Korpuslinguistik und interdisziplinäre Perspektiven auf Sprache

Band 8

Roman Schneider

Mehrfach annotierte Textkorpora

Strukturierte Speicherung und Abfrage

narr\f
ranck
e\atte
mpto

CLIP 8

Korpuslinguistik und interdisziplinäre Perspektiven auf Sprache

Corpus Linguistics and Interdisciplinary Perspectives on Language

Bd. / Vol. 8

Die Bände der Reihe werden in einem Peer-review-Verfahren geprüft / CLIP is a peer reviewed series.

Roman Schneider

Mehrfach annotierte Textkorpora

Strukturierte Speicherung
und Abfrage

narr\f
ranck
e\atte
mpto

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de abrufbar.

Internet: www.narr.de
E-Mail: info@narr.de

Redaktion: Norbert Cußler-Volz, Mannheim
Layout: Andy Scholz, Essen (www.andyscholz.com) / Joachim Hohwieler, Mannheim
Druck: CPI books GmbH, Leck

ISSN 2191-9577
ISBN 978-3-8233-8286-7

Inhalt

1. Einführung und Motivation

Die zunehmende Verfügbarkeit digitaler Archive natürlicher Sprache hat die Voraussetzungen, unter denen sich Wissenschaftler im interdisziplinären Spannungsfeld zwischen Sprachwissenschaft, Computerlinguistik und Informatik[1] mit der Erforschung von Sprachphänomenen beschäftigen, in den letzten Jahren fundamental verändert. Gelegentlich wird gar von einem damit einhergehenden Paradigmenwechsel in der Linguistik gesprochen: Weg von einer mutmaßlich theoretisch-spekulativen Arbeitsweise, bei der Introspektion und Intuition des Forschers als primäre Mittel zur Erhebung sprachlicher Daten dienen, hin zu einer weitestgehend empirischen Ausrichtung, bei der umfangreiche Sammlungen geschriebener und – aus vielerlei methodologischen und technologischen Gründen derzeit noch in weitaus geringerem Ausmaß – gesprochener Sprache die Basis für die mathematisch präzise Formlierung von Generalisierungen über den zu beschreibenden Wirklichkeitsausschnitt bilden.

Allerdings verliert dieser scheinbare Antagonismus zwischen Theorie und Empirie bzw. zwischen qualitativer und quantitativer Sprachforschung bei näherer Betrachtung viel von seiner Schärfe. Ungeachtet des nachvollziehbaren Bestrebens neuer Paradigmen, vormals etablierte Denkschemata und Arbeitsweisen als fürderhin unhaltbar oder zumindest erkenntnishemmend darzustellen, schließen sich quantitative Untersuchungen und qualitative Schlussfolgerungen tatsächlich keineswegs aus. Stattdessen ist beim Streben nach wissenschaftlicher Erkenntnis ein sich ergänzendes Wechselspiel zu beobachten: Die empirische Exploration umfangreicher Sprachartefakte gibt uns einerseits Mittel zur Aufdeckung inhärenter Regularitäten und damit zur Erstellung von Hypothesen an die Hand, die im Idealfall zur Formulierung umfassender Gesetze und Modelle beitragen. Existierende Theorien lassen sich andererseits quantitativ überprüfen und gegebenenfalls modifizieren oder verwerfen. Kurz: Gleichgültig, ob der konkreten Beobachtung eine (rational begründete) Theorie vorangeht oder umgekehrt – während irgendeines Arbeitsschritts in der wissenschaftlichen Untersuchungssystematik werden Aussagen über systemisch-strukturelle Spracheigenschaften stets mit der empirischen Analyse von Sprachwirklichkeit konfrontiert.

---

[1] In diesem Spannungsfeld wirken selbstverständlich punktuell weitere Wissenschaftsbereiche mit, z.B. die Medien- und Kommunikationswissenschaft, Psychologie, Neurologie und Soziologie, um nur einige der qua Untersuchungsgegenstand bzw. Erkenntnisinteresse potenziell involvierten Disziplinen zu nennen.

Genau an diesem Punkt lassen sich maschinell aufbereitete Sprachkorpora durch nichts ersetzen, wenn man darauf Wert legt, linguistische Analysen an authentischem Sprachmaterial durchzuführen und grundsätzlich generalisierbar (Kriterium der Validität) sowie reproduzierbar (Kriterium der Reliabilität) anzulegen. Das heranzuziehende Datenmaterial – in Heyer et al. (2008) wird hierfür das Bild vom „Wissensrohstoff Text" geprägt – ist tendenziell hochkomplex. Um den an sie gestellten Ansprüchen gerecht zu werden, bestehen moderne sprachwissenschaftlich motivierte Korpora nämlich nicht allein aus den natürlichsprachlichen Primärdaten (Rohtexten), sondern ganz wesentlich zusätzlich aus linguistischen Beschreibungen, d.h. morphologischen, morphosyntaktischen, phonetischen, prosodischen oder sonstigen Klassifikationen. Diese werden gemeinhin in Form sogenannter Mehrebenen-Annotationen (engl. *multi-layer annotations* bzw. *multi-level annotations*) kodiert und angesichts der mittlerweile immensen Größe forschungsrelevanter Digitalarchive zumeist mit Hilfe maschineller Parser und Tagger generiert. Für die vergleichende Auswertung und zum Erkennen problematischer Etikettierungen erscheint dabei sogar in vielen Fällen der parallele Einsatz mehrerer solcher Werkzeuge angeraten.

Ergänzend zu primär wortbasierten[2] Annotationen erfordern linguistische Untersuchungen häufig einen Rückgriff auf übergeordnete syntaktische Strukturinformationen (Wortgruppen-, Phrasen- bzw. Abhängigkeitsstrukturen, Satzgrenzen etc.), wie sie in spezialisierten Baumbanken (engl. *tree banks*)[3] vorliegen. Für manche Fragestellungen sind auch semantische[4] (z.B. thematische Rahmen, Agens vs. Patiens) oder pragmatische[5] Annotationen (z.B. Koreferenzen oder Diskursstrukturen) interessant. Forschungen zum Zweit-

[2] Neben der Wortform (Token) beziehen sich linguistische Annotationen und Abfragen gelegentlich auch auf andere Basissegmente, z.B. Morpheme, Silben, Phoneme etc.

[3] Prominente Vertreter sind die Penn Treebank (Marcus et al. 1993) für das Englische sowie in Deutschland die Tübinger Baumbanken (Telljohann et al. 2009), NEGRA (Brants et al. 1999) und in dessen Nachfolge TIGER (Brants et al. 2002); einen vergleichenden Überblick bietet z.B. Kübler (2010). Für korpuslinguistisch Interessierte mit eigenen Primärtexten, aber ohne angemessene technische Infrastruktur und computerlinguistisches Know-how, stellt das Projekt „Treebank.info" (http://treebank.info) unter Nutzung eines Hochleistungsrechners des Regionalen Rechenzentrums Erlangen eine intuitiv bedienbare grafische Web-Oberfläche für das syntaktische Parsing (unter Nutzung des Stanford-Parsers; http://nlp.stanford.edu/software/lex-parser.shtml) und Retrieval zur Verfügung.

[4] Siehe z.B. die Arbeiten im Saarbrücken Lexical Semantics Acquisition Project (Burchardt et al. 2009) oder die Dokumentation der *Groningen Meaning Bank* (http://gmb.let.rug.nl) in Bos et al. (2017).

[5] Beispiele hierfür sind die RST Discourse Treebank (Carlson 2002), die Bangla RST Discourse Treebank (Das/Stede 2018) oder das Potsdam Commentary Corpus PCC (Stede 2004, (Hg.) 2016).

bzw. Fremdsprachenerwerb nutzen fehlerannotierte Lernerkorpora; Untersuchungen etwa zur Gebärdensprache verwenden multimodale Text-/Ton-/Video-Korpora mit Annotationen zu Gestik, Mimik etc. Hinzu kommen im Idealfall jeweils korpus- oder textspezifische Metadaten wie Textsorte, Medium, Domäne, Publikationsdatum und -ort sowie soziologisch-demografische Parameter wie Wohn-/Geburtsort, Geschlecht oder Alter der Autoren etc., die für die multidimensionale Einbeziehung außersprachlicher Faktoren in linguistische Analysen unverzichtbar sind:[6]

> Erst die Kopplung der ausdrucksseitig bestimmten Reichweiten-Varietätentypen (Dialekte, Regiolekte und Standardlekte) mit semantisch bestimmten qualitativen Funktions-Varietätentypen (also Semantiktypen des Alltags oder verschiedener Fachdisziplinen usw.) ermöglicht die angemessene Charakterisierung von Erscheinungsformen des Deutschen." (Felder/Gardt 2014, S. 23)

Als Konsequenz beinhalten linguistische Korpussammlungen gleichermaßen unstrukturierte, semi-strukturierte und strukturierte Daten (vgl. hierzu auch Lobin/Lemnitzer 2004 bzw. Lobin 2009):

- Unstrukturierte Daten: In diese Kategorie fallen textuelle Primärdaten, die im Originalzustand abgesehen von der simplen horizontalen Verkettung von Einzelwörtern keinerlei explizit kodierten Relationen aufweisen.
- Semi-strukturierte Daten: Hierunter werden gemeinhin um Annotationen angereicherte Texte verstanden. Die Datenbasis wird dabei um strukturelle Klassifikationen angereichert; allerdings lassen sich diese Informationen ohne explizite Aufbereitung (insbesondere Indizierung) nur begrenzt präzise und dokumentübergreifend abfragen.
- Strukturierte Daten: Im Idealfall der umfassenden Strukturierung liegen zusätzlich zur Annotation auch korpus- und textspezifische Metadaten in Form indizierter Listen oder Tabellen vor. Eindeutige Schlüssel-Wert-Beziehungen erlauben dann die gezielte Recherche nach sämtlichen kodierten Merkmalsausprägungen.

Aus der Perspektive derjenigen, die für das Retrieval – d.h. für die Konzeption und Implementierung entsprechender Korpusabfragen – zuständig sind, kann sich die Definition verbindlicher Abfrageschlüssel an dieser Stelle rasch als beträchtliche Herausforderung herausstellen. Während gängige Retrievalsysteme in anderen Nutzungszusammenhängen mit einer überschaubaren Anzahl

[6] Ein prominentes Beispiel für einen solchen Ansatz ist Douglas Bibers „Multidimensionale Analyse" zur Aufdeckung von Zusammenhängen zwischen linguistischen Features und textspezifischen Variationsparametern; vgl. hierzu Biber (1993b), Biber/Conrad/Reppen (1998) und Sardinha/Pinto (2014).

verschiedenartiger Suchattribute operieren (Lagerverwaltungsysteme etwa mit eindeutigen Bestellungs- und Kundennummern, Internet-Suchmaschinen mit Volltext-Schlüsselwörtern sowie eng umrissenen Dokumentattributen wie „Sprache“ oder „Dateityp“) und auf diesen Datenstrukturen nach mittlerweile erprobtem Muster Indizes für performante[7] Suchabfragen anlegen können, lassen sich bei sprachwissenschaftlich motivierten Korpusrecherchen die abzufragenden Suchattribute schwerlich vorab eingrenzen. Um den möglichen Erkenntnisgewinn nicht von vornherein irreparabel einzuschränken, ist potenziell jedes Primär- und Metadatum sowie jedes Annotationselement auf jeder Informationsebene für die Erkennung bzw. Erklärung linguistischer Regelmäßigkeiten und Zusammenhänge interessant.

Neben der Heterogenität der abzufragenden Informationstypen spielt für Systemmodellierungen auch das schiere Korpusvolumen eine bedeutsame Rolle. Entscheidend für den Aussagewert empirischer Herangehensweisen bei der Abbildung von Sprachwirklichkeit ist aus statistischer Perspektive die Verfügbarkeit einer mengenmäßig angemessenen Datenbasis. Methodisch valide quantitative Aussagen zur intendierten Grundgesamtheit (also dem tatsächlichen Sprachgebrauch, einer im Grunde fiktiven Größe) setzen umfangreiche Stichproben (*samples*) voraus, gerade um damit auch seltenere Phänomene präzise auffinden und beschreiben zu können. Dieser Umstand schlägt sich seit Jahren in einem positiv beschleunigten Größenwachstum einschlägiger Textsammlungen nieder:

> Thirty years ago when this research started it was considered impossible to process texts of several million words in length. Twenty years ago it was considered marginally possible but lunatic. Ten years ago it was considered quite possible but still lunatic. Today it is very popular. (Sinclair 1991, S. 1)

Mittlerweile sind Textkorpora mit Tokenzahlen im zweistelligen Milliardenbereich keine Fiktion mehr, nicht zuletzt weil sich die hard- und softwaretechnischen Voraussetzungen zur Speicherung großer Datenmengen sprunghaft verbessert haben. Außerdem bietet sich – selbstverständlich stets unter Beachtung von Fragen des Urheber- und Persönlichkeitsrechts – beim Korpusaufbau neben der Kooperation mit „traditionellen“ Textgebern (d.h. Verlagen) zunehmend der Rückgriff auf die immense Menge frei verfügbarer Internet-

[7] Mit dem Begriff „Performanz“ soll – an dieser Stelle sowie im weiteren Verlauf – im Kontext von Korpusrecherchesystemen bzw. Korpusabfragen in erster Linie auf den Geschwindigkeitsaspekt Bezug genommen werden. Performante Suchabfragen zeichnen sich in diesem Sinne durch minimierte Antwortzeiten aus.

quellen an.[8] Das Anlegen umfangreicher Textarchive ist dadurch selbst für kleinere und mittlere Forschungsprojekte realisierbar geworden und es gehört wenig prophetische Gabe dazu, für die kommenden Jahre eine weitere massive Zunahme sehr großer Sprachkorpora für die unterschiedlichsten Anwendungsgebiete vorauszusagen.

Diese aus empirischer Sicht („more data are better data"; Church/Mercer 1993) grundsätzlich begrüßenswerte Entwicklung entpuppt sich allerdings zunehmend als digitale Achillesferse der praktischen Korpusarbeit: Korpusverwaltungs- und -recherchesysteme müssen mit einem außergewöhnlichen quantitativen Wachstum sowie vielschichtigen Rahmenbedingungen und Anforderungen Schritt halten und sich zwangsläufig zu hochspezialisierten Softwareprodukten entwickeln. Dabei verbinden wir mit „more data" nicht allein Probleme der Skalierbarkeit. Mindestens ebenso bedeutsam erscheint die Beherrschung der Heterogenität der linguistisch relevanten Primär- und Metadaten. Anders als traditionelle Volltextsuchmaschinen oder Information Retrieval (IR)-Systeme sollen Korpussysteme dabei

1) nicht allein die horizontale Verkettung von Sprachelementen abdecken, sondern darüber hinaus
2) die multiple Gliederung in recherchierbare Einheiten (z.B. durch Tokenisierung) erlauben,
3) hochkomplex strukturierte (ggf. konkurrierende) Annotationen der unterschiedlichsten linguistischen Beschreibungsebenen sowie
4) mannigfaltige inner- und außersprachliche Metadaten erschließbar machen. Weiterhin gilt es,
5) dem Aspekt der Persistenz als Grundvoraussetzung solider wissenschaftlicher Forschung höchste Aufmerksamkeit zu schenken, also einmal gewonnene Rechercheergebnisse auch bei wechselndem (hier zumeist: anwachsendem) Datenbestand exakt referenzierbar,[9] reproduzierbar und damit nachvollziehbar zu halten, sowie
6) standardisierbare Schnittstellen für eine statistische Weiterverarbeitung bereitzustellen, da umfangreiche Ergebnislisten kaum mehr unmittelbar linguistisch interpretierbar sind.

---

[8] Vgl. z.B. Biemann et al. (2013); Carstensen et al. (2010); Gatto (2014); Kilgarriff/Grefenstette (2003); Lüdeling et al. (2007); Renouf/Kehoe (2013); Schäfer/Bildhauer (2013); Schneider et al. (2013); Volk (2002).

[9] Für die Referenzierung von Sprachressourcen spezifiziert beispielsweise der ISO-Standard 24619:2011 ein PID (*persistent identifier*)-basiertes Framework: „Language resource management – Persistent identification and sustainable access (PISA)"; vgl. www.iso.org/iso/catalogue_detail.htm?csnumber=37333.

Diese Punkte unterscheiden Korpussysteme signifikant von Online-Textrecherchediensten wie Google, Bing oder Yahoo. Deren auf einem vergleichsweise unkomplizierten Schlüsselwort-Index sowie einer verteilten – und dadurch zwar weitgehend ausfallsicheren, aber nicht jederzeit konsistenten – Infrastruktur basierende Angebote können deshalb weder als Leitbild noch als Machbarkeitsstudie für das Design linguistischer Ressourcen dienen. Beispielsweise erleichtern sich Internet-Suchmaschinen den Umgang mit für sie hinderlichen Spracheigenheiten bisweilen durch gezielte Kunstgriffe, etwa durch das Ausfiltern hochfrequenter Funktionswörter aus den Suchindizes. Auch müssen sie, ebenso wie Frontends für soziale Netze, zwar kurze Abfragezeiten bieten, dürfen hierfür jedoch gegebenenfalls die berücksichtigte Datenbasis situationsbedingt einschränken. All dies führt zu Einschränkungen bei Recall und Precision, die für das wissenschaftliche Korpusretrieval nicht akzeptabel sind. Weiterhin ist das Auswahlverfahren der angezeigten Fundstellen bei traditionellen Internet-Recherchen gemeinhin nicht transparent, d.h. der Endnutzer kann nicht feststellen, ob diese tatsächlich zufällig aus der Grundgesamtheit ermittelt wurden. Da für induktive statistische Berechnungen jedoch echte Zufallsstichproben (*random samples*) erforderlich sind, lässt sich der Wert einer via Google et al. gezogenen Auswahl kaum zuverlässig einschätzen. Für linguistisch motivierte Recherchen birgt all dies schwer kalkulierbare Problematiken; Kilgarriff (2007) spricht nicht zuletzt deshalb von „Googleology" als „bad science".

Einschlägige Forschungsarbeiten beschäftigen sich mittlerweile mit der Modellierung umfangreicher elektronischer Korpussysteme und dabei speziell mit Repräsentation, Austausch und Integration von Mehrebenen-Annotationen sowie mit der Mächtigkeit und Ausdrucksstärke korpuslinguistischer Abfragesprachen.[10] Für die Datenhaltung und das Retrieval kommt in entsprechenden Implementierungen in Abhängigkeit von Projekthintergrund und verfügbarem Know-how ein breites Spektrum technischer Lösungen zum Einsatz, das von dateibasierten Eigenentwicklungen bis hin zu Datenbankmanagementsystemen

[10] Vgl. z.B. Baumann et al. (2004); Burghardt/Wolff (2009); Carletta et al. (2005); Chiarcos et al. (2008); Christ (1994); Dipper et al. (2007); Evert/Fitschen (2010); Evert/Hardie (2011); Farrar (2006); Gleim et al. (2007); Good/Hendryx-Parker (2006); Gut et al. (2004); Ide/Suderman (2007); Jakubíček et al. (2010); Janus/Przepiórkowski (2007); Kepser et al. (2010); Kepser (2003); Lezius (2002); Le Maitre et al. (1998); Müller (2005); Rehm et al. (2009); Steiner/Kallmeyer (2002); Witt (2002); Wynne (Hg.) (2005); Zeldes et al. (2009). Zur Interoperabilität zwischen verschiedenen Abfragesystemen arbeitet die ISO TC37 SC4 Working Group 6 (ISO/WD 24623-1) mit dem Ziel der Standardisierung einer „Corpus Query Lingua Franca" (CQLF; vgl. Mueller 2010; Bański et al. 2014).

(DBMS) unterschiedlicher Provenienz[11] (relational, objektorientiert, Graphdatenbanken, dokumentenorientierte/XML-Datenbanken, NoSQL, sortierte Key-Value-Speicher etc.) reicht. Als vergleichsweise spärlich erforscht erscheinen nach wie vor viele der damit verbundenen informatischen Grundlagen, etwa die gezielte Optimierung von Speichermodellen für sehr große Mengen textueller Primär-, Annotations- und Metadaten (*very large corpora*) sowie die konkrete Implementierung performanter Suchanfragen mit komplexen Suchparametern oder regulären Ausdrücken.

Doch exakt bei der Umsetzung dieser zentralen Anforderungen stoßen existierende Systeme allzu häufig an ihre Grenzen. Hier gilt es, der in der Informatik wohlbekannten Skalierungsproblematik[12] zu begegnen: Die durch die erfolgreiche Weiterentwicklung von Speichermedien und Annotationswerkzeugen sowie die zunehmende Verfügbarkeit digitaler Quellen (also elektronisch verwalteter Buch- und Zeitungstexte sowie z.B. frei erhältlicher Webinhalte) anfallende Menge an Sprachdaten übersteigt immer öfter unsere Möglichkeiten der Auswertung. Und in gleicher Weise gilt, dass sich ohne flexibel parametrisierbare Retrievalschnittstellen vergleichsweise wenig Nutzen aus großen, mehrfach annotierten und mit Metadaten angereicherten Sprachkorpora ziehen lässt.

Das in Abbildung 1 dargestellte Wachstum des Umfangs einiger ausgewählter prominenter Korpussammlungen für das Englische illustriert die Dringlichkeit, nachhaltig effektive Lösungen für die Verarbeitung immer voluminöserer Sprachdatenmengen aufzuzeigen. Während des zurückliegenden halben Jahrhunderts ist ein näherungsweise exponentielles Wachstum hinsichtlich der Anzahl gespeicherter Wortformen erkennbar:[13]

– Das seit Anfang der 1960er Jahre an der Brown Universität in Providence/Rhode Island kompilierte „Brown University Standard Corpus of Present-

[11] Dabei scheint der relationale Ansatz nach wie vor prominent vertreten zu sein, z.B. im Leipziger Wortschatz-Projekt (http://wortschatz.uni-leipzig.de) oder als Grundlage des im Sonderforschungsbereich 632 entwickelten ANNIS (http://annis-tools.org). Unabhängig vom tatsächlichen Einsatz relationaler DBMS für das Korpusretrieval wird allerdings gelegentlich auch deren grundsätzliche Eignung für moderne datenintensive Spezialanwendungen kritisch hinterfragt (Stonebraker et al. 2007).

[12] Skalierung bzw. Skalierbarkeit wird in diesem Zusammenhang als Problem der Softwareentwicklung verstanden, den Zugriff auf wachsende Datenmengen unter Vermeidung überproportionaler Abfragezeiten zu gewährleisten.

[13] Vgl. z.B. Zinsmeister (2010, S. 488). Weiterhin muss berücksichtigt werden, dass nicht nur die Anzahl der Wörter (also der Primärdaten) rapide zunimmt, sondern zusätzlich die Annotations- und sonstigen Metadaten zur Steigerung des auszuwertenden Datenvolumens beitragen. In Kapitel 2 wird dieser Umstand vertiefend thematisiert.

Day American English“ (kurz „Brown-Korpus“, vgl. Kucera/Francis 1964), gemeinhin als Urvater und Prototyp der ersten Generation digitaler Korpora angesehen, enthielt 1.014.312 (ca. 10^6) fortlaufende Wortformen. Die Inhalte waren in 500 Textauszüge (*samples*) aufgeteilt, mit Angaben zu Genre bzw. Texttyp versehen und in einer nachfolgenden Version auch um Wortklasseninformationen (*POS tags*) angereichert.

– Im zwanzig Jahre später von John Sinclair in Birmingham/UK initiierten COBUILD-Korpus (Moon (Hg.) 2009), auch als „Bank of English“ bekannt und Grundlage für *Collins' Cobuild English Language Dictionary*, versammelten sich bereits Mitte der 1980er Jahre ca. 10 Millionen (10^7) Wortformen; mittlerweile umfasst das Korpus mehr als 2 Milliarden Token.

– Für die sogenannte zweite Generation digitaler Korpora steht exemplarisch das „British National Corpus“ (BNC) mit 100 Millionen (10^8) Token (Aston/Burnard 1998). Zwischen 1991 und 1994 wurde dieses statische Referenzkorpus für primär schriftsprachliche Inhalte (ca. 90%, daneben ca. 10% mündliche Sprachdaten) mit Samplegrößen von bis zu 45.000 Wörtern zusammengestellt.

– Als Vertreter von Sprachkorpora der dritten Generation mit über 10^9 laufenden Wortformen gelten das WaCky-Projekt („Web as Corpus kool ynitiative“, Baroni et al. 2009), das für vier Einzelsprachen via Web-Crawling jeweils mehrere Milliarden Wortformen gesammelt hat, weiterhin das in ähnlichen Dimensionen angesiedelte COW-Projekt („Corpora from the Web“, Schäfer/Bildhauer 2012; Evaluation der Architektur in Schäfer 2015) oder das English Gigaword-Korpus des Linguistic Data Consortium (LDC) an der Universität von Pennsylvania. Letzteres enthielt in seiner ersten Ausgabe von 2003 (Graff/Cieri 2003) laut Dokumentation 1.756.504.000 Wörter in knapp 12 Gigabyte großen SGML-Dateien; die fünfte Ausgabe von 2011 überschritt die 4-Milliarden-Grenze.

– Unabhängig von der Frage, ob Googles „Web 1T 5-Gram Corpus“ (Brants/Franz 2006) als Web-basierte Sammlung von Textfragmenten (Unigramme bis Pentagramme) sämtliche Kriterien[14] eines wissenschaftlichen Textkorpus erfüllt, stellen die enthaltenen über eine Trillion (1.024.908.267.229, ca. 10^{12}) Wörter sowie Frequenzangaben der verschiedenen N-Gramme eine potenziell ertragreiche Ressource für vielfältige linguistische Untersuchungen dar.

[14] Bedenkenswert erscheinen etwa die Stratifizierung der Sammlung sowie der Umstand, dass keine vollständige Texte enthalten sind, auf deren Basis quantitativ verlässliche wort- und satzübergreifende Aussagen getroffen werden könnten.

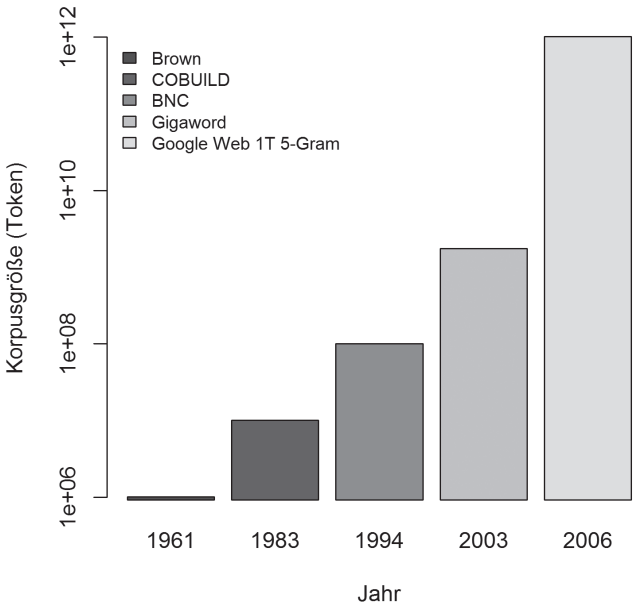


Abb. 1: Erstellungszeit und Umfang ausgewählter Korpora (logarithmische Darstellung)

Ein naheliegender informationstechnologischer Ansatz zur Überwindung der mit dieser Entwicklung einhergehenden Engpässe ist die Nutzung von Programmier-Paradigmen und -Frameworks zur Aufteilung komplexer Operationen in separate Einzelschritte sowie zu deren nebenläufiger Abarbeitung. In diesem Zusammenhang erfreuen sich in der Informatik seit den 1970er Jahren Modelle wie Datenfluss-Programmierung (*dataflow programming*) oder Aktoren (*actors*) großer Beliebtheit, ein jüngeres populäres Beispiel ist der Map-Reduce-Ansatz. Dies ist vor dem Hintergrund einer steigenden Anzahl verteilter Netzwerkarchitekturen und Mehrkernsysteme zu sehen: Während noch bis vor wenigen Jahren Einzelkernprozessoren den Computermarkt dominierten und Leistungssteigerungen primär durch eine kontinuierliche Erhöhung der CPU-Taktfrequenz erreicht wurden, verbreiten sich im PC- und Serverbereich mittlerweile verstärkt Mikroprozessoren mit zwei, vier, acht oder mehr Prozessorkernen. Solche Systeme lassen sich wiederum zu Multiprozessor-Architekturen zusammenfassen, so dass bereits in mittelgroßen Rechenzentren Cluster mit zwei- bis dreistelligen Prozessorkernzahlen zum Einsatz kommen. Ein Ende dieses Trends, der bis in das Hochleistungsspektrum hineinreicht, ist nicht in Sicht.

Es bietet sich an, dieses parallele Hardware-Potenzial für das rechenintensive Korpusretrieval zu nutzen. Die in den nachfolgenden Kapiteln vorgestellte und evaluierte Grundidee besteht darin, nicht allein die zu analysierenden Ausgangsdaten zu segmentieren (datenorientierte Parallelisierung), sondern komplexe linguistische Phänomenbeschreibungen in überschaubare, unabhängig voneinander abarbeitbare Aufgaben zu unterteilen (problemorientierte Parallelisierung). Aus computerlinguistischer Sicht besteht ein wissenschaftspraktisches Desiderat in der Erforschung der Angemessenheit dieses Ansatzes für korpuslinguistische Szenarien, d.h. in der Evaluation unter Verwendung genuin linguistischen Datenmaterials. Aus sprach- und informationstechnologischer Perspektive wäre dabei insbesondere zu untersuchen, nach welchen Kriterien komplexe Korpusrecherchen[15] zielführend in kausal unabhängige Teilabfragen segmentiert werden können, also welche inner- und außersprachlichen Parameter im Einzelnen kombiniert oder separat abgefragt werden sollen. Aus technischer Perspektive sollte überprüft werden, wie solche Teilabfragen parallelisiert werden können, welche Speicher- und Abfragekonstrukte dabei für die Echtzeitverarbeitung[16] in Betracht kommen und ob – und wenn ja: ab/bis zu welchem Level – dieses Vorgehen signifikante Vorteile gegenüber nicht-zerlegten Abfragen erwarten lässt. Weiterhin sind Überlegungen zur Koordination anzustellen, um um klären, wann und wie einzelne Teilergebnisse zusammenzuführen bzw. weiterzuverarbeiten sind. Dabei gilt es zu berücksichtigen, dass verteilte Berechnungen segmentierter Probleme zumeist mit einem insgesamt höheren Rechenbedarf und zusätzlichem Verwaltungsaufwand einher gehen, umfangreiche Teildatenmengen beim Bearbeiten bzw. Verschieben entsprechend hohe Latenzzeiten produzieren – und parallele Algorithmen ganz grundsätzlich ein ungleich komplexeres Laufzeitverhalten an den Tag legen als sequenzielle Implementierungen.

Neben diesen zentralen Performanzfragen beinhaltet die Konzeption empirisch nutzbarer Korpusrecherchesysteme idealerweise auch Schnittstellen zu statistischen Datenanalysen und deren Visualisierung in Form von Baumgraphen, Diagrammen, Karten oder Netzdarstellungen (vgl. Bubenhofer/Ku-

[15] Damit sind Abfragen gemeint, die kombiniert auf den sprachlichen Primärdaten, linguistischen Annotationen und (außersprachlichen) Metadaten operieren, also z.B. „Suche alle Verbzweitsätze mit satzeinleitendem Subjunktor (*weil*, *obwohl*, *wobei* etc.) in Reden und Interviews der letzten zehn Jahre".

[16] Korpusrecherchen werden in der wissenschaftlichen Praxis in der Regel ad hoc durchgeführt und sollten innerhalb enger Zeitschranken zu Ergebnissen führen. Das schließt selbstverständlich nicht aus, dass für spezielle Probleme auch längere Stapelverarbeitungen notwendig sein können. Gerade für die Akzeptanz von Online-Abfragesystemen ist jedoch erfahrungsgemäß die Vermeidung massiver Verzögerungen entscheidend.

pietz (Hg.) 2018). Linguisten erwarten als Ergebnis einer Korpusabfrage nicht allein eine Liste von Fundstellen oder eine KWIC (*key word in context*)-Ansicht, sondern bedienen sich deskriptiver oder induktiver statistischer Methoden und Testverfahren zur Überprüfung weiterführender Aussagen. Beispiele hierfür sind Angaben zur Streuung der Daten, zur Standardabweichung oder zur Signifikanz beobachteter Zusammenhänge (z.B. Chi-Quadrat-Test oder Log-Likelihood-Test). Entsprechende Berechnungen lassen sich ab einer gewissen Größenordnung kaum noch mit den weitverbreiteten Kalkulationsprogrammen à la Excel durchführen, sondern erfordern spezialisierte Statistiksoftware wie z.B. das „Ngram Statistics Package (NSP)"[17] oder das GNU-Teilprojekt „R".[18] Speziell die 1992 von Ross Ihaka und Robert Gentleman entwickelte statistische Programmiersprache R erfreut sich dank ihrer Mächtigkeit und Erweiterbarkeit großer Popularität in der korpuslinguistischen Praxis.[19] Retrievalsysteme für Sprachkorpora sollten folglich zumindest normierte Schnittstellen zum Export von Ergebnisdaten in entsprechende Statistiksoftware vorsehen. Da das dabei zu transportierende Datenvolumen und die damit einher gehenden Latenzzeiten eine Weiterverarbeitung in Echtzeit tendenziell erschweren, erscheint darüber hinaus perspektivisch eine unmittelbare Integration statistischer Pakete und Prozeduren in Retrievalsysteme wünschenswert (vgl. Kupietz/Diewald/Fankhauser 2018).

Im vorliegenden Buch soll ein konkreter Modellierungs- und Implementierungsansatz vorgestellt werden, der sich der geschilderten Problematiken multidimensionaler Korpusrecherchen annimmt. Ausgehend von einer Charakterisierung der wesentlichen Anforderungsmerkmale linguistisch motivierter Retrievalsysteme sollen angemessene Speicherungs- und Abfragestrategien für mehrfach annotierte Sprachkorpora entwickelt, evaluiert und – wo möglich – optimiert werden. Prinzipielle (architektonische) Performanzprobleme bei der Verarbeitung sukzessive anwachsender Sprachdatenbestände gilt es dabei gezielt einzukreisen, um durch empirische Messungen auf einem Referenzsystem zu begründeten Entscheidungsgrundlagen für das Systemdesign zu gelangen. Darauf aufbauend wird ein Abfragemodell präsentiert, das komplexe Korpusanfragen problemorientiert segmentiert. Seine Wirksamkeit wird ebenfalls anhand einer Referenzimplementierung gemessen und bewertet. Insgesamt sollen auf diese Weise die Handhabung von *very large corpora* sowie die passgenaue Implementierung von Korpusrecherchesystemen umfassend informatisch unterstützt werden.

---

[17] Das NSP ist frei verfügbar unter www.d.umn.edu/~tpederse/nsp.html.

[18] Zentrale Website ist das „Comprehensive R Archive Network" (R Core Team (Hg.) 2016).

[19] Vgl. z.B. Baayen (2008); Gries (2008, 2016); Hansen-Morath et al. (2018); Ihaka/Gentleman (1996); Meindl (2011).

Angesichts des interdisziplinären Umfelds und der als grundlegend erachteten Prämisse, dass sich das erarbeitete Modell in möglichst vielen konkreten Projektkontexten bzw. Infrastrukturen zu bewähren hat, werden wir gezielt auf etablierte Standards zurückgreifen. Hierzu gehören in erster Linie:

– die „Extensible Markup Language“ (XML) sowie flankierende Standards wie XPath für Erstellung, Austausch und Import annotierter Textkorpora sowie für die Adressierung informationstragender Fragmente,
– Datenbankmanagementsysteme (DBMS) für die plattformunabhängige, persistente und skalierbare Speicherung heterogener Datentypen,
– die „Structured Query Language“ (SQL) als gleichermaßen funktionsmächtige und flexible Abfragesprache.

Das nachfolgende Kapitel 2 beschäftigt sich mit der Bedeutung umfangreicher natürlichsprachlicher Korpora für die linguistische Hypothesenbildung bzw. für die Präzisierung von Annahmen darüber, wie menschliche Sprache funktioniert. Es bietet, mit einem Fokus auf deutschsprachige Textkorpora, einen Überblick über Standards, existierende Korpussammlungen und deren Abfrageschnittstellen. Anhand ausgewählter, gleichermaßen linguistisch wie technisch motivierter Fallbeispiele wird in die Problematiken der Mehrebenen-Annotation und multidimensionaler Suchkriterien zur Beschreibung komplexer Sprachphänomene eingeführt. Darauf aufbauend entwickeln wir einen exemplarischen Anforderungskatalog für linguistisch motivierte Korpusabfragen.

Kapitel 3 diskutiert informatische Aspekte hinsichtlich Verwaltung und Verarbeitung sehr großer Korpusvolumina. Einleitend erfolgt eine Vorstellung des technischen Spektrums verfügbarer Speicherungsoptionen. Anschließend thematisieren wir Modellierung und Implementierung unseres Referenz-Prototyps. Die getroffenen Designentscheidungen werden hinsichtlich ihrer grundsätzlichen Adäquatheit empirisch evaluiert. Die ermittelten absoluten Abfragezeiten sollen keinesfalls den Ausgangspunkt für ein Benchmarking unterschiedlicher Korpusdatenbanken bilden – eine diesbezüglich umfassende Analyse ist aufgrund variabler Hardware-Voraussetzungen und Korpusgrößen in wechselnden Projektkontexten schwerlich exhaustiv durchführbar. Stattdessen werden verschiedenartige Modelle relativ zueinander in Bezug gesetzt und grundlegende Tendenzen bzw. Auswirkungen aufgezeigt.

Um der praktischen Relevanz der Thematik Rechnung zu tragen, findet in Kapitel 4 eine Evaluation der linguistisch motivierten Beispielabfragen des Anforderungskatalogs aus Kapitel 2 auf zwei abgestuft leistungsfähigen Hardwareplattformen und mit variablen Korpusvolumen statt. Dabei wird

insbesondere die Leistungsfähigkeit mehrfach verketteter Datenbank-Queries auf den Prüfstand gestellt. Anhand eines Vergleichs von Suchattributen und Laufzeiten grenzen wir für die korpuslinguistische Praxis problematischen Abfragen ein.

Kapitel 5 versucht sich an der weiterführenden Optimierung komplexer Korpusabfragen. Dabei geht es insbesondere um die Algorithmisierung und Implementierung der im vorigen Kapitel ermittelten „Flaschenhälse". Ein wesentlicher Schwerpunkt liegt in der Darstellung softwaregesteuerter, problemorientierter Segmentierung und Parallelisierung als Chance für den Umgang mit sehr großen Sprachdatenmengen. Durch die Aufteilung von Korpusrecherchen mit mehreren heterogenen Suchattributen in kleinere Suchschritte sollen Gesamtlaufzeiten signifikant reduziert werden. Unter Einbeziehung quantitativer Eigenarten natürlicher Sprache (bzw. deren Repräsentation in Form von Text- und Annotationsdaten) wird geklärt und evaluiert, wie sich korpuslinguistische Abfragen idealerweiser segmentieren lassen. Das vorgestellte Verfahren versteht sich nicht als Konkurrenz, sondern als fallspezifische Ergänzung zu etablierten vorgelagerten (z.B. physikalisches Clustering) oder nachgelagerten (z.B. interne Anfrageoptimierer) Techniken.

In Kapitel 6 wird ein webbasierter Prototyp präsentiert, der ein Retrievalinterface für die evaluierten Suchalgorithmen bereitstellt. Diese Anwendung umfasst, neben statischen und dynamischen Übersichtslisten relevanter Korpusinhalte, flexibel parametrisierbare Suchformulare für das Auffinden linguistischer Phänomene. Für statistische Auswertungen der Retrievalergebnisse existiert eine Schnittstelle zu mithilfe der Statistiksoftware „R" programmierten Modulen.

Kapitel 7 schließlich fasst die gewonnenen Erkenntnisse zusammen und bietet punktuelle Ausblicke auf weiterführende Fragestellungen bzw. Einsatzszenarien.

Die einzelnen Kapitel entstanden im Rahmen der Forschungsprojekte „Korpusgrammatik – grammatische Variation im standardsprachlichen und standardnahen Deutsch" und „Grammatische Datenbanken und Informationssysteme" am Institut für Deutsche Sprache (IDS) in Mannheim. Gegenstand dieser Unternehmungen ist die korpusgestützte Erforschung der Variation im standardsprachlichen und standardnahen Deutsch. Die gewonnenen Erkenntnisse bilden eine Grundlage für die Erstellung einer Grammatik des Deutschen, in der Variation im Sprachgebrauch empirisch fokussiert und umfassend aufgearbeitet wird.[20] Auf Basis der morphosyntaktisch annotierten

[20] Vgl. hierzu Bubenhofer et al. (2014); Fuß et al. (2018); Konopka et al. (Hg.) (2011) sowie Konopka/Fuß (2016).

IDS-Korpora geschriebener Sprache wurde ein Projektkorpus mit knapp acht Milliarden Textwörtern definiert, das zentrale Textklassen sowie nationale und großregionale Varietäten des Deutschen in geeigneter quantitativer Relation repräsentiert. Der in den Kapiteln 3 bis 6 vorgestellte Prototyp kann in diesem Zusammenhang als Referenz für zukünftige Recherchelösungen dienen.

2. Linguistische Anforderungen an Sprachkorpora

Um die Bedeutung umfangreicher natürlichsprachlicher Korpora für die Erforschung und Analyse menschlicher Sprache sachgemäß einschätzen zu können, lohnt sich ein kurzer Blick auf die grundlegenden Ziele wissenschaftlicher Arbeit. Für Real- oder Erfahrungswissenschaften – also für diejenigen akademischen Disziplinen, die sich real existierende Objekte und Sachverhalte als Forschungsgegenstand gewählt haben – lassen sich diese Ziele durch den Dreischritt „Deskription – Explikation – Prognose" ausdrücken. Unter Deskription verstehen wir dabei die systematische Beschreibung beobachtbarer Phänomene unter Verwendung präziser Begrifflichkeiten, verlässlicher Erhebungsinstrumente und geeigneter Methoden. Um Hypothesen und Erkenntnisse intersubjektiv nachvollziehbar zu machen, sollten die durchgeführten Untersuchungen widerspruchsfrei dokumentiert sowie reproduzierbar – und damit überprüfbar – sein. Auf die Deskription aufbauend richtet sich das weiterführende Erkenntnisinteresse gemeinhin auf das Formulieren von Erklärungen zu Beschaffenheit, Funktion und Entwicklung des empirisch explorierten Realitätsausschnitts (Explikation). Dies geschieht idealerweise durch den Aufbau einer Hierarchie empirischer Regelhaftigkeiten, die zunächst Einzelaussagen erklären und in einem Wechsel aus induktiven und deduktiv-nomologischen Schritten allmählich zu allgemeingültigen Gesetzen und explanatorischen Theorien führen. Diese decken sukzessive immer größere Bereiche der beobachteten Realität ab, schließen Erkenntnislücken und erlauben in der Folge generalisierende wahrheitsfähige Vorhersagen auch über bislang nicht explizit untersuchte Phänomene (Prognose).

Eine elementare Aufgabe von Wissenschaft besteht darin, relevante Fragen zu stellen – und im besten Falle wohlbegründete Antworten zu liefern. Dazu bedarf es eines für den Untersuchungsgegenstand adäquaten Instrumentariums. Bezogen auf den Gegenstandsbereich der natürlichen Sprache impliziert dieser Ansatz, dass Hypothesen und weiterführende Erklärungen zur Beschaffenheit des Explanandums statistisch-probabilistischer Art sein sollten. In den Naturwissenschaften besitzt diese Maxime bereits eine längere Tradition, und sie ist aus mehreren Beweggründen auch auf konkretes menschliches Sprachverhalten anwendbar. Zum einen, weil Heuristiken bereits für unvollständige Datensets belastbare Aussagen produzieren können und damit tendenziell für Grundlagenforschungen mit einem zwangsläufig begrenzten Wissensstand angemessen erscheinen. Zum anderen aufgrund des stochastischen Charakters von Sprache, die sich auf einem komplexen

Zusammenspiel von Determinismus und Zufälligkeit (vgl. Köhler 1986) gründet: Die syntagmatische Konstruktion von Sprachäußerungen, gleich unter welchen medialen (z.B. akustisch vs. grafisch/visuell), interaktiven (z.B. dialogisch vs. monologisch) oder sonstigen Rahmenbedingungen, geschieht nicht rein deterministisch nach einem strikt vorgegebenen Muster. Vielmehr wählt der Sprecher/Schreiber sukzessive unter verschiedenen Möglichkeiten. Und auch die diachrone Entwicklung von Sprachen wäre unter Annahme ausnahmslos deterministischer Regeln schwerlich vorstellbar, vielmehr vollzieht sich Sprachwandel auf allen linguistischen Beschreibungsebenen (Phonologie, Morphologie, Syntax, Semantik) durch die Verstärkung bzw. Abschwächung von Tendenzen unter bestimmten inner- und außersprachlichen Rahmenbedingungen. Es liegt also auf der Hand, dass quantitative „Je-desto"-Aussagen angemessener für die realitätsnahe Beschreibung sprachlicher Phänomene sind als rein deterministische „Wenn- dann"-Regeln.

Dieser Umstand geht einher mit einer sprachwissenschaftlichen Präferenz für intervall- und verhältnisskalierte Begrifflichkeiten. Im Unterschied zu den Ausprägungen einer Nominalskala ohne natürliche Rangfolge erlauben diese die Zuordnung gradueller Merkmalsausprägungen zu einer Zahlenmenge sowie eine Interpretation von Rangfolge und Abständen auf der Kardinalskala. Ein typisches Beispiel hierfür ist das Konzept der Mehrdeutigkeit: Mit einem nominalskalierten Polysemie-Begriff sind lediglich zwei Ausprägungen ausdrückbar, nämlich ob ein sprachlicher Ausdruck für unterschiedliche Bedeutungsinhalte steht oder nicht. Erst eine intervall- bzw. verhältnisskalierte Variante ermöglicht Aussagen darüber, wie viele Bedeutungen sich genau identifizieren lassen, sowie weiterführende empirische Untersuchungen zu den verschiedenen Ausprägungen.

Über dieses Beispiel hinaus finden sich in der Sprachforschung leicht weitere Belege für einen methodologischen Fortschritt durch quantitative Begrifflichkeiten. Beispielsweise bezeichnet „Kookkurrenz" qualitativ das gemeinsame Auftreten zweier sprachlicher Einheiten A und B in derselben übergeordneten Einheit (zumeist Worte innerhalb desselben Satzes), ggf. unter Festlegung der maximalen Wortdistanz (häufig 1, man spricht dann von Nachbarschaftskookurrenzen). Bereits eine komparative Erweiterung des Begriffs ermöglicht den Ausdruck von Graduierungen (A tritt häufiger gemeinsam mit B als mit C auf) und lenkt den Blick auf diejenigen linguistisch zumeist besonders interessanten Fälle, in denen das gemeinsame Auftreten häufiger zu beobachten ist als bei einer reinen Zufallsverteilung. Bildet man darüber hinaus die Frequenzen der ermittelten syntagmatischen Muster unter Einbeziehung der Gesamtzahl aller untersuchten Wörter bzw. Sätze als

relative Werte auf eine metrische Skala ab, so lassen sich Kookkurrenzstärken für das gemeinsame Auftreten linker und rechter Nachbarn exakt ermitteln.[21] Die Signifikanz einer Kookkurrenz wird damit operationalisierbar und Kookkurrenzpartner werden mit mathematischen Methoden vergleichbar – was z.B. bei der Suche nach einer fundierten grammatischen oder semantischen Interpretation des Phänomens hilfreich sein kann.[22] Vermittels statistischer Tests kann unter fixierten Rahmenbedingungen die ungerichtete Hypothese, dass bestimmte Kookkurrenzpartner voneinander unabhängig und das gemeinsame Vorkommen durch bloßen Zufall erklärbar ist, mit einer festlegbaren Irrtumswahrscheinlichkeit abgelehnt oder angenommen werden.

Deskriptive Ansätze, die im Sinne Humboldts nicht nur das „todte Gerippe"[23] menschlicher Sprache analysieren, sondern verlässliche empirische Sachaussagen zum realen Sprachgebrauch treffen wollen, benötigen authentisches Sprachmaterial. Darunter verstehen wir Texte oder transkribierte Ton-/Videoaufnahmen, die in möglichst natürlichen, alltagsnahen Sprachgebrauchssituationen entstanden sind: Erzählungen, Berichte, Interviews, Gespräche usw. Allerdings ist unter dem Gesichtspunkt der Authentizität nicht jede potenziell verfügbare Sprachdatenquelle für beliebige empirische Analysen des Untersuchungsgegenstands geeignet. „Künstliche" Inhalte, also beispielsweise konstruierte Beispielsätze oder auch lyrische Texte mit besonderen syntaktischen Organisationsprinzipien, erfüllen in diesem Sinne nicht ohne weiteres die Aufnahmekriterien für sprachwissenschaftliche Korpora – es sei denn, das Ziel der avisierten Erhebung sei eine spezielle (etwa literaturwissenschaftliche) Forschungsfrage. Beispiele hierfür sind empirische Bewertungen stilistischer Eigenheiten, Untersuchungen zur Evolution des Schreibstils einzelner Autoren oder die Autorenbestimmung vermittels statistischer Textmerkmale.[24]

---

[21] Perkuhn/Belica (2004) verwenden beispielsweise die log-likelihood-ratio (LLR) als Maßzahl für die Abweichung des normalen vom beobachteten Verhalten.

[22] Gleiches gilt im Übrigen für den in der Linguistik notorisch vage und uneinheitlich verwendeten Begriff der Kollokation (zumeist: konventionalisierte Verbindung zwischen sprachlichen Einheiten mit primär semantischem Hintergrund); einen Überblick vermitteln z.B. die theoretischen Vorbemerkungen in Konecny (2010) sowie Kapitel 11 in Kunze/Lemnitzer (2007). Eine quantitative Interpretation kann auch hier zur Schärfung der wissenschaftlichen Terminologie (vgl. Suchowolec et al. 2018) beitragen.

[23] „Auch kann das Studium der Sprachen nicht von dem ihrer Litteraturen getrennt werden, da in Grammatik und Wörterbuch nur ihr todtes Gerippe, ihr lebendiger Bau aber nur in ihren Werken sichtbar ist." (Humboldt 1988, S. 428).

[24] Vgl. z.B. Brocardo et al. (2013); Cheng (2012); Fialho/Zyngier (2014); Laffal (1997); Wickmann (1989); Zenkov (2017).

Authentisches Sprachmaterial kann entweder in Form eines Sprachdatenarchivs bzw. einer maschinenlesbaren Sammlung von Texten bereitgestellt werden und lässt sich dann dank computerlinguistischer Vorarbeiten für unterschiedliche Untersuchungszwecke nutzen, oder aber es wird im Kontext der angestrebten Untersuchungen zielgerichtet zusammengestellt. Eine solche intentionale Datenerhebung kann durch die Erfassung von Spontandaten (durch Interviews, Labor- oder Feldexperimente usw.) oder Semi-Spontandaten (etwa durch die Auswertung von Fragebögen) geschehen. In all diesen Fällen sollte allerdings berücksichtigt werden, dass der situative Kontext, also der nicht intendierte Einfluss einer Kommunikationssituation auf die aufgezeichnete sprachlich-kommunikative Handlung, unter Umständen den Aussagewert der Daten beeinträchtigt. Darüber hinaus erschweren gelegentlich logistisch bedingte Restriktionen eine belastbare Interpretation: Bei den genannten Formen der Datenerhebung fällt bereits für die Akquise und Betreuung kleiner bis mittlerer Gruppengrößen ein nicht unbeträchtlicher organisatorischer Aufwand an; weiterhin erreicht der datensammelnde Forscher – wenn man einmal vom Mittel des elektronischen Online-Fragebogens absieht – lediglich eine begrenzte Anzahl von Probanden. Daraus folgt in der Praxis, dass aus einer zumeist recht überschaubaren Stichprobenmenge auf die Grundgesamtheit geschlossen werden muss, was die Verlässlichkeit der getroffenen Aussagen grundsätzlich mindert.

Vielversprechender als die Spontanakquise erscheinen mithin eine Nutzung existierender, projektunabhängiger Spracharchive sowie die Anwendung korpusgestützter Methoden zur Aufdeckung natürlich entstandener stochastischer Regelhaftigkeiten. Neuere sprachwissenschaftliche Forschungsarbeiten bedienen sich hierzu in zunehmendem Maße immer umfangreicherer elektronischer Korpora, d.h. maschinenlesbarer Sprachdatensammlungen, deren Inhalte in einem linguistisch unreflektierten Kontext entstanden sind und einen unverzerrten Blick auf die Sprachwirklichkeit erlauben.[25] Da implizites Sprachwissen, konventionalisierte Sprachroutinen und neuropsychologische Implikationen der Sprachproduktion nicht explizit fassbar sind, werden Sprachkorpora als Korrelate der Sprachwirklichkeit herangezogen und allgemeine Regularitäten des Sprachverhaltens systematisch abgeleitet. Angereichert werden diese Textsammlungen mit linguistisch motivierten Annotationen sowie außersprachlichen Metadaten. Auf einer derartigen, gleichermaßen in die Tiefe als auch in die Breite erschöpfenden Beobachtungsgrundlage lassen sich linguistische Phänomene empirisch valide analysieren, diskutieren und – im optimalen Fall – erklären. Hypothesen können systematisch über-

[25] Vgl. Altmann/Altmann (2008); Biber/Conrad/Reppen (1998); Bubenhofer et al. (2013); Köhler (2005); McEnery/Wilson (2001).

prüft sowie sprachlich komplexe Muster und Korrelationen mit Hilfe statistischer Werkzeuge aufgedeckt werden.[26] Voraussetzung hierfür ist eine fach- und sachgerechte Erschließung sämtlicher Korpusebenen, was in Anbetracht der mittlerweile erreichten Korpusgrößen eine komplexe interdisziplinäre Herausforderung darstellt.

Typologisch existieren eine Reihe von Vorschlägen für die Klassifizierung von Sprachkorpora nach formalen und inhaltlichen Merkmalen.[27] Unterschieden wird beispielsweise hinsichtlich der Funktionalität (multifunktional angelegte Datensammlungen vs. Spezialkorpora für dezidierte Fragestellungen, Sprachausschnitte, Inhaltsdomänen usw.), des Mediums (textbasiert, verbal oder multimedial/multimodal, incl. Grenzbereiche wie z.B. computervermittelte Kommunikation), der Sprachauswahl (mono-, bi- oder multilingual, aus Übersetzungen aufgebaute Sammlungen heißen auch Parallelkorpora), des Beobachtungszeitraums (synchron vs. diachron), der Erweiterungsfähigkeit (statische Korpora vs. fortlaufend aktualisierte Monitorkorpora) oder der bereitgestellten Sekundärdaten (Annotationen und Metadaten in unterschiedlicher Anzahl und Granularität). Diese Klassifizierungen lassen sich nicht in jedem Fall trennscharf durchführen, vermitteln aber einen grundlegenden Überblick über das Spektrum möglicher Korpusanwendungen und stehen damit stellvertretend für die mannigfaltigen Benutzerwünsche im Kontext der empirischen Sprachforschung.

Die Problematik strikter Korpustypologien wird am Beispiel des dem empirischen Teil des vorliegenden Buchs zugrunde liegenden Deutschen Referenzkorpus DeReKo deutlich: Zwar impliziert der Name strenggenommen den Anspruch, die deutsche Sprache in ihrer Gesamtheit zu dokumentieren (im Englischen existiert hierfür der Begriff des *general corpus*), doch tatsächlich versteht sich DeReKo als „Urstichprobe" (*primordial sample*, vgl. Perkuhn et al. 2012, S. 49) zur Generierung passgenauer Analysegrundlagen, sogenannter virtueller Korpora. Seine Teilbestände umfassen gleichermaßen Monitorkorpora (etwa zahlreicher Zeitschriften und Zeitungen) als auch Fachsprachensammlungen oder historische Quellen. Folglich ist DeReKo zunächst im besten Sinne ein „opportunistisches Korpus", das möglichst viel und dabei möglichst verschiedenartige Sprachbelege in Subkorpora versammelt und dem Anwender die endgültige Entscheidung darüber überlässt, welche Ausschnitte für sein konkretes Erkenntnisinteresse am zweckmäßigsten sind.

[26] Vgl. Gries (2008b); Hansen/Wolfer (2017); Hansen-Morath et al. (2018); Keibel et al. (2008); Lüdeling/Kytö (2008); Manning/Schütze (1999); McEnery et al. (2010); Perkuhn et al. (2012); Schneider (2014); Tognini-Bonelli (2001).

[27] Vgl. hierzu Carstensen et al. (2010); Lemnitzer/Zinsmeister (2015); Scherer (2006) oder Zinsmeister (2010) sowie Bestandsaufnahmen wie z.B. Beal/Corrigan/Moisl (Hg.) (2007).

2.1 Natürlichsprachliche Korpora in der Sprachwissenschaft

Einsatzgebiete empirischer Ressourcen und Methoden umfassen Fragestellungen aus vielen sprachwissenschaftlichen Teildisziplinen und -bereichen. Diese reichen von der linguistischen Grundlagenforschung – beispielsweise zur Aufstellung und Präzisierung von Theorien darüber, wie natürliche Sprache generell funktioniert – sowie pragmatisch, syntaktisch oder semantisch motivierten Untersuchungen in der Allgemeinen Sprachwissenschaft bis hin zur Text- oder Stilanalyse sowie der lexikografischen oder forensischen Forschung in der Angewandten Sprachwissenschaft. Auch in der universitären Lehre werden Korpora zunehmend zur Vermittlung sprachlichen Wissens eingesetzt.[28] Weiterhin beschränkt sich ihr Nutzwert nicht allein auf die akademische Forschungslandschaft. Angesichts einer zunehmenden Durchdringung alltäglicher Lebensbereiche mit moderner Informations- und Kommunikationstechnik profitieren sprachnah angesiedelte Produkthersteller und Entwickler computerlinguistischer Anwendungen von digitalen Sprachressourcen. Diese werden zur Evaluation komplexer *NLP* (*Natural Language Processing*)-Systeme herangezogen, also beispielsweise für das Testen maschineller Übersetzungen, für die Konzeption computergestützter Sprachlernhilfen oder für die Optimierung von Spracherkennung und -generierung für die Mensch-Maschine-Kommunikation (Dialogsysteme etc.).

Korpuslinguistische Ansätze in den genannten Forschungsgebieten sowie für unterschiedliche Einzelsprachen haben in einer gewissen zeitlichen und räumlichen Staffelung Akzeptanz und Verbreitung gefunden. Während in der deskriptiv arbeitenden anglo-amerikanischen Lexikografie das datengetriebene Analyseparadigma bereits seit Jahrzehnten zum Einsatz kommt,[29] beispielsweise für Untersuchungen zur Wortschatzentwicklung, bei der frequenzgesteuerten Auswahl von Lemmastrecken oder zum sprachstatistisch fundierten Auffinden von Einzelbedeutungen (auch Lesarten genannt), bedienten sich andere Bereiche der Linguistik lange Zeit vorrangig Herangehensweisen, die primär durch Introspektion, Kompetenz und Intuition des

[28] Vgl. z.B. Abel/Zanin (Hg.) (2011) und Bubenhofer (2011).

[29] So diente bereits das in den 1960er Jahren kompilierte Brown-Korpus (Kucera/Francis 1964) als Grundlage für das *American Heritage Dictionary of the English Language (AHD)*. Auf der anderen Seite des Atlantiks bildete zwei Jahrzehnte später das COBUILD-Korpus (auch „Bank of English" genannt) die Datenbasis für die Produktion des *Collins COBUILD English Language Dictionary*, vgl. Moon (Hg.) (2009). Vgl. weiterhin z.B. Engelberg/Lemnitzer (2009), Geyken (2004), Klosa (2007), Lemnitzer/Zinsmeister (2015, S. 143ff.), Storjohann (2012b, S. 481), Storrer (2011); neuere grundlegende Überblicke über die Entwicklung der Korpuslinguistik finden sich z.B. in Kupietz/Schmidt (Hg.) (2018), Lobin/Schneider/Witt (Hg.) (2018), O'Keeffe/McCarthy (Hg.) (2012) oder McEnery/Hardie (2013).

Forschers geleitet wurden und bei denen Sprachkorpora eher dokumentarisch-unterstützend zum Einsatz kamen. Das mag zum einen auf wissenschaftstheoretische Überlegungen zurückzuführen sein – etwa auf eine individuelle Orientierung am philosophischen Rationalismus, eine Präferenz für generative Modelle, präskriptive wissenschaftliche Methoden etc. oder auf methodische Vorbehalte hinsichtlich der Fassbarkeit des unendlichen Untersuchungsgegenstands „Sprache“[30] durch notwendigerweise endliche Korpussammlungen. Daneben spielte jedoch sicherlich nicht zuletzt die gelegentlich unbefriedigende Verfügbarkeit authentischer, ausreichend umfangreicher und mit linguistisch relevanten Zusatzinformationen angereicherten Korpora zur Recherche nach morphologisch-syntaktischen Konstrukten, semantischen Relationen oder Interaktionsmerkmalen eine Rolle.[31] In jüngster Zeit deutlich verbesserte Korpusinfrastrukturen inspirieren inzwischen beinahe sämtliche sprachwissenschaftlichen Forschungsbereiche. Korpusgestützte, deskriptive Ansätze eröffnen innovative Blickwinkel auf altbekannte Fragestellungen und haben diverse Bereiche der Linguistik nachhaltig vorangebracht, obwohl bislang noch nicht allzu viele gleichermaßen erprobte wie belastbare Algorithmen zur Verfügung stehen und eine breitere Verankerung quantitativer Methoden z.B. in der Grammatiktheorie bzw. Grammatikographie weiterhin erstrebenswert bleibt. Exemplarisch hierfür stehen Untersuchungen von Alternations- bzw. Variationsphänomenen, die im zurückliegenden Jahrzehnt unter Einsatz statistischer Modelle und Werkzeuge ein hochaktuelles linguistisches Forschungsthema geworden sind.[32]

---

[30] Menschliche Sprache als Gegenstand wissenschaftlicher Betrachtung ist in dem Sinne unendlich, dass keine abzählbare Liste aller real produzierten Sprachäußerungen denkbar ist. Bereits die Aufstellung einer erschöpfenden Übersicht über den Gesamtwortschatz einer Einzelsprache ist aus epistemologischen Erwägungen heraus unmöglich, da sich Sprache einerseits als primäres menschliches Kommunikationsmedium permanent weiterentwickelt und neue Ausdrücke ad hoc gebildet werden können, andererseits jedoch stets nur anhand eines begrenzten Sprachausschnitts beobachtet werden kann. Noch viel mehr gilt dies für (syntaktische) Kombinationen und Verwendungsweisen lexikalischer Zeichen in geschriebener oder gesprochener Sprache. In Abschnitt 2.4 wird deshalb dargestellt, wie Korpuslinguisten trotz solcher Beschränkungen der Datenquellen zu belastbaren Aussagen gelangen können.

[31] Vgl. Godfrey/Zampolli (1997, S. 382): „The fact that we still lack adequate linguistic resources for the majority of our languages can be attributed to: The tendency, predominant in the ’70s and the first half of the ’80s, to test linguistic hypotheses with small amounts of (allegedly) critical data, rather than to study extensively the variety of linguistic phenomena […]“.

[32] Vgl. hierzu z.B. Benor/Levy (2006); Bresnan et al. (2007); Bubenhofer et al. (2014); Gries (2003); Keibel et al. (2008); Konopka et al. (Hg.) (2011); Konopka/Fuß (2016); Müller (2007); Wöllstein et al. (2018).

In seinem wegweisenden Plädoyer für eine nachhaltige Kooperation zwischen Korpuslinguisten und traditionellen „Lehnstuhl-Linguisten“[33] schreibt Fillmore (1992, S. 35):

> I don’t think there can be corpora, however large, that contain information about all the areas of English lexicon and grammar that I want to explore […] [But] every corpus that I’ve had a chance to examine, however small, has taught me facts that I couldn’t imagine finding out about in any other way.

Konkrete Beispiele für den zweiten Teil dieses Plädoyers hinsichtlich des Alleinstellungsmerkmals von Sprachkorpora zur Aufdeckung bislang unbekannter bzw. primär introspektiv bewerteter Phänomene lassen sich mühelos finden. So nimmt sich Müller (2002) der Diskussion um die Vorfeldfähigkeit von Verbpartikeln an, d.h. der Frage, ob Partikelverben als untrennbare morphologische Objekte angesehen werden sollten. Ausgehend von einer umfassenden Darstellung des diesbezüglichen Forschungsdiskurses – der mehrheitlich eindeutig in die Richtung tendiert, die Vorfeldstellung von Partikeln speziell bei mehrteiligen Verbalkomplexen als nicht akzeptabel und mithin „ungrammatisch“ zu bewerten – überprüft er existierende Vorstellungen. Insbesondere weist er korpusgestützt[34] nach, dass Verbpartikeln unter gewissen Voraussetzungen durchaus vorangestellt werden können bzw. dass dies zumindest eindeutig belegbar ist („Vor hat er das jedenfalls.“). Dieser Nachweis von Korpusevidenz bietet eine fundierte Basis für ausführlichere empirische Analysen der (syntaktischen, semantischen usw.) Rahmenbedingungen. Korpusanalysen helfen dem Sprachforscher hier beim Erkennen, was es überhaupt bedeutet, wenn wir von „Vorfeldstellung“ oder „Verbalkomplex“ sprechen, und erweisen sich somit als sinnvolle Ergänzung intuitiver Urteile.

Dabei ist zu beachten, dass sich syntaktische und andere Phänomene in gesprochener Alltagssprache erkennbar von denjenigen unterscheiden können, die in geschriebenen Texten beobachtbar sind. Soll beispielsweise dialogisches sprachliches Handeln analysiert werden, so scheiden redaktionelle/redigierte Texte, Belletristik, Fachtexte etc. als Datenmaterial aus. Entsprechende Untersuchungen konzentrieren sich naheliegenderweise vielmehr auf „natürliche“

[33] Engl. *armchair linguists*; diese bewusst humoristische Bezeichnung für intuitiv-kompetenzbasiert arbeitende Sprachwissenschaftler diente seinerzeit als Ermunterung zur Einbeziehung empirischer Ansätze. Zur produktiven Überwindung der vorgeblichen Polarität zwischen Korpuslinguistik, Psycholinguistik, Kognitiver Linguistik etc. durch die Besinnung auf gemeinsame Erkenntnisinteressen und Untersuchungswerkzeuge vgl. z.B. Gries (2010).

[34] Die Recherchen fanden unter Nutzung der syntaktisch annotierten NEGRA-Baumbank (www.coli.uni-saarland.de/projects/sfb378/negra-corpus/negra-corpus.html) sowie des Deutschen Referenzkorpus (DeReKo) und dessen Frontends COSMAS (www.ids-mannheim.de/cosmas2/) statt.

Gesprächssituationen, die in Real-World-Kontexten entstanden sind. Vielversprechend erscheint in diesem Zusammenhang ferner die Einbeziehung computervermittelter Kommunikation (*computer-mediated communication, CMC*) zur Untersuchung von Sprachphänomenen im Grenzbereich zwischen Mündlichkeit und Schriftlichkeit. Internetbasierte Diskurse in E-Mails, Weblogs, Chat-Gruppen oder Online-Diskussionsforen lassen sich – unter Beachtung juristischer Einschränkungen durch Eigentums- oder Persönlichkeitsrechte – vergleichsweise unkompliziert digital archivieren, in weiterverarbeitbare Formate konvertieren und auswerten. CMC bezeichnet dabei nicht nur ein neuartiges Online-Medium, sondern nimmt Bezug auf den von üblichen Webseiten variierenden konzeptuellen – sprich: eher dialogisch ausgerichteten – Hintergrund. Insbesondere eröffnet CMC-Forschung damit Einblicke in den Sprachgebrauch in Situationen, die vergleichsweise nah an „echter" Mündlichkeit bzw. Face-to-face-Kommunikation liegen.[35]

Unabhängig von medialen Fragen lassen sich umfangreiche Untersuchungskorpora umso nutzbringender auswerten, je akkurater statistische Werkzeuge an deren spezifischen Rahmenbedingungen angepasst werden. Große Datenmengen sind grundsätzlich umso interessanter – im Sinne von: informativer – je zielgerichteter sie ausgewertet und je detaillierter die Ergebnisse dieser Auswertungen für eine wissenschaftliche Interpretation aufbereitet werden können. Die moderne Sprachwissenschaft betrachtet empirische Sprachanalysen und deren Vorarbeiten zu Recht als überaus anspruchsvolle Vorgänge, die informatische Unterstützung erfordern. Beispielsweise lassen sich bei unübersichtlichen Mengen an Sprachbelegen die Ergebnisse linguistisch motivierter Mustersuchen kaum noch ad hoc, etwa durch Konkordanzen, interpretieren. Hierfür ist die Anbindung statistischer Arbeitsumgebungen vorzuziehen, die komplexe Post-Retrieval-Szenarien unter Einbeziehung aller zweckmäßigen Metadaten ermöglichen und den Forschenden systematisch dabei unterstützen, empirische Beobachtung, statistische Analyse und linguistische Interpretation fachgerecht zu kombinieren. Eine maßgebliche Rolle spielen in diesem Zusammenhang auch bildgebende Werkzeuge. Visuelle Verfahren, mit deren Hilfe sich einzelne Aspekte eines Phänomens hervorheben oder für eine Fragestellung unerhebliche Komplexitäten ausblenden lassen, stellen ein nicht zu unterschätzendes Hilfsmittel bei der Beurteilung linguistischer Hypothesen dar (Bubenhofer/Kupietz (Hg.) 2018); vgl. hierzu auch die

[35] Vgl. z.B. Beißwenger (2018); Beißwenger et al. (2014); Beißwenger/Storrer (2008); Farzindar/Inkpen (2015); Gurevych/Zesch (2013); Herring (2010, 2011); Ogura/Nishimoto (2004). Zur Aufbereitung und Annotation deutschsprachiger CMC-Korpora, insbesondere hinsichtlich einer Nutzung des verbreiteten STTS-Tagsets, vgl. Bartz et al. (2014).

in Kapitel 6 vorgestellte Schnittstelle zwischen Korpusrecherche und Statistikwerkzeug.

Die empirische Erforschung sprachimmanenter Phänomenbereiche erfordert weiterhin mathematische Exaktheit und damit verbunden eine technisch-physische Integrität der Primärdaten. Insbesondere der Nachweis statistischer Regularitäten erfolgt in der Regel unter Beachtung strikter Gültigkeitsbedingungen, zu denen etwa die vollständige Einbeziehung intakter Forschungsobjekte zählt. So lassen sich auf Häufigkeitsverteilungen, Längenmessungen etc. basierende Gesetzmäßigkeiten der Textebene nicht unter Zuhilfenahme von Korpora nachweisen, die aus willkürlich kompilierten Belegsammlungen oder Textfragmenten bestehen. Zu diesen quantitativen Korrelationen zählen beispielsweise Verteilungsgesetