolution (Invited Talk). In *CICLing '01: Proceedings of the Second International Conference on Computational Linguistics and Intelligent Text Processing*, Seiten 110–125, London, UK, 2001. Springer-Verlag.

[MLD+10] Lukas Michelbacher, Florian Laws, Beate Dorow, Ulrich Heid, und Hinrich Schütze. Building a Cross-lingual Relatedness Thesaurus using a Graph Similarity Measure. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, und Daniel Tapias, Editoren, *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may 2010. European Language Resources Association (ELRA).

[MMG99] Andrei Mikheev, Marc Moens, und Claire Grover. Named Entity recognition without gazetteers. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, Seiten 1–8, Morristown, NJ, USA, 1999. Association for Computational Linguistics.

[Mon] Monster. http://monster.de.

[MP01] Elaine Marsh und Dennis Perzanowski. MUC-7 Evaluation of IE Technology: Overview of Results. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, 2001.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, und Hinrich Schütze. *An Introduction to Information Retrieval*. Cambridge University Press, 2008.

[MS99] Christopher D. Manning und Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999.

[MTU+01] Diana Maynard, Valentin Tablan, Cristian Ursu, Hamish Cunningham, und Yorick Wilks. Named Entity Recognition from Diverse

Text Types. In *Recent Advances in Natural Language Processing 2001 Conference, Tzigov Chark*, 2001.

[MWC05] Einat Minkov, Richard C. Wang, und William W. Cohen. Extracting personal names from email: applying named entity recognition to informal text. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Seiten 443–450, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[Ope] OpenNLP. http://opennlp.sourceforge.net/.

[PD97] David D. Palmer und David S. Day. A statistical profile of the Named Entity task. In *Proceedings of the fifth conference on Applied natural language processing*, Seiten 190–193, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[Pin04] Brandon Pincombe. Comparison of human and LSA judgements of pairwise document similarities for a news corpus. Technical Report DSTO-RR-0278, DSTO, 2004.

[PM04] Fuchun Peng und Andrew Mccallum. Accurate information extraction from research papers using conditional random fields. In *HLT-NAACL 2004: Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Seiten 329–336, 2004.

[Pol85] Michael Polanyi. *Implizites Wissen*. Suhrkamp, 1985.

[Pre01] Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. McGrap-Hill, fifth edition, 2001.

[Res95] Philip Resnik. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *IJCAI*, Seiten 448–453, 1995.

[RFCa] RFC 2616: Hypertext Transfer Protocol - HTTP/1.1. http://www.ietf.org/rfc/rfc2616.txt.

[RFCb] RFC 2617: HTTP Authentication: Basic and Digest Access Authentication. http://www.ietf.org/rfc/rfc2617.txt.

[RFCc] RFC 2965: HTTP State Management Mechanism. http://www.ietf.org/rfc/rfc2965.txt.

[RGM01] Sriram Raghavan und Hector Garcia-Molina. Crawling the Hidden Web. In *VLDB '01: Proceedings of the 27th International Conference on Very Large Data Bases*, Seiten 129–138, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[Rog52] P. M. Roget. *Roget's Thesaurus of English words and phrases*. Available from Project Gutemberg, Illinois Benedectine College, Lisle IL (USA), 1852.

[Sau99] Dieter Sauer. *Paradoxien der Innovation : Perspektiven sozialwissenschaftlicher Innovationsforschung*. Campus-Verl., 1999.

[SBH+08] Stefanie Springer, Severin Beucker, Daniel Heubach, Fabian Kaiser, Dierk-Oliver Kiehne, und Mihály Jakob. Mit Softwaretools zu nachhaltigen Produkt- und Serviceinnovationen. *Ökologisches Wirtschaften*, 23(3), 2008.

[SC08] Philipp Sorg und Philipp Cimiano. Cross-lingual Information Retrieval with Explicit Semantic Analysis. In *Working Notes for the CLEF 2008 Workshop*, 2008.

[SdVC06] Ian Soboroff, Arjen P. de Vries, und Nick Craswell. Overview of the TREC 2006 Enterprise Track. In Ellen M. Voorhees und Lori P. Buckland, Editoren, *TREC*, Volume Special Publication 500-272. National Institute of Standards and Technology (NIST), 2006.

