Indexierung und Fulcrum-Evaluierung

Krause, Jürgen; Mutschke, Peter

Veröffentlichungsversion / Published Version
Arbeitspapier / working paper

GESIS

IZ InformationsZentrum Sozialwissenschaften

IZ-Arbeitsbericht Nr. 17

Indexierung und Fulcrum-Evaluierung

Jürgen Krause, Peter Mutschke

Mai 1999

InformationsZentrum
Sozialwissenschaften

Lennéstraße 30
D-53113 Bonn
Tel.: 0228/2281-0
Fax.: 0228/2281-120
email: krause@bonn.iz-soz.de
mutschke@bonn.iz-soz.de
Internet: http://www.social-science-gesis.de

ISSN:	1431-6943
Herausgeber:	Informationszentrum Sozialwissenschaften der Arbeitsgemeinschaft Sozialwissenschaftlicher Institute e.V. (ASI)
Druck u. Vertrieb:	Informationszentrum Sozialwissenschaften, Bonn Printed in Germany

Das IZ ist Mitglied der Gesellschaft Sozialwissenschaftlicher Infrastruktureinrichtungen e.V. (GESIS), einer Einrichtung der Wissenschaftsgemeinschaft Gottfried Wilhelm Leibniz (WGL)

Inhalt

1 Einleitung

Kommerzielle Textretrievalsysteme basieren heute im wesentlichen auf intellektuell und/oder automatisch ermittelten Deskriptoren, die mit oder ohne zusätzliche Thesaurusrelationen mit Hilfe der Booleschen Algebra recherchiert werden (cf. Kap. 2). Dem stehen die quantitativ-statistischen Verfahren gegenüber, die zu nach Relevanz geordneten Ergebnislisten führen (ranking). Sie setzten sich in den letzten Jahren verstärkt auch im kommerziellen Bereich durch (cf. Kap. 3).

Die Boolesche Algebra läßt eine Verbindung der Anfrageterme mittels der logischen Operatoren AND, OR, NOT zu. Meist kommen formale Zusatztechniken wie Trunkierung (Rechts-, Links-, Innentrunkierung) oder Nachbarschaftssuche (zur engeren Definition des AND-Operators) hinzu, die ausschließlich durch Exact-Pattern-Match-Verfahren wirken.

Aus dem Blickwinkel der Inhaltserschließung (Indexierung) sollen die in beiden Entwicklungsrichtungen zugrundeliegenden Deskriptoren - unabhängig von jedem syntaktischen und hierarchischen Bezug (oder allgemeiner: ohne jede Relationierung untereinander mit Ausnahme einiger weniger Aspektangaben) - das Dokument inhaltlich charakterisieren. Die Benutzer verwenden die gleichen unrelationierten, inhaltskennzeichnenden Begriffe bei der Recherche.

Sucht man Ansatzpunkte für eine Verbesserung der Inhaltserschließungskomponente bzw. für eine möglichst wirtschaftliche Lösung, kann vom Retrieval nicht abstrahiert werden. Z.B. lassen sich Maßnahmen auf der Seite der Inhaltserschließung häufig gegenüber solchen auf der Rechercheseite austauschen. So wird eine bestimmte Art der Inhaltserschließung eventuell nur gewählt, um den Retrievalalgorithmus effizient gestalten zu können. Ein einfaches Beispiel ist die Trunkierungsfunktion; sie wird weitgehend überflüssig, wenn Kompositazerlegung und Grundformenreduktion bei der Inhaltserschließung eingesetzt werden. Kompositazerlegung und Grundformenreduktion lassen sich aber auch durch äquivalente Generierungsverfahren auf der Rechercheseite ersetzen. Ob intern ein Algorithmus das Suchwort des Benutzers zu allen Wortformen expandiert oder ob die Wortformen des Dokuments bei der Deskriptorenvergabe auf Grundformen reduziert werden, bemerkt der Benutzer im Idealfall nicht.

Vor diesem Hintergrund stellt sich die Frage nach dem adäquatesten Inhaltserschließungskonzept für ein Informationssystem. In Kapitel 2 werden daher die Vor- und Nachteile der intellektuelle Inhaltserschließung und der automatische Indexierung im Kontext von Booleschen und quantitativ-statistischen Recherchesystemen ganz generell diskutiert. In Kapitel 3 erfolgt eine Evaluierung der informationstechnologischen Grundlagen für ein modernes Informationssystem.

2 Intellektuelle Inhaltserschließung versus automatische Indexierung

2.1 Intellektuelle Inhaltserschließung versus automatische Indexierung im Rahmen von Booleschen Recherchesystemen

Bei der Inhaltserschließung von Literaturdatenbanken stehen sich heute zwei Grundprinzipien gegenüber: die intellektuelle und die automatische Indexierung. Hersh 1995:76 geht davon aus, daß „most modern commercial databases“ intellektuell indexiert sind. In vielen Fällen ist die intellektuelle Inhaltserschließung jedoch mit der Möglichkeit einer Freitextsuche verbunden, wodurch eine Mischform beider Konzepte entsteht. Der Text wird zweifach indexiert.

Die Einordnung als „automatisch indexiert“ sagt noch nichts über die Einbindung in ein Recherchemodell aus. Sie kennzeichnet nur, daß die Inhaltserschließung weitgehend ohne menschlichen Eingriff erfolgt. Werden automatische Indexierungsverfahren mit einem Booleschen Retrievalsystem verbunden, ermittelt der Indexierungsalgorithmus im einfachsten Fall die unterschiedlichen Zeichenketten eines Textes und eröffnet die Möglichkeit, einzelne Terme über Stoppwortlisten auszufiltern. Über computerlinguistische Verfahren läßt sich die Termermittlung z.B. durch die Zusammenführung von Wortformen zu einer Grundform, durch die Kompositazerlegung usw. weiter verbessern. Automatische Indexierung bei sogenannten Best-Match-Retrievalverfahren bedeutet dagegen, daß die Terme zusätzlich mit statistischen Informationen, z.B. über die Anzahl des Vorkommens eines Terms im Dokument oder in der Datenkollektion, versehen werden. Das Recherchemodul wertet diese Zusatzinformationen im Rahmen einer Formel aus, die die

Relevanz des Dokuments in bezug auf die gestellte Anfrage ermitteln soll, um eine nach Relevanz geordnete (=gerankte) Liste der Ergebnisdokumente ausgeben zu können.

Somit lassen sich die meisten Aspekte alternativer Inhaltserschließungsansätze nur im Kontext des zugeordneten Retrievalmodells diskutieren, wobei es in der Praxis einige Affinitäten zwischen der Ausprägung des Retrievalmodells und der Form der Inhaltserschließung gibt. Die intellektuelle Indexierung wird z.B. fast ausschließlich im Rahmen der Booleschen Recherche eingesetzt und diskutiert, obwohl intellektuell ermittelte Terme durchaus auch Best-Match-Verfahren ergänzen könnten.

2.1.1 Pro und Contra intellektuelle Indexierung

Intellektuelle Indexierungsverfahren verkörpern am klarsten die Grundidee einer möglichst weitgehenden Regulation, die Konsistenz erreichen soll. Alle vergebenen Begriffe (Ausnahme: Zusatzfeld für „freie Begriffe“) entnimmt der Indexierer einem ständig zu pflegenden Thesaurus. Die hier enthaltenen Terme bilden ein geschlossenes semantisches System. Ihr Vorteil ist, daß die Indexierungstiefe bei der Thesauruserstellung kontrolliert werden kann und eine semantische Vereinheitlichung bereits auf der Ebene der Inhaltserschließung erzwungen wird. Die Begriffe „verlieren“ allerdings ihre umgangssprachliche Semantik und sind quasi formalsprachlich in dem beschränkten, vorgegebenen Vokabular zu interpretieren. Diese Eigenschaft kontrollierter, intellektueller Indexierung wird am deutlichsten, wenn ein Indexierer einen Wunschbegriff im Thesaurus nicht findet.

Die Frage, ob eine intellektuelle Indexierung auf der Basis kontrollierter Thesauri einer automatischen qualitativ überlegen sei, führt zu den unterschiedlichsten Einschätzungen, die sich in der Regel auf Evaluationsergebnisse zu verschiedenen Studien aus dem englischsprachigen Raum stützen (cf. als Überblick Hersh 1995: chapter 7).

Blair/Maron 1985 schlossen aus ihren Evaluationen zu juristischen Dokumenten, die mit STAIRS indexiert wurden, daß frühere Evaluationsergebnisse, die für die intellektuelle Indexierung zu nachteiligen Ergebnissen gekommen waren, sich nicht auf reale, große Textbestände übertragen ließen. Sie konstatierten immer dann Vorteile für das intellektuelle Verfahren, wenn ein hoher Recall erwünscht sei:

> „fulltext system [=automatische Indexierung] means the additional cost of inputting and verifying 20 times the amount of information that a manually indexed system [= intellektuelle Indexierung] would deal with. This difference alone would more than compensate for the added time needed for manual indexing and vocabulary construction."

Salton 1986 wendet sich gegen diese Schlußfolgerung. Er kommt nach einer Sichtung verschiedener weiterer Studien und einer Analyse der Argumentation von Blair/Maron 1985 zu dem Schluß:

> „...these conclusions may be more sentiment than fact. ... the evidence from several retrieval evaluations conducted with very large collections does not support the notion of output overload ... comparisons between manual and automatic indexing systems on large document collections indicate that the automatic-text-based systems are at least competitive with, or even superior to the systems based on intellectual indexing." (Salton 1986:650)

D. R. Swanson, den Salton 1986:656 mit seiner Arbeit von 1960 noch zitiert als „probably the earliest result showing the superiority of automatic text searching", kommt in Swanson 1988:95 zu dem Schluß:

> „Machines cannot recognize meaning and so cannot duplicate what human judgement in principle can bring to the process of indexing and classifying documents ... Consistently effective fully automatic indexing and retrieval is not possible"

Dahlberg 1996:82 formuliert die Überzeugung der Überlegenheit einer intellektuell durch Fachwissenschaftler vorgenommenen Begriffswahl überspitzt, aber gerade dadurch klar verständlich:

> „... vermeinte man seit mehr als 30 Jahren nun schon, mit dem Computer das menschliche Denkenmüssen ersetzen zu können. Gott sei Dank geht das nicht -... bei geschriebenen Texten, in denen die Wortwahl eines Autors mit ihren vielfachen Möglichkeiten eine unendlich große Vielfalt erlaubt, Gedanken in Worte zu kleiden, handelt es sich um eine unvorhersehbare Situation, die auch als indeterminierter Prozeß bezeichnet werden kann, der sich bekanntlich jeglicher befriedigenden Programmierung entzieht."

Es ist wichtig zu erkennen, daß die obigen generellen Kritikpunkte gegen die intellektuelle Indexierung in der Regel nicht das dahinter stehende informationswissenschaftliche Prinzip in Frage stellen. Es geht entweder um Erweiterungen und Zusätze (Behandlung vager Informationen, Ranking u.a.) oder um Alternativen, die wegen der Kosten der intellektuellen Verfahren vorgeschlagen werden, welche bei einer steten Ausdehnung des Umfangs der zu erschließenden Dokumente als nicht mehr tragbar scheinen.

Die DATEV-Datenbanken sind ein frühes Beispiel für die „Reinform" traditioneller automatischer Freitextsysteme (cf. DATEV 1994). Mit Ausnahme einiger aspektgebundener Deskriptoren regelt nur eine Stoppwortliste die Auswahl. Ein weiteres Beispiel sind die JURIS-Datenbanken (Basis GOLEM/PASSAT, cf. Möller 1993). Hier ist interessant, daß nach über 15jähriger praktischer Erfahrung vehement die bei Entwicklungsbeginn abgelehnte intellektuelle Erschließung wieder als Heilmittel für die empirisch beobachteten unbefriedigenden Retrievalleistungen von JURIS angesehen wird (cf. Wolf 1992; Möller 1993).

Per se ist somit das Modell der intellektuellen Inhaltserschließung durchaus ein gangbarer Weg, vor allem wenn es durch die Verbindung mit der Freitextrecherche zu einem Mischsystem von intellektueller und automatischer Indexierung erweitert wird. Was sich geändert hat, sind die Rahmenbedingungen. Der technologische, wirtschaftliche, politische und gesellschaftliche Wandel der letzten Jahre brachte Strömungen und Meinungen hervor, die zum Modell der Informationsservicestellen - wie es sich in den letzten 20 Jahren mit seinen Dienstleistungen konsolidiert hat - in einigen Punkten in Widerspruch geraten (cf. Krause 1996b). Die Irritationen reichen bis in die Wahl der Verfahren zur Inhaltserschließung hinunter, da Entscheidungen in diesem Bereich auch organisatorische und wirtschaftliche Rahmenbedingungen tangieren.

2.1.2 Meinungen über die Datenkonsistenz intellektueller Erschließungsverfahren

Als detaillierte Kritikpunkte gegen einen der Hauptvorteile der intellektuellen Indexierung, der erreichbaren Qualität der Datenkonsistenz, werden genannt:

- Die Kosten der intellektuellen Aufbereitung sind zu hoch.

Häufig überschätzen Außenstehende die tatsächlich anfallenden Zusatzkosten. Im Schnitt bezahlen Servicestellen für die intellektuelle Deskriptorenvergabe etwa 22,00 DM pro Dokument. Mehr ließe sich bei der Abstrakterstellung sparen (etwa 35,00 DM pro Dokument). So lange es sich in einigen Bereichen jedoch nicht durchsetzen läßt, daß zumindest Autorenzusammenfassungen üblich sind, kann dieser Posten nicht eingespart werden. Auch eine automatische Indexierung braucht zumindest diese Textgrundlage. Der Anteil von Zeitschriftenaufsätzen mit Autorenreferaten liegt aber z.B. bei den Daten der sozialwissenschaftlichen Literaturdatenbank SOLIS nur bei 20 %. Die Alternative Volltextindexierung kommt wegen der Rechtslage derzeit in vielen Bereichen nicht in Frage. Die Verlage erlauben die Wiedergabe vollständiger Zeitschriftaufsätze und Bücher nicht, weil sie um den Absatz ihrer Printprodukte fürchten (wohl zu recht).

Mittelfristig gesehen könnte sich bei beiden Alternativen rasch etwas ändern. Es sollte auch versucht werden, diese Veränderungen zu induzieren.

- Die hohe Qualität und vor allem die Datenkonsistenz der intellektuellen inhaltlichen Erschließung durch Fachwissenschaftler an Informationszentren wird bestritten:

    * Jeder Fachwissenschaftler indexiere nach eigenen Kriterien und Wissenshintergrund. Die Konsistenz werde postuliert, aber in der Praxis nicht erreicht.

    * Fachwissenschaftler an Informationsservicestellen verlieren die Verbindung zu neueren Entwicklungen. Zumindest indexieren sie im Geist früherer Ansichten. Deshalb sollte eine zentrale Informationsstelle zumindest weitgehend die fachwissenschaftliche Kompetenz universitärer Forschungsstellen nutzen, um einer wachsenden Fachferne entgegenzuwirken.

- Benutzer solcher Systeme sehen in der Regel nicht in der Schlagwortliste nach, sondern formulieren ihre Anfrage direkt. Wird der Thesaurus nicht ständig gepflegt und um neue (Mode-)Begriffe eines Fachgebiets erweitert, ergeben sich zu große Diskrepanzen zwischen den vom Benutzer gewählten Begriffen und den Thesaurusstrukturen.

Interessant ist, daß all diese Detailargumente wiederum nicht prinzipieller, sondern wirtschaftlicher Natur sind. Bei einem höheren Geldeinsatz und veränderten Organisationsformen ließen sich die angegebenen Nachteile - so sie

denn empirisch nachweisbar wären - durchaus beseitigen, ohne das zugrunde liegende Postulat der intellektuellen Indexierung aufzugeben.

Einigen der Argumente läßt sich auch durch die Koppelung mit der Freitextrecherche begegnen. So kann der Benutzer bei einem Mischsystem die konsistenzstiftenden Vorteile des kontrollierten Vokabulars nutzen und die Freitextrecherche immer dann zuschalten, wenn er vom Autor gewählte Begriffe im Text direkt finden möchte.

2.1.3 Grenzen der intellektuellen Indexierung als zentralistisch orientiertes Verfahren

Nicht unerheblich für die Ablehnung der intellektuellen Indexierung dürfte sein, daß sie zweifellos von allen Ansätzen die weitestgehende Regulation voraussetzt, deren Regeln wiederum eine zentrale Instanz vorgibt. An sie ist die intellektuelle Indexierung in bezug auf die Durchsetzungsfähigkeit gebunden. Alle „Mitspieler“ eines Anwendungsgebiets müssen somit von der Richtigkeit dieses Vorgehens überzeugt werden. Weltweit gesehen, scheint dies zumindest im Kontext internationaler Kooperationen im Internet eine Illusion. Der breite Zugang zu den Netzwerken wirkt einer zentralistischen Doktrin der Informationserschließung per se entgegen. Überall auf der Welt können Gruppen auftreten, die zu Spezialgebieten Informationen sammeln. Der Benutzer wird auf sie zugreifen wollen, gleich nach welchen Verfahren sie erschlossen oder in welchem System sie angeboten werden. Die zuständige Informationsservicestelle müßte deshalb mit diesen Anbietern Kontakt aufnehmen und sie überzeugen, bestimmte Normen der Inhaltserschließung einzuhalten. Das mag im Einzelfall funktionieren, jedoch nie als generelle Strategie. Es wird immer eine Fülle von Angeboten geben, die sich vorgegebenen Leitvorstellungen nicht unterordnen lassen. Früher lehnten die zentralen Informationsservicestellen Dokumente ab, die nicht bestimmte Regeln der Erschließung einhielten, wodurch der Benutzer (idealiter) immer einem homogenisierten Datenbestand gegenüberstand. Darauf ist die gesamte IuD-Methodik, einschließlich der Verwaltungsstruktur der Zentren, ausgerichtet. Ob man dies für richtig oder falsch hält, diese Ausgangssituation ist in einem System weltweiter Vernetzung nicht mehr gegeben.

2.1.4 Fazit intellektuelle Indexierung versus automatische Indexierung für Boolesche Retrievalsysteme

Die Frage nach der effizientesten Art der Inhaltserschließung muß heute als offen angesehen werden. Sie läßt sich nur durch eine anwendungsbezogene, vergleichende Evaluation der verschiedenen vorgeschlagenen Verfahren klären und ist zudem stark mit der Verbreitung von Autorenabstracts in einem Fachgebiet und mit der rechtlichen Ausgangssituation bei der Verwertung von Volltexten verknüpft.

Wie in Kap. 2.2 deutlich werden wird, herrscht die gleiche Unsicherheit, die hier im Kontext des Booleschen Retrievals diskutiert wurde, auch bei der Alternative der statistisch-quantitativen Ansätze, die die heutigen Standardverfahren nicht ergänzen, sondern ersetzen würden.

In der Praxis ist allerdings davon auszugehen, daß bei neu aufzubauenden Datenbeständen wegen der höheren Kosten automatische Indexierungsverfahren präferiert werden.

Es hat sich jedoch gezeigt, daß man zumindest bei bereits intellektuell indexierten Datenbeständen, die weiter gepflegt werden, mit der Lösung, intellektuell zu indexieren, in der Recherche jedoch die Freitextsuche zusätzlich zuzulassen, auf der sicheren Seite der Argumentation ist. Bis zu einem gewissen Grad gleichen sich die Nachteile beider Grundverfahren bei diesem Mischansatz aus.

a) Die intellektuelle Indexierung ist teurer als die automatische. Gleichzeitig weiß man, daß die Vor- und Nachteile beider Verfahren je nach Anwendungsgebiet und Textgrundlage unterschiedlich stark ausgeprägt sein können. Deshalb müssen die generellen Aussagen auf der Grundlage der zu indexierenden Fachtexte überprüft werden. Es ist empirisch zu klären, ob den Mehrkosten ein vernünftiger Qualitätsvorteil gegenübersteht. Bisher stützen sich die Datenbankanbieter in der Regel auf Plausibilitätsargumente, die aus Besonderheiten der Textgrundlage abgeleitet sind.

b) Umfangreichere neue Gebiete wie z.B. Internetquellen können in der Regel nicht mehr intellektuell erschlossen werden, da hierzu die Geldmittel fehlen. Deshalb muß sich jeder Datenbankanbieter unabhängig von der Problemstellung in a) mit der Frage auseinandersetzen, welche der vorgeschlagenen automatischen Indexierungsverfahren für die Dokumente seines Fachgebiets am geeignetsten sind. Automatische Indexierungsverfahren werden in Zukunft auch dann zum Einsatz kommen, wenn empirische

Tests mit fachgebietsspezifischen Daten qualitative Vorteile zugunsten des Mischsystems mit intellektueller Erschließung aufzeigen.

Entsprechendes gilt, wenn die Datenbanken in ein weltweites Netz von Datenbeständen integriert werden, deren Produzenten sich weitgehend dem Einfluß jeder Pränormierung entziehen (Schalenmodell).

Im Rahmen der Verwendung eines Mischsystems auf der Basis des Booleschen Retrieval ist zu fragen, welche Freitextkomponente die besten Ergebnisse liefert. Daß Komponenten dieser Art zu besseren Ergebnissen führen, gilt heute jedoch als weitgehend gesichert (cf. Krause 1996a).

2.2 Automatische Indexierung im Rahmen quantitativ-statistischer Verfahren

Die quantitativ-statistischen Verfahren des IR (unter Bezeichnungen wie Best-Match bzw. Nearest-Neighbour-Methoden mit oder ohne Relevance Feedback) verändern den Retrievalprozeß gegenüber dem Standardmodell des IR tiefgreifend. Sie verstehen sich als Techniken vor allem gegen die folgenden negativen Eigenschaften des Booleschen Retrieval:

- Das Boolesche Retrieval teilt den Dokumentenbestand - ohne jede Zwischenstufen - in zwei diskrete Untermengen: in Dokumente, die den „exact match“ erfüllen (= die relevanten Dokumente), und solche, die es nicht tun. Dokumente mit drei gefundenen Termen werden in einer aus vier, mit dem UND-Operator verknüpften Termen bestehenden Suchanfrage genauso zurückgewiesen, wie solche ohne jede Übereinstimmung.

- Alle ausgebenen Dokumente sind aus Systemsicht gleichwertig. Das letzte Dokument der Ergebnisliste kann das Informationsbedürfnis des Benutzers durchaus am besten erfüllen. Auf der Deskriptorenebene entspricht dem der Zwang, alle Deskriptoren als „gleich wichtig“ anzusetzen, was Benutzer als unzulässige Vereinfachung ansehen.

- Benutzer haben häufig Probleme, die logischen Operatoren UND, ODER und NICHT adäquat einzusetzen. Ein Grund hierfür ist, daß die Semantik der logischen Operatoren nicht mit der Semantik der natürlichsprachlichen Terme übereinstimmt. Außerdem müssen die Prioritäten der Booleschen Operatoren bekannt sein.

Erst mit der starken Verbreitung des Internets setzen sich außerhalb der wissenschaftlichen Beschäftigung mit dem IR langsam einfache Formen der statistisch basierten Verfahren durch (z.B. FREEWAIS, INQUERY, TOPIC, Fulcrum). Gleichzeitig kapitulieren die Suchmaschinen des Internet jedoch vor der Fülle der Daten und ihrer Heterogenität; ihre Entwickler kehren zu einfachsten Verfahren und zu den hierarchischen Klassifikationen, die intellektuell vorzunehmen sind, zurück (cf. Ulisch 1997).

2.2.1 Grundsätzliche Gemeinsamkeiten: Ranking der Ergebnisliste und Deskriptorenreihung

Nicht-Boolesche Retrievalmodelle lassen sich hinsichtlich des theoretischen Hintergrundes in probabilistische (statistische Wahrscheinlichkeitstheorie), vektorielle (Vektorraummodell) und Fuzzy-Retrievalmodelle (Theorie unscharfer Mengen) unterscheiden, die die Ähnlichkeitsfunktion verschieden interpretieren. Wie empirische Ergebnisse aus den TREC-Studien zeigten, wirken sich die theoretischen Unterschiede kaum auf die Retrievalergebnisse aus (cf. Womser-Hacker 1996: Kap. 5), weshalb man sich für ein erstes Verständnis auf die allen Ansätzen gemeinsame Grundarchitektur zurückziehen kann.

Best-Match-Verfahren lassen sich dadurch charakterisieren, daß der Benutzer bei der Anfrage Deskriptoren, ohne Boolesche Operatoren zu verwenden, aneinanderreiht und die relevantesten Dokumente zu Beginn der Ergebnisliste stehen sollen. Dieses sogenannte *ranking* wird vom System aufgrund von Ähnlichkeitskriterien erzeugt.

2.2.1.1 Ähnlichkeitsermittlung zwischen Anfrage und Dokument

Die vom System ermittelten Ähnlichkeiten legen die Reihenfolge der Dokumente in der Ergebnisliste fest. Am weitesten verbreitet ist das sogenannte „vector dot product“, bei dem sich die Ähnlichkeit aus der Produktsumme der (Gewichte der) Terme berechnet, die in Anfrage und Dokument gemeinsam vorkommen. Je höher der ermittelte Wert ist, um so weiter oben steht das Dokument in der Ergebnisliste. Häufig wird eine Mindestähnlichkeit durch einen bestimmten Schwellenwert (z.B. 0,5) festgelegt oder auch die benutzerseitig definierte Anzahl der gewünschten Dokumente als Begrenzungskriterium herangezogen.

2.2.1.2 Gewichtung

In der Regel geht zumindest eine Gewichtung der Dokumentterme in das Ähnlichkeitsmaß ein. Sie wird für jeden Term in bezug auf bestimmte quantitative Eigenschaften der Dokumente bzw. der Dokumentenkollektion automatisch bestimmt. So basiert z. B. das „inverse document frequency"-Gewichtungsmaß auf der Anzahl der Dokumente in der Datenbank N und der Frequenz des Terms t in der Dokumentenkollektion. Die Berechnungsformel $G = \log(N/F(t))$ bewirkt, daß allgemeine Terme (quantitatives Kennzeichen: hohe Frequenz) weniger zur Relevanz eines Terms beitragen als spezifische Terme, die selten vorkommen. Das Maß kann zusätzlich die Frequenz eines Terms in einem Dokument (with-in term frequency) mit einbeziehen. Je häufiger t in einem Dokument vorkommt, ein um so stärkeres Gewicht hat es, und je seltener es dann auch noch in den anderen Dokumenten auftritt, um so besser.

Einige Gewichtungsmaße berücksichtigen auch die Anzahl der verschiedenen Terme innerhalb eines Dokuments und/oder legen Grenzen für die Vorkommenshäufigkeit eines Terms fest (z.B.: t muß mindestens dreimal in der Datenkollektion auftreten). Nahe liegt weiter, textsortenspezifische Gewichtungsregeln einzuführen, z. B. Terme in Überschriften höher zu gewichten als andere. In diesen anwendungsabhängig festzulegenden Varianten scheint ein großes Verbesserungspotential für die Ansätze des nicht-Booleschen Retrieval zu liegen.

2.2.1.3 Relevance Feedback

Das Verfahren setzt eine mindestens zweistufige Anfrage und die Mitarbeit des Benutzers voraus. Er bewertet die Ergebnisliste, indem er z. B. bei INQUERY ankreuzt, wenn ein ausgegebenes Dokument für ihn „relevant" war. Das System nutzt dieses dynamische Kontrollwissen aus der aktuellen Dialogsituation, um die ursprüngliche Anfrage „neu zu berechnen". Anfrageterme, die häufig in als „relevant" angegebenen Dokumenten vorkommen, bekommen ein höheres Gewicht bzw. werden sie der Ursprungsanfrage hinzugefügt. Hierdurch soll die modifizierte Ergebnisliste dem Informationsbedürfnis des Benutzers näher kommen.

2.2.1.4 Fazit

Ansätze wie sie oben vorgestellt wurden, lassen sich als eine Form der automatischen Indexierung sehen, bei der gleichzeitig die Einbettung in das Boolesche Retrieval durch Best-Match-Verfahren ersetzt wird. Dieses In-

haltserschießungs- und Retrievalmodell gilt als der derzeit erfolgversprechendste IR-Ansatz ohne weitgehende intellektuelle Eingriffe bei der Inhaltserschließung.

2.3 Vorgehensweise zur Weiterentwicklung der Inhaltserschließung

Die offenen Fragen nach der leistungsfähigsten Inhaltserschließung fachspezifischer Dokumente und IR-Entwicklungstategien lassen sich nur auf der Basis einer Reihe von empirischen Tests für ein spezifisches Anwendungsfeld beantworten. Dabei sollte eine Mischform aus intellektueller und automatischer Indexierung als konzeptueller Ausgangspunkt dienen. Es gibt keine vernünftigen Gründe, wieder auf eine rein intellektuelle Dokumenterschließung zurückzugehen. Die als erster Schritt zu testende Alternative ist somit die Mischform aus Freitextrecherche und intellektueller Indexierung im Kontext eines booleschen Retrievalsystems, d.h. boolesche Suche mit Deskriptoren und Freitextbegriffen, versus der automatischen Indexierung im Kontext eines statistischen Retrievalsystems.

In diesem Kontext ist zu fragen, ob sich die Freitextkomponente als Bestandteil der Mischform verbessern läßt, wenn z.B ein System ohne computerlinguistische Algorithmen arbeitet. Daß Komponenten dieser Art zu besseren Ergebnissen führen, gilt heute als weitgehend gesichert (cf. Krause 1996a), ist jedoch auf der Basis der fachspezifischen Texte zu überprüfen.

Als zu testendes Modell einer automatischen Indexierung bietet sich ein quantitativ-statistischer Ansatz an, wobei gleichzeitig die Einbettung in das Boolesche Retrieval durch Best-Match-Verfahren ersetzt wird. Diese Verbindung von Inhaltserschließungs- und Retrievalmodell gilt als der derzeit erfolgversprechendste IR-Ansatz ohne weitgehende intellektuelle Eingriffe bei der Inhaltserschließung.

Auch wenn sich bei diesen Tests Vorteile für das Mischsystem ergeben sollten, wird die automatische Indexierungsvariante nicht uninteressant. Umfangreichere neue Gebiete wie z.B. Internetquellen lassen sich nicht mehr intellektuell erschließen, da hierzu die Geldmittel fehlen. Deshalb muß man sich auch in diesem Fall mit der Frage auseinandersetzen, welche der vorgeschlagenen automatischen Indexierungsverfahren für spezifische Anwendungsfelder am geeignetsten sind (Schalenmodell, cf. Krause 1996a).

Ein Vergleichstest zwischen der Mischform und einem quantitativ-statistischen Verfahren kann nur der Ausgangspunkt für weitere Evaluationsschritte sein, in denen die für ein spezifisches Anwendungsfeld günstigste Ausprägung eines quantitativ-statistischen Verfahrens ermittelt wird. Am erfolgversprechendsten scheint dabei die Integration heuristischer Faktoren in bestehende quantitativ-statistische Ansätze. Hierbei geht es u.a. um die Frage, ob sich formalstrukturelle Texteigenschaften für die Dokumentgewichtung ausnutzen lassen. Die Grundüberlegung ist, daß sich wichtige Informationen eher in strukturell hervortretenden Elementen eines Dokuments wie im Titel, den Überschriften, Zusammenfassungen oder Tabellen befinden.

Generell wäre natürlich auch bei Best-Match-Verfahren eine Anreicherung der automatischen Indexierungskomponente durch intellektuell ermittelte Deskriptoren denkbar, die dann höher gewichtet würden.

Darüber hinaus sollte getestet werden, ob sich die effizienteste Variante des Mischverfahrens mit der des quantitativ-statistischen Verfahrens im Sinne eines Extended Booleschen Retrieval performanzsteigernd miteinander verbinden lassen.

Nach Abschluß der bisher eingeführten Evaluationsschritte wäre für ein spezifisches Fachgebiet eine empirisch fundierte Entscheidung möglich, welches Inhaltserschließungs- und Recherchemodell die besten Performanzwerte verspricht bzw. welche Performanzsteigerung mit welchem Kostenfaktor verbunden ist.

Gleichzeitig weiß man jedoch, daß auch innerhalb spezifischer Fachwissenschaften Differenzierungen zwischen einzelnen Benutzergruppen und Themengebieten vorliegen, und daß sich zugleich für die Inhaltserschließung wesentliche Randbedingungen - wie die Verwendung der Fachterminologie - in der Zeit verändern.

Vor diesem Hintergrund empfiehlt es sich, daß in Informationssystemen alle drei Indexierungs- und Retrievalvarianten zur Verfügung stehen:

- Boolesche Suche mit intellektuell vergebenen Deskriptoren
- Boolesche Suche mit Freitextbegriffen
- Statistisches Retrieval auf der Basis automatischer Indexierung.

3 Informationstechnologische Grundlagen

3.1 Datenbanksysteme

Die Wahl einer bestimmte Datenbanksoftware ist eine Entscheidung von grundsätzlicher strategischer Bedeutung. Sowohl die Datenbankmaschine selbst als auch die Entwicklungswerkzeuge bilden die technologische Plattform für die Umsetzung eines integrierten, informationswissenschaftlich elaborierten Gesamtkonzepts für den Umgang mit allen relevanten Unternehmensdaten. Vor diesem Hintergrund wurden in einer am Informationszentrum Sozialwissenschaften durchgeführten Studie die gängigsten relationalen Datenbanksysteme für größere UNIX-Plattformen untersucht und bewertet.

Die Entscheidung für ein bestimmtes Produkt kann sich nicht an Funktionalität und Performance des Database Management Systems (DBMS) allein orientieren. Entscheidungsrelevant sind ebenso die zusätzlichen, sich um den Datenbankkern herum gruppierenden Komponenten und Werkzeuge, die eine integrierte Anwendungsentwicklung ermöglichen oder die Voraussetzung bilden für die Umsetzung eines unternehmensweiten integrierten Client/Server-Konzepts. Dies gilt um so mehr als sich die Datenbanksysteme hinsichtlich der Kernfunktionalität weitgehend ähneln.

Eine zentrale Grundanforderung an ein modernes Datenbanksystem ist allerdings hohe Flexibilität auf der Modellierungsseite. Diese Grundanforderung führt direkt zu einem relationalen oder sogar objektorientiertem Datenmodell. Die Vor- und Nachteile dieser beiden Konzepte sollen daher kurz skizziert werden.

3.1.1 Vor- und Nachteile relationaler Datenbanktechnologie

Mit der Relationentheorie von Codd[1] wurde die Datenorganisation und -manipulation auf eine mathematisch präzise Grundlage gestellt (Mengenlehre, Normalisierungslehre, Synthesealgorithmen). Relationale Datenbanken erlauben die Implementierung von redundanzfreien und damit konsistenten Datenmodellen und eine mengenorientierte Datenverarbeitung. Darin unterscheiden sich relationale Datenbanken grundsätzlich von hierar-

[1] Vgl. hierzu: Schlageter/Stucky (1987), Ullmann (1988), Sauer (1992).

chischen und netzwerkorientierten Datenbanken. Mit dem Relationenmodell als konzeptionelles Datenbankschema sind folgende Vorteile verbunden:

- Sehr einfaches Datenmodell
- Garantierte Datenkonsistenz (bei normalisierten Modellen), da Änderungen nur an einer Stelle vorgenommen werden
- Beliebige Einstiegspunkte in die Datenbank, da alle Relationen „gleichberechtigt“ sind (keine ausgewiesenen Einstiegspunkte wie bei hierarchischen und netzwerkorientierten Datenbanken)
- Verknüpfung über Inhalte
- Da jedes Objekt in Einzelteile zerlegt wird und die Daten mengenorientiert verarbeitet werden, sind komplizierte Abfragen auf große Datenmengen und mächtige Operationen gegen die Daten möglich (s. Quantoren)
- Hohes Maß an Datenunabhängigkeit: kein Navigieren in der Datenbank durch den Benutzer; die „Navigation“ obliegt völlig dem Datenbanksystem; die Sicht der Benutzer auf die Daten wird abstrakter; es sind weniger Kenntnisse über die physische Datenorganisation erforderlich
- Standardisierte einheitliche Datenbanksprache (SQL).

Gerade aufgrund der garantierten Datenkonsistenz und der flexiblen Abfragemöglichkeiten haben sich relationale Datenbanksysteme zum de-facto-Industriestandard entwickelt.[2]

Die Vorteile des relationalen Ansatzes sind aber zugleich auch seine Schwächen:

- Beziehungstypen werden wie Entity-Typen in Relationen (Tabellen) abgelegt. Dies ist eine komfortable Vereinfachung, bedeutet aber, daß semantische Beziehungen zwischen Entitäten, wie sie im Entity-Relationship-Modell definiert werden, im Relationenmodell nicht explizit ausgedrückt werden können („semantische Lücke“).
- Is-a- und Part-of-Beziehungen lassen sich ebenfalls nicht ausdrücken, da es im klassischen Relationenmodell nicht möglich ist, den Tupeln wiederum Sub-Tupel zuzuordnen.

---

[2] Vgl. auch Gerkens (1994).

- Die bei der Normalisierung erforderliche Zerlegung der Daten in Einzelteile erschwert inhaltlich und unter Performance-Gesichtspunkten die Suche nach komplexeren Zusammenhängen: Daten, die getrennt voneinander gespeichert werden, aber gemeinsam zu benutzen sind, um Informationsobjekte zu erzeugen, so wie der Anwender sie kennt, müssen auf der Anwendungsebene in Form von Joins wieder zusammengeführt werden.
- Es läßt sich wenig Wissen über die Regeln und Funktionen speichern, die mit den Daten verbunden sind.

3.1.2 Vor- und Nachteile objektorientierter Datenbanken

Mit objektorientierten Datenmodellen werden dagegen drei grundlegende Konzepte in die Datenbankentechnologie eingeführt, die die Nachteile des Relationenmodells weitgehend auffangen[3]:

- Gegenstände und Aspekte, d.h. Objekte eines zu modellierenden Weltausschnitts sind sehr oft selbst aus anderen Objekten in beliebiger Weise zusammengesetzt (Teil-von-Beziehung bzw. *use*-Relation zwischen Objekten). Um diese Zusammenhänge möglichst vollständig, d.h. 1:1 in der Datenbank abbilden zu können, ist eine hohe Ausdrucksmächtigkeit des Datenmodells erforderlich, die herkömmliche Datenbanksysteme mit ihrer geringen Strukturierungstiefe nicht oder nur eingeschränkt erfüllen können. Objektorientierte Datenbanken gestatten dagegen die Definition von beliebig komplexen Objekten und erlauben damit eine realitätsnahe Modellierung mit hoher Strukturkomplexität. Auf diese Weise wird eine vereinfachte Sicht der Welt abgebildet, die die semantische Interpretierbarkeit des Modells erhöht.
- Mit den Objekten zusammen können Objektmethoden (Wissen über die Daten) hinterlegt werden, durch deren Anstoßen Verarbeitungsoperationen, die zusammen mit den Daten definiert werden, automatisch ablaufen.
- Klassifikation kann explizit ausgedrückt werden: Eigenschaften bestehender Klassen können an neue (Unter-)Klassen vererbt werden, so daß Operationen nur einmal kodiert werden müssen.

Aber auch mit objektorientierten Datenbanken sind spezifische Probleme verbunden:

---

[3] Vgl. hierzu: Drucks (1992), Bauer (1995a), Schmatz/Weikert (1995).

- Objektorientierte Datenbanksysteme sind noch nicht marktreif.
- Es gibt bislang noch keine mathematische Fundierung objektorientierter Datenmodellierung, die mit der Relationentheorie vergleichbar wäre, sondern nur allgemeine Gestaltungsprinzipien. Die Qualität und der semantische Informationsgehalt objektorientierter Datenbanken steht und fällt damit mit dem Klassenmodell, das - in weitaus stärkerem Maße als das Relationenmodell - eine individualistische Konstruktion ist. Zusammenhänge zwischen Entitäten müssen fest verpointert werden, womit viele Probleme semantischer Netzwerke v.a. hinsichtlich der Modifizierbarkeit des Modells wieder eingeführt werden (Nethacking).
- Die Flexibilität der Abfragemöglichkeiten des Relationenmodells ist hier wieder aufgegeben worden, da die Abfragemöglichkeiten auf die Objekte eingeschränkt sind, die auch modelliert wurden.
- Objektorientierte Datenbanken haben notwendigerweise einen ungleich höheren Speicherbedarf als relationale Datenbanken, da mehr Information hinterlegt werden muß.
- Es gibt Probleme mit der Identität von Objekten.

Eine Variante, die die Vorteile relationaler und objektorientierter Modellierungstechniken integrieren würde, sind sogenannte *objekt-relationale* Systeme, bei denen auf ein Relationenmodell eine objektorientierte Schicht aufgesetzt wird[4]. Bei der Auswahl eines Datenbanksystems ist daher auch nach den Weiterentwicklungsoptionen in Richtung Objektorientierung zu fragen.

3.1.3 Anforderungen an ein relationales Datenbanksystem

Über ein relationales Datenmodell und bestimmte Kernfunktionalitäten des DBMS (Datenintegrität und -sicherheit, Performance etc.) hinaus sind folgende informationstechnologische Anforderungen an ein modernes relationales Datenbanksystem in besonderer Weise relevant:

- Verteilte Datenhaltung und Zugriff auf verteilte, heterogene Datenbanken:

  Die geographische und inhaltliche Aufgabenverteilung innerhalb der Verbände erfordern komfortable und mächtige Möglichkeiten der verteilten

[4] Vgl. auch Drucks (1992), Vorwerk (1995).

Datenhaltung und ggf. sogar Möglichkeiten eines homogenen Zugriffs auf verteilte, heterogene Datenquellen.

- Interaktive graphische Browsing-, Datenanalyse- und Reportingwerkzeuge:
  Die fehlende Möglichkeit, in benutzerfreundlicher Form beliebige ad-hoc-Anfragen an die Datenbank zu stellen, die nicht auf fest definierte Masken eingeschränkt sind, sowie statistische Analysen und Berichte erstellen zu können, ist das Grunddilemma der zu Zeit eingesetzten Datenbanktechnologie.
- Integrierte Textretrievalfähigkeit:
  Im Zusammenhang mit der Integration von heterogenen Datenbestände stellt sich das Problem der Kombination von strukturierten und unstrukturierten Daten. So muß es z.B. möglich sein, Textdokumente innerhalb der gleichen Datenbankumgebung zu speichern, in der sich auch die strukturierten Daten befinden, so daß sich innerhalb eines SQL-Befehls Abfragen auf strukturierten Datensätzen mit Volltext-Recherche verbinden lassen.[5] Dieses Problem stellt sich auch im Hinblick auf den Aufbau eines *Information Warehouse*, das aggregierte Information für den alltäglichen Informationsbedarf zur Verfügung stellt[6].
- Ausnutzung paralleler Rechnerarchitekturen, insbesondere die parallele Verarbeitung von Datenbankanforderungen
- Mächtige graphische Frontend-Tools, die eine visuelle Modellierung von Datenbanken und Arbeitsabläufen (inkl. Anwendungsgenerierung) sowie eine integrierte und portable Entwicklung von Datenbankoberflächen und (komplexeren) Berichten erlauben.
- Hardware- und Betriebssystemunabhängigkeit
- Unterstützung von Workflowsystemen
- Internet-Anbindung, WWW-Fähigkeit
- Support und Zukunftssicherheit (technologische und wirtschaftliche Potenz des Herstellers).

---

5 Vgl. auch Ritter (1995).
6 Vgl. Haarmann (1995).

3.1.4 Funktionale und strategische Vorteile von ORACLE

Auf dem Hintergrund der Anforderungen an ein modernes Datenbanksystem und der Produktbeschreibungen sprechen folgende entscheidungsrelevanten Aspekte für einen Einsatz der Datenbanktechnologie von ORACLE:

Funktionale Vorteile:

- Integrierte Textretrievalfunktionalität (ORACLE CONTEXT):
  Neben dem ansonsten hohen Grad an Relationalität hinsichtlich der DDL- und DML-Funktionalität bietet ORACLE als *einziger* Anbieter eine integrierte Lösung für die Behandlung von großen Textdaten in einer ORACLE-Datenbank. Dieses Konzept erlaubt nicht nur die Recherche in Textbeständen mit den gängigen Textretrievalfunktionen, sondern auch eine kombinierte Suche in strukturierten und unstrukturierten Datensätzen. Gleichzeitig werden alle Sicherheitskonzepte des Datenbanksystems für eine CONTEXT-Anwendung zur Verfügung gestellt. Eine integrierte Textretrievalfunktionalität ist insbesondere für Anforderungen interessant, die in Richtung *Information Warehouse* und Integration von Textdatenbestände gehen. Eine der wichtigsten strategischen Konzepte von ORACLE für die nächsten Versionen ist die volle Integration des Text-Servers (bisher „nur“ eine Zusatzkomponente) in den Datenbank-Kernel. Da dieses Werkzeug jedoch eine neuere Entwicklung ist, gibt es bisher wenig Erfahrungswerte insbesondere bzgl. des Performance-Verhaltens des TextServers in der Anwendung, so daß hier ganz grundsätzlich zunächst mit Schwierigkeiten gerechnet werden muß.

- Hohe Skalierbarkeit und ausführliches Tuning-Konzept:
  Die dynamische Lastprofilkonfiguration während der Laufzeit, die Ausnutzung des Cache und die parallele Query-Technologie sind bei ORACLE ausgesprochen performance-günstige Faktoren. Grundsätzlich ist Performance jedoch eine Frage der Gesamtumgebung (Hardware-Architekur des Servers und seine Bestückung, LAN-Performance, Ausstattung der Client-PCs, Performance der Client-Applikationen).

- Hohes Maß an Datensicherheit, Stabilität und Verfügbarkeit der Datenbank:
  Ausführliche Behandlung von Sicherheits- und Transaktionsmanagementproblemen, auch in einer verteilten Umgebung. Schema-Modifikationen und Sicherheitsmaßnahmen (Backup, Recovery) können im laufenden Be-

trieb vorgenommen werden. ORACLE-Datenbanken laufen laut Anwenderangaben äußerst stabil.

- Mächtige Frontend-Tools und Applikationswerkzeuge, integrierte Anwendungsentwicklung von CASE bis Reporting:
  Für alle relevanten Bereiche der Benutzer-Interaktion mit der Datenbank (Browsing, Statistiken etc.) und der Entwicklung von Datenbankapplikationen (Oberflächen, Reports, Graphiken) werden von ORACLE leistungsfähige graphische Werkzeuge angeboten. Die Applikationswerkzeuge unterstützen eine deklarative Anwendungsentwicklung, so daß Standardprobleme einfacher bis mittlerer Komplexität ohne Programmieraufwand realisierbar sind. Ein besonderer Vorzug der Produktpalette von ORACLE ist, daß die Tools auf eine einfache Integration mit anderen ORACLE Werkzeugen, z.T. auch mit Drittanbieter-Produkten, ausgerichtet sind. Auf diese Weise können vollständig integrierte Lösungen auf der Basis von CASE-, Forms-, Reports- und Graphics-Anwendungen entwickelt werden. Darüber hinaus wird sowohl auf der Ebene der Datenbank als auch in allen Bereichen der Applikationsentwicklung (Forms, Reports etc.) die gleiche Programmiersprache benutzt (PL/SQL). Damit verringert sich nicht nur der Einarbeitungsaufwand für die Entwickler, es wird auch der Austausch von PL/SQL-Modulen zwischen den Anwendungen sowie zwischen Client und Server unterstützt. Alle Oracle-Produkte sind auf einer Vielzahl von Plattformen verfügbar und unterstützen den Zugriff auf Fremddatenbanken über ODBC oder Gateways und den Datenaustausch mit anderen Windows-Programmen (Excel etc.).
- Verteilte Datenhaltung, ausgezeichnetes Replikationskonzept
- Weiterentwicklung in Richtung Objektorientierung
- Workflow-Unterstützung: Integration mit Lotus Notes (geplant)
- Internet-Anbindung.

Strategische Vorteile:

- ORACLE ist der Marktführer und bietet daher am meisten Zukunfts- und Investitionssicherheit. ORACLE ist deshalb nicht nur ein Trendsetter auf dem DBMS-Markt (s. TextServer), sondern auch im Bereich SQL-Standardisierung. Hinsichtlich Plattformen-Unabhängigkeit (s. optimale Abstimmung der Release-Stände für UNIX und DBMS) und Portabilität der Anwendungen kommt ORACLE daher eine strategische Bedeutung zu.

- *Ein* Anbieter für DBMS, Textretrieval und Frontend-Entwicklung: Müssen dagegen Werkzeuge unterschiedlicher Hersteller benutzt werden, besteht die Gefahr, daß Probleme von einem auf den anderen geschoben werden.

3.2 Textretrievalsysteme

3.2.1 Oracle Context

3.2.1.1 Systemarchitektur und Datenbankintegration

Als einziger Hersteller bietet ORACLE eine integrierte Textretrievalkomponente an (ORACLE Context), mit der eine ORACLE-Datenbank um die Fähigkeit zur Volltextrecherche erweitert werden kann. Oracle-Context ermöglicht die Speicherung beliebig großer Dokumente, die sowohl im ursprünglichen Dateisystem als auch innerhalb der Datenbank (Datentyp *long*) abgelegt sein können. Dabei können bestehende Tabellen strukturierter Daten nachträglich durch Textspalten ergänzt werden. Umgekehrt können zu einer Texttabelle strukturierte Datenfelder hinzugefügt werden. Da auf diese Weise alle Informationen (Dokumente, Daten, Indizes, Thesauri) in Tabellen einer Datenbank gespeichert werden, können auch alle Verwaltungs- und Sicherheitsfunktionen des DBMS (Backup, Recovery, Zugriffsschutz, Lesekonsistenz, Transaktionsschutz bei gleichzeitigen Zugriff im Mehrbenutzerbetrieb etc.) genutzt werden.

Der Oracle-Context erlaubt eine kombinierte SQL-Abfrage in strukturierten Datensätzen und unstrukturierten Textdatenbeständen (contains-Operator). Dadurch können Dokumente nicht nur über Spaltenwerte einzelner Tabellen (z.B. Autor, Erscheinungsjahr etc.) gefunden werden, sondern auch über freie Textpassagen in großen Textdokumenten. Der Zugriff erfolgt über einen Wort- und einen Bitmap-Index (Location List), der für alle Dokumente in der Datenbank angelegt wird. In einem Bitstring der Location List können bis zu 500.000 Dokumente referenziert werden, d.h das Vorkommen eines Wortes kann mit einer einzigen I/O-Operation in bis zu 500.000 Dokumenten festgestellt werden. Bei mehr Dokumenten müßte für das betreffende Wort ein neuer Bitstring, d.h. eine neue Zeile in der Location List angefangen werden. Ein weiterer Index ermöglicht die Abspeicherung zusätzlicher Information (Position eines Wortes innerhalb eines Dokuments, Häufigkeit des Wortes).

Zur Partitionierung der Indices können zusätzlich Virtuelle Texttabellen angelegt werden, die beliebig viele, ggf. auch verteilt gehaltene Texttabellen sowie weitere Virtuelle Texttabellen umfassen können. Auf diese Weise ent-

steht eine baumartige Struktur, die bei einer Suche rekursiv durchlaufen wird. Dadurch können nicht nur große Textdatenbestände mit einer einzigen Abfrage erfaßt werden. Es können Abfragen auf Textdatenbestände auch parallelisiert werden, so daß schnelle Antwortzeiten auch bei großen Datenmengen möglich sind. Die Technologie der Virtuellen Texttabellen bietet sich an, wenn mehr als 500.000 Dokumente indiziert werden müssen, so daß für ein Wort in der Location List nicht mehrere Bitstrings abgefragt werden müssen. Durch eine Verteilung auf mehrere Texttabellen, auf die über eine Virtuelle Texttabelle zugegriffen wird, wird so gewährleistet, daß es für jedes Wort pro Bitmap Index genau einen Bitstring gibt. Fraglich ist allerdings, ob bei einem Einsatz Virtueller Texttabellen auch komplexe Join-Operationen mit strukturierten Datensätzen performant abgearbeitet werden.

Linguistische Komponenten werden in erster Linie nur für das Englische angeboten. Für die deutsche Sprache wird nur die Stammformenreduktion unterstützt.

3.2.1.2 Suchmöglichkeiten und Ranking

Die Suchmöglichkeiten des Oracle-Textservers umfassen alle gängigen Textretrievalfunktionen:

- Suchbar sind einzelne Begriffe, zusammengesetzte Wörter oder Satzteile
- Verwendung von Platzhaltern, Rechts- und Linkstrunkierung, Abstandssuche; alle Möglichkeiten können zusammen verwendet werden.
- Durch die Integration in SQL (Boolesche Logik) können auch komplexe, geschachtelte Abfragen verarbeitet werden (*contains*-Operator).
- Abfragen können gespeichert und wiederverwendet werden.
- Thesaurus-Funktionalität
- Erzeugung einer Trefferliste.

Der Text wird in einem Textfenster im ASCII-Format dargestellt; die gesuchten Textstellen werden dabei hervorgehoben. Das Textverarbeitungssystem, mit dem das Dokument erstellt wurde, kann von ORACLE aus über einen Formatmanager aufgerufen werden, so daß das Dokument im Originalformat bearbeitet werden kann. Eine Aktualisierung des Dokuments und die Indizierung kann direkt online in der Datenbank oder später im Batch-Modus nachvollzogen werden. Eigene Formatmanager sind über eine offene Schnittstelle

(API) in den Textserver integrierbar. Dialoganwendungen mit dem Textserver können in Forms- oder Pro*C-Applikationen eingebunden werden.

Oracle-Context stellt ein einfaches Ranking-Verfahren zur Verfügung, das im wesentlichen auf dem im Information Retrieval üblichen Standardmaß der inversen Dokumenthäufigkeit beruht, d.h. Häufigkeit der Suchterme im Dokument in Relation zu der Häufigkeit derselben in der gesamten Kollektion, wobei die Terme nach Maßgabe ihrer Häufigkeit in der Anfrage und in der Kollektion gewichtet werden. Ein elaboriertes Retrievalmodell (wie das Vektorraummodell oder andere probabilistische Retrievalkonzepte) stellt Context allerdings nicht zur Verfügung. Auch ist es nicht möglich, zwischen mehreren Verfahren zu wählen oder auf das Ranking durch entsprechende Parametrisierung Einfluß zu nehmen.

3.2.2 Fulcrum SearchServer

Am Informationszentrum Sozialwissenschaften soll ein vorhandenes Volltextretrievalsystem (Fulcrum SearchServer 3.7) als technische Grundlage für die automatische Indexierung von großen Textkorpora und für die Suche in diesen Datenbeständen eingesetzt werden. Das System wurde auf der Grundlage der technischen Dokumentation, einer Test-Installation, eines zweitätigen Consulting-Seminars und ausführlichen Retrieval-Tests evaluiert. Bewertungskriterien waren, neben technischen Daten und Systemgrenzen, insbesondere die Unterstützung von Booleschem und statistischem Retrieval, die verfügbaren Text-operationen und Ranking-Methoden beim Retrieval, die Unterstützung von Thesauri und Stoppwortlisten, die Möglichkeiten der Integration in bestehende Datenbankumgebungen (Oracle), die Mächtigkeit der linguistischen Komponenten und das API.

3.2.2.1 Systemarchitektur, Datenmanagement und Indexierung

Fulcrum SearchServer ist ein Client-Server-basiertes Volltextdatenbanksystem, das als zentrales Speicherungsprinzip ein relationales Datenmodell in Form von Tabellen verwendet. Eine Zeile repräsentiert dabei ein Textobjekt bzw. Dokument und jede Spalte ein einzelnes suchbares Attribut eines Textobjektes, z.B. Titel des Dokuments oder Abstract. Das bedeutet nicht nur, daß neben den eigentlichen Volltexten auch strukturierte Daten gespeichert werden können, sondern das auch eine kombinierte Suche in strukturierten und unstrukturierten Datensätzen möglich ist.

Mit dem *Fulcrum Surfboard* steht ein Internet-Gateway zur Verfügung, mit dem sowohl WWW-Browser als auch Z39.50-Clients auf die Datenbank zugreifen können.

Als Datenmanipulations- und Abfragesprache fungiert *SearchSQL*. SearchSQL ist eine Untermenge von SQL, der Standardabfragesprache für relationale Datenbanken, stellt aber auch eine Reihe von Fulcrum-spezifischen Erweiterungen (v.a. für die Suche) zur Verfügung. SearchSQL unterstützt allerdings keine Join-Operationen, so daß komplexere Abfragen über mehrere Tabellen in einem *select*-Statement nicht möglich sind. Eine Abfrage über mehrere Tabellen ist nur durch Definition einer View bzw. Anlegen eines speziellen View-Files möglich.

Bis auf Gleitkommazahlen werden alle relevanten Datentypen unterstützt:

- *char*: alphanumerische Zeichenkette mit einer festen Länge von maximal 32767 Zeichen
- *varchar*: alphanumerische Zeichenkette mit einer variablen Länge von maximal 32767 Zeichen
- *apvarchar*: externes Textdokument mit einer maximalen Länge von 2 GB
- *integer*, *smallint*: numerische Daten (32 bzw. 16 Bit)
- *date*: Datumsangaben im Format YYYY-MM-DD.

Jeder Datentyp hat einen default-Indexierungsmodus, der den Indexierungs- und Suchmodus für die Spalte festlegt:

- *normal* (bei Textspalten): ermöglicht die Suche nach Wörtern oder Phrasen; der Index enthält einen Eintrag für jedes Wort in der Spalte;
- *literal* (bei alphanumerischen Daten): der Index enthält einen Eintrag für jede durch Hochkommata getrennte Sequenz von Zeichen;
- *value* (bei numerischen Werten): ermöglicht die Anwendung von mathematischen Vergleichsoperatoren; der Index enthält einen Eintrag für jedes Datum oder jeden numerischen Wert in der Spalte; Spalten des Datentyps *date*, *integer* und *smallint* müssen mit dem *value*-Modus definiert werden.

Der Benutzer hat auf der Basis dieser Datentypen und Indexierungsmodi die Möglichkeit, eigene Datentypen (*domains*) zu definieren.

Für jede Tabelle können spezifische Systemparameter definiert werden, die das Datenmanagement für die Tabelle beschreiben. Die beiden wichtigsten sind die Indexierungsparameter *immediate* und *periodic*. Während der *immediate*-Parameter dafür sorgt, daß Daten, die geändert oder neu eingefügt werden, sofort indexiert und damit suchbar gemacht werden, sind bei dem *periodic*-Parameter die Daten erst nach Ausführung eines *validate index* verfügbar.

3.2.2.2 Behandlung von Textdokumenten

Volltexte können sowohl in der Datenbank als auch im Dateisystem abgelegt werden. Dokumente können im laufenden Betrieb eingefügt und re-indexiert werden. Eine Volltextsuche ist sowohl in den Textspalten einer Tabelle als auch in externen Dokumenten möglich.

Auf externe Textdokumente wird über sog. *text reader* zugegriffen, die das externe Dokumentformat in das *Fulcrum Technologies Document Format* (FTDF) konvertieren. Diese Konvertierung erfolgt dynamisch (*on demand*), so daß externe Dokumente im Originalformat recherchiert werden können. Durch Kombination mehrerer Text-Reader (*text reader list*) können auch komplexere Formatkonvertierungen durchgeführt werden.

Mehrere externe Dokumente können zu einem *document library file* logisch zusammengefaßt werden. Dies ermöglicht eine individuelle Indexierung von Bibliotheksdokumenten. Der Zugriff erfolgt dann über einen speziellen Text-Reader.

Fulcrum SearchServer unterstützt drei Klassen von Text-Readern:

- *storage access text reader*: Low-Level-Text-Reader mit direktem Zugriff auf die externe Datenquelle. Zu dieser Klasse gehören auch der *ODBC text reader* und der *library text reader*, mit dem auf Fulcrum Textobjekt-Bibliotheken zugegriffen werden kann. Ein weiteres Feature dieser Klasse ist die Zeilenexpansion bei sog. *containern*, die mehrere Textobjekte enthalten. Dieser Text-Reader ermöglicht z.B. die Expansion von Dokumentenbibliotheken und die Directory- und Email-Expansion.

- *format translation text reader*: Konvertierer vom Originalformat nach FTDF; unterstützt werden alle gängigen Textverarbeitungsprogramme (*multi-format text reader*), HTML-, Email- und Internet-News-Dokumente sowie PDF-Dateien und die Konvertierung von externen Zeichensätzen.

- *FTDF parsing text reader*: Konvertierer zum Laden von Daten in spezifische Tabellenspalten und zur dynamischen Ableitung von Textsegmenten (*zones*, s.u.). Das vom *translation text reader* generierte Format wird dabei nach entsprechenden Mustern durchsucht und dynamisch durch *zone control codes* ergänzt, wobei das Originalformat erhalten bleibt. Auf diese Weise können auch strukturierte Dokumente (z.B. SGML-Dokumente) konvertiert werden.

Über die genannten Standardmethoden hinaus ist auch die Entwicklung eigener Text-Reader mit Hilfe des Fulcrum-API möglich.

Die Text-Reader versehen die Texte mit Kontrollsequenzen, die Indexierungsmodi und Format- und Darstellungsattribute festlegen. Die Kontrollsequenzen können auch manuell in die Dokumente eingefügt werden. Zusätzlich ist es mit Hilfe sog. *character variant rules* möglich, die Behandlung von Zeichen mit Akzenten festzulegen, so daß auch Worte mit diakritischen Zeichen gefunden werden können, unabhängig davon, ob die Umlaute noch im Text erhalten sind oder bereits konvertiert wurden.

Ein ausgefeiltes Tabellenvalidierungskonzept sorgt für eine Synchronisierung der physischen Tabellendaten mit dem Status der externen Dokumente. Ändert sich ein externes Dokument oder wird ein neues in die Tabelle „eingefügt“, so ändert sich automatisch der Indexierungsstatus der Zeile. Beim nächsten *validate index* (mit der Option *validate table*) werden nur noch geänderte oder neu hinzugekommene Dokumente indexiert. Die ganze Tabelle muß nicht nochmal re-indexiert werden. Leider erfolgt bei einer Änderung externer Dokumente nicht auch ein automatischer Update des Indexes. Dieser muß vielmehr manuell angestoßen werden.

Ein besonderer Vorzug des Fulcrum SearchServers ist, daß Textspalten in sog. *zones* eingeteilt werden können. Auf diese Weise kann ein Text in einzelne Bestandteile (z.B. Titel, Abstract und Artikeltext) zerlegt werden, die einen eindeutigen Namen bekommen und auf die dann die Suche eingeschränkt werden kann. Eine *zone* kann jedoch nur zusammen mit der ganzen Textspalte eingefügt oder geändert werden.

Die Zuordnung von Textteilen zu *zones* erfolgt entweder über einen Text-Reader oder durch Einfügen von entsprechenden Kontrollsequenzen in die Texte. Eine *zone* kann sich auch aus anderen *zones* zusammensetzen (Gruppierung von *zones*), so daß in einer Query nicht alle einzelnen *zones* benannt werden müssen.

Jeder *zone* kann ein individueller Indexierungsmodus zugewiesen werden, der von dem für die ganze Spalte geltenden verschieden sein kann. Hierzu müssen in den Text entsprechende *index mode delimiter* eingetragen werden, die den jeweiligen Indexierungsmodus zu Beginn und am Ende einer *zone* an- bzw. abschalten. Die Benutzung verschiedener Indexierungsmodi innerhalb einer Spalte ist jedoch nicht zu empfehlen, da bei einer Suche über die ganze Spalte hinweg die Indexierungsmodi der einzelnen *zones* innerhalb einer Spalte nicht zugrundegelegt werden, sondern der Modus, der für die ganze Spalte definiert wurde.

Ein weiterer kritischer Punkt aus administrativer Sicht bei der Benutzung von *zones* ist, daß sich Änderungen bezüglich der *zone control codes* und der *index mode delimiter* im Data Dictionary, d.h. in der *zone*-Beschreibung des *create-schema*-Statements widerspiegeln müßen. Wurde ein Text-Reader benutzt, setzt dies eine genaue Kenntnis der Steuerungszeichen voraus, die der Reader verwendet.

3.2.2.3 Datenbankintegration

Für die Integration von relationalen Datenbanken steht ein ODBC-Text-Reader zur Verfügung. Im Rahmen der Evaluation wurden Datenbestände des IZ testweise in eine Oracle-Datenbank geladen und über den ODBC-Text-Reader in die Fulcrum-Datenbank integriert. Die Indexierung der Oracle-Tabellen über den ODBC-Reader verlief reibungslos und performant. Auch bei der Suche in den Texten traten keine technischen Probleme auf.

3.2.2.4 Suche, Rankingmethoden und Retrievalqualität

Bei Ausführung eines *select*-Statements referenziert die SearchServer-Engine das Data Dictionary, um zu bestimmen, wie die Daten, in denen gesucht werden soll, indexiert wurden. Soll in einer ganzen Spalte gesucht werden, wird der Indexierungsmodus der Spalte zugrunde gelegt, nicht der der einzelnen *zones* innerhalb einer Spalte. Dies muß bei der Verwendung verschiedener Indexierungsmodi innerhalb einer Spalte berücksichtigt werden.

Fulcrum Searchserver stellt linguistische Komponenten zur Verfügung, mit deren Hilfe verschiedene Wortformen aus den Suchtermen generiert und in die Suche miteinbezogen werden können. Unterstützt werden Stammformenreduktion, Kompositazerlegung (Deutsch, Schwedisch), spelling variants (Englisch, Deutsch), character variants (Deutsch, Französisch) sowie die Behandlung von Sonderzeichen (Bindestriche u.ä.).

In die Suche kann darüber hinaus ein Thesaurus miteinbezogen werden, der für jede Datenbanktabelle angelegt werden kann. Der Thesaurus kann nicht nur Begriffsrelationen, sondern auch Regeln enthalten, die Wortvarianten erzeugen (z.B. konjugierte oder deklinierte Formen). Ferner wird die Eliminierung von Stoppwörtern unterstützt.

Die Darstellung der Ergebnis-Dokumente ist im ASCII-Format, im Originalformat des Dokumentes sowie, bei Verwendung des Surfboard, in HTML möglich. Bei der ASCII- und HTML-Darstellung erfolgt ein Highlighting der Suchbegriffe.

Für die Suche werden drei verschiedene Retrievalmodelle angeboten:

- *strict Boolean*: exact-match-Retrieval mit scharfer Interpretation von Boole'schen Operatoren
- *fuzzy Boolean*: vage Interpretation von Boole'schen Operatoren
- *vector space*: statistisches Modell, welches die Ähnlichkeit zwischen Anfrage(vektor) und Dokument(vektor) berechnet

Das Retrievalmodell kann sowohl global als auch im *select*-Statement spezifiziert werden. Jedes Retrievalmodell kann mit verschiedenen Ranking-Methoden verknüpft werden. Dabei werden folgende Möglichkeiten unterstützt:

1. *hits count*: Häufigkeit der Suchbegriffe im Dokument, unabhängig von der Häufigkeit der Terme in der gesamten Kollektion (funktioniert nicht in Kombination mit *fuzzy Boolean* und *vector space*)
2. *terms count*: Anzahl der gefundenen Suchbegriffe pro Dokument, unabhängig von der Häufigkeit der Terme im Dokument (funktioniert nicht in Kombination mit *fuzzy Boolean* und *vector space*)

3. *terms ordered*: Häufigkeit der Terme im Dokument in Relation zu der Häufigkeit derselben in der gesamten Kollektion (inverse Dokumenthäufigkeit)

4. *critical terms ordered*: wie *terms ordered*, legt jedoch größeres Gewicht auf Terme, die in der Kollektion seltener vorkommen.

Der Benutzer kann auf die Ranking-Algorithmen Einfluß nehmen, indem er die Suchterme gewichtet (optional).

Um die Qualität des Textretrievals beurteilen zu können, wurde auf eine Testkollektion von ca. 13.000 SOLIS- und FORIS-Dokumenten zurückgegriffen, die bereits die Grundlage für den am IZ im Rahmen des GIRT-Projektes[7] durchgeführten Retrievaltest war. Zu Vergleichszwecken wurden alle neun Anfragen aus dem GIRT-Test sowohl an die Fulcrum-Datenbank als auch an ein methodisch vergleichbares statistisches Volltextretrievalsystem (freeWAIS-sf), das im Kontext der Information-Retrieval-Forschung entwickelt worden ist, gestellt. Es wurde als Retrievalmodell das Vektorraummodell in Kombination mit *critical terms ordered* gewählt (die von Fulcrum empfohlene Methode); freeWAIS-sf ist ebenfalls ein Vektorraum-Retrievalmodell, basiert jedoch auf der inversen Dokumenthäufigkeit.

Die Retrievalqualität der Systeme wurde nach den im Information Retrieval üblichen Maßen Precision und Recall als Mittelwert über alle neun Anfragen gemessen. *Precision* ist ein Maß für die Fähigkeit des Systems, die für eine Anfrage relevanten Dokumente von den nicht-relevanten zu trennen und somit nur relevante Dokumente nachzuweisen (je näher der Wert an 1 ist, desto weniger irrelevante Dokumente werden geliefert). *Recall* gibt an, wie viele der relevanten Dokumente das System zurückliefert (je näher der Wert an 1 ist, desto mehr relevante Dokumente werden geliefert). Für die Ermittlung der Recall- und Precision-Werte konnte auf intellektuelle Relevanzbewertungen, die im Rahmen des GIRT-Retrievaltests durchgeführt wurden, zurückgegriffen werden.

---

[7] Im GIRT-Projekt (German Indexing and Retrieval Testdatabase) werden existierende bzw. in der Entwicklung befindliche Indexierungs- und Retrievalsysteme auf ihre Leistungs- und Einsatzfähigkeit für den Bereich der Fachinformation überprüft. Vgl. dazu Frisch/Kluck (1997)

Dabei ergab sich nach 10 bzw. 30 Dokumenten folgendes Bild:

	Fulcrum SearchServer 3.7		freeWAIS-sf 2.0	
	Precision	Recall	Precision	Recall
10	0,41	0,08	0,32	0,06
30	0,27	0,17	0,26	0,14

Das Ergebnis weist bezüglich der Precision einen deutlichen Vorteil des Fulcrum SearchServers gegenüber freeWAIS-sf im oberen Bereich der Ranglisten (Gruppe der ersten zehn Ergebnis-Dokumente) aus. Die anderen Werte sind nicht signifikant, zeigen aber, das sich der SearchServer mit (anerkannten) Konkurrenzprodukten messen lassen kann.

Beim Fulcrum SearchServer wurden stichprobenweise auch die anderen Retrievalmodelle und Ranking-Methoden getestet. Dies führte jedoch nicht bzw. nur in Einzelfällen (bei *terms count*) zu einer Verbesserung der Retrievalergebnisse.

4 Fazit

Der Fulcrum SearchServer ist ein ausgefeiltes und stabiles Volltextretrievalsystem, das eine Vielzahl von Dokumentformaten unterstützt und umfangreiche Konfigurationsmöglichkeiten bietet. Sein relationales Datenmodell ist gut dazu eignet, strukturierte Informationen zu verwalten und erlaubt eine kombinierte Suche in strukturierten und unstrukturierten Datensätzen. Komplexere Abfragen über mehrere Tabellen sind jedoch nicht möglich. Für die Administration einer Fulcrum-Datenbank steht ein komfortables User-Interace zur Verfügung.

Eine Fulcrum-Datenbank läßt sich reibungslos in eine bestehende relationale Datenbank-Umgebung (z.B. Oracle) integrieren.

Das System stellt dem Benutzer eine Reihe mächtiger Retrievalkonzepte und Ranking-Methoden zur Verfügung. Die Retrievalqualität kann mit der vergleichbarer (anerkannter) Systeme konkurrieren oder ist sogar noch besser. Die Qualität der Ergebnisse konnte mit den Texten der GIRT-Initiative getestet werden. Hierzu liegen intellektuelle Relevanzbewertungen und Tests mit

anderen Indexierungssystemen vor. Für Fulcrum ergaben sich vergleichbare Ergebnisse (intellektuelle Parallelisierung). Fulcrum SearchServer eignet sich somit grundsätzlich als Standard-Werkzeug für die automatische Indexierung.

Für die Entwicklung eigener Client-Applikationen wird ein ausführliches API angeboten, das alle verbreiteten Betriebssystemplattformen unterstützt. Für die Fulcrum-Anbindung in bestehende Informationssysteme wurde mit dem Fulcrum-API bzw. einer darauf aufsetzenden C++-DLL ein multi-threaded NamedPipe-Server implementiert, der bis 255 parallele Transaktionen gegen die Fulcrum-Datenbank unterstützt.

Eine Anbindung an die Windows-Entwicklungsumgebungen, wie z.B. PowerBuilder, ist über ODBC gegeben.

5 Literatur

BAUER, M. (1995a): Verkapselt und vererbt: Objektorientierte Datenbankkonzepte. *online 3, 58-65.*

Belkin, N.J. (1996): Intelligent Information Retrieval: Whose Intelligence? In: Krause, J.; Herfurth, M.; Marx, J. (Hrsg.): Herausforderungen an die Informationswirtschaft, Informationsverdichtung, Informationsbewertung und Datenvisualisierung. Proceedings des 5. Internationalen Symposiums für Informationswissenschaft (ISI'96), Konstanz, S. 25-31.

Belkin, N.J; Cool, C.; Croft, W.B.; Callan, J.P. (1993): The Effect of Multiple Query Representation on Information Retrieval System Performance: In: Korfhage, R.; Rasmussen, E.; Willett, P. (Hrsg.): Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Pittsburg, S. 339-346.

Blair, D. C.; Maron, M. E. (1985): An evaluation of retrieval effectiveness for a full-text document-retrieval system. Commun. ACM 28, 3, S. 289-299.

Dahlberg; I. (1996): Replik zur Kritik des Hauptartikels „Ingetraut Dahlberg: Zur „Begriffskultur" in den Sozialwissenschaften: Lassen sich ihre Probleme lösen?". EuS 7; H. 1, S. 82.

DRUCKS, H.J. (1992): Die Zukunft liegt jenseits des Coddschen Datenmodells. *Computerwoche 34, 31-34.*

Endres-Niggemeyer, B. (1992): Abstrahieren, Indexieren und Klassieren. Ein empirisches Prozeßmodell der Dokumentrepräsentation. Habilitationsschrift, Universität Konstanz. Informationswissenschaft.

Fox, E.A.; Shaw, J.A. (1994): Combination for Multiple Searches. In: Harman, D.K. (Hrsg.): The Second Text Retrieval Conference (TREC-2). NIST Special Publication 500-215, S. 243-252.

Frisch, E.; Kluck, M. (1997): Pretest zum Projekt German Indexing and Retrieval Testdatabase (GIRT) unter Anwendung der Retrievalsysteme Messenger und freeWAISsf. Bonn 1997 (IZ-Arbeitsbericht; Nr. 10).

Fuhr; N; Müller, P. (1987): Probabilistic Search Term Weighting - Some Negative Results. In: Van Rijsbergen. C.J.; Yu, C.T. (Hrsg.): Proceedings of the Tenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, S. 13-18.

GERKENS, R. (1994): Migrationskonzepte von IBM, CA und System Performance. Wie sich der Übergang in die relationale Welt gestalten läßt. *Computerwoche 31, 29-31.*

HAARMANN, G. (1995): Vom Datenkunden zum Informationskönig. Durch Data-/Information-Warehousing sollen aus Daten entscheidungsrelevante Informationen werden. *FOCUS 2, 24-27.*

Hersh, W. R. (1995): Information Retrieval: A Health Care Perspective, Springer.

Ingwersen, P. (1996): The Cognitive Framework for Information Retrieval: A Paradigmatic Perspective. In: Krause, J.; Herfurth, M.; Marx, J. (Hrsg.): Herausforderungen an die Informationswirtschaft, Informationsverdichtung, Informationsbewertung und Datenvisualisierung. Proceedings des 5. Internationalen Symposiums für Informationswissenschaft (ISI'96), Konstanz, S. 65-78.

Krause, Jürgen (1996a): Informationserschließung und -bereitstellung zwischen Deregulation, Kommerzialisierung und weltweiter Vernetzung („Schalenmodell"). Bonn 1996 (IZ-Arbeitsbericht; Nr. 6).

Krause, J. (1996b): Principles of Content Analysis for Information Retrieval Systems. An Overview. In: Zuell, C.; Harkness, J.; Hoffmeyer-Zlotnik, J. (Eds.): Text Analysis and Computer. ZUMA-Nachrichten Spezial, S. 77-104.

Krause, J.; Zimmer, M. (Hrsg.) (1996): Informationsservice des IZ Sozialwissenschaften. Datenbankentwicklung und -nutzung, Netzwerke, Wissenschaftsforschung. Bonn.

Lee, W.C.; Fox, E.A. (1988): Experimental Comparison of Schemes for Interpreting Boolean Queries. Virginia Tech M.S. Thesis, Technical Report TR-88-27. Department of Computer Science.

Möller, Tong (1993): Juris für Juristen. (Law for Jurists). Dissertation - Universität des Saarlandes.

Noreault, T.; Koll, M.; McGill, M.J. (1977): Automatic Ranked Output from Boolean Search in SIRE. Journal of the American Society for Information Science, November, S. 333-339.

RITTER, U. (1995): Die traditionelle Aufgabe wird sich grundlegend ändern. Künftige Datenbanken erfordern die Kombination von strukturierten und unstrukturierten Daten. *FOCUS 2, 32-33.*

Salton, G. (1986): Another look at automatic text-retrieval systems. Communications of the ACM, July, 29, 7, S. 648-656.

SAUER, H. (1992): Relationale Datenbanken. Theorie und Praxis. *Addison-Wesley. Bonn-München-Paris.*

SCHLAGETER, G. und STUCKY, W. (1987): Datenbanksysteme: Konzepte und Modelle. *Teubner Studienbücher.*

SCHMATZ, K.-D. und WEIKERT, P. (1995): Die Anforderungen der Anwender berücksichtigen. Auswahl und Bewertung von objektorientierten Datenbanksystemen. *FOCUS 2, 19-21.*

Swanson, D.R. (1988): Historical note: Information retrieval and the future of an illusion. Journal of the American Society for Information Science, 39, S. 92-98.

Tague-Sutcliffe, J; Blustein, J. (1995): A Statistical Analysis of the TREC-3 Data. In: Harman, D.K. (Hrsg.): Overview of the Third Text Retrieval Conference (TREC-3), NIST Special Publication 500-225, S. 385-398.

Ulisch, C. (1997): Integration eines aktiven Informationssystems für das Internet in den Kontext eines Fachinformationsdienstleisters. Diplomarbeit Universität Koblenz, Institut für Informatik.

ULLMANN, J.D. (1988): Principles of Databases and Knowledge-Base Systems. Volume I. *Computer Science Press.*

VORWERK, R. (1995): In der Praxis gibt es auch Grautöne. Relationale Datenbanken eignen sich durchaus zur Speicherung von Objekten. *FOCUS 2, 22-23.*

Wilkinson, R. (1994): Effective Retrieval of Structured Documents. In: Croft, W.B.; Van Rijsbergen, C.J. (Hrsg.): Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. London et al., S. 311-317.

Wolf, Gerhard (1992): JURIS - Ein denkbarer einfacher Zugang zu allen Informationen, die Sie brauchen. jur-pc. 4. S. 1524-1810.

Womser-Hacker, Ch. (1996): Das MIMOR-Modell. Mehrfachindexierung zur dynamischen Methoden-Objekt-Relationierung im Information Retrieval. Habilitationsschrift, Universität Regensburg. Informationswissenschaft.