[SLWM04] Ruihua Song, Haifeng Liu, Ji-Rong Wen, und Wei-Ying Ma. Learning block importance models for web pages. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, Seiten 203–211, New York, NY, USA, 2004. ACM Press.

[SM86] Gerard Salton und Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[SMHM99] Craig Silverstein, Hannes Marais, Monika Henzinger, und Michael Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12, 1999.

[SMS+08] Dieter Spath, Bernhard Mitschang, Josef Schmid, Daniel Heubach, Severin Beucker, Holger Schwarz, Mihály Jakob, Fabian Kaiser, und Stefanie Springer. *nova-net: Innovation in der Internetökonomie*. nova-net Konsortium, 2008. Abschlussbericht.

[SP06] Michael Strube und Simone Paolo Ponzetto. WikiRelate! Computing Semantic Relatedness Using Wikipedia. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, Seiten 1419–1424, Boston, Mass., July 2006.

[Spr06] Stefanie Springer. *Nutzung von Internet und Intranet für die Entwicklung neuer Produkte und Dienstleistungen*. Fraunhofer IRB Verlag, 2006.

[Tem] Temis, Luxid. http://www.temis.com/.

[TKSDM03] Erik F. Tjong Kim Sang und Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Walter Daelemans und Miles Osborne, Editoren, *Proceedings of CoNLL-2003*, Seiten 142–147. Edmonton, Canada, 2003.

[Tri05] Matthias Trier. A Tool for IT-supported Visualization and Analysis of Virtual Communication Networks in Knowledge Communities. In Otto K. Ferstl, Elmar J. Sinz, Sven Eckert, und Tilman Isselhorst, Editoren, *Wirtschaftsinformatik 2005*, Seiten 963–983. Physica, Heidelberg, 2005.

[Val05] ValiWatch 2005 - Untersuchung zum deutschsprachigen Web. http://www.validome.org/lang/ge/html/valiwatch-web-2005, 2005.

[VC01] Martin Volk und Simon Clematide. Learn - Filter - Apply - Forget. Mixed Approaches to Named Entity Recognition. In *NLDB'01: Proceedings of the 6th International Workshop on Applications of Natural Language to Information Systems*, Seiten 153–163. GI, 2001.

[Viv] Vivisimo. http://www.vivisimo.com.

[VKV+06] Max Völkel, Markus Krötzsch, Denny Vrandecic, Heiko Haller, und Rudi Studer. Semantic Wikipedia. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, Seiten 585–594, New York, NY, USA, 2006. ACM Press.

[WDM+07] Ryen W. White, Steven M. Drucker, Gary Marchionini, Marti Hearst, und m. c. schraefel. Exploratory search and HCI: designing and evaluating interfaces to support exploratory search interaction. In *CHI '07 extended abstracts on Human factors in computing systems*, CHI '07, Seiten 2877–2880, New York, NY, USA, 2007. ACM.

[WHZ+07] Pu Wang, Jian Hu, Hua-Jun Zeng, Lijun Chen, und Zheng Chen. Improving Text Classification by Using Encyclopedia Knowledge. In *ICDM 2007: Proceedings of the Seventh IEEE International Conference on Data Mining*, Seiten 332–341, Oct. 2007.

[Wil08] Brian Wilson. MAMA: What is the Web made of? http://dev.opera.com/articles/view/mama/, 2008.

[WKB05] Ryen W. White, Bill Kules, und Ben Bederson. Exploratory search interfaces: categorization, clustering and beyond: report on the XSI 2005 workshop at the Human-Computer Interaction Laboratory, University of Maryland. *SIGIR Forum*, 39:52–56, December 2005.

[WR09] Ryen W. White und Resa A. Roth. Exploratory Search: Beyond the Query-Response Paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 1(1):1–98, January 2009.

[WYDM04] Wensheng Wu, Clement Yu, AnHai Doan, und Weiyi Meng. An interactive clustering-based approach to integrating source query interfaces on the deep Web. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, Seiten 95–106, New York, NY, USA, 2004. ACM.

[XIN] XING. http://xing.com.

[YSK03] Dawit Yimam-Seid und Alfred Kobsa. Expert-Finding Systems for Organizations: Problem and Domain Analysis and the DEMOIR Approach. *Journal of Organizational Computing and Electronic Commerce*, 13(1):1–24, 2003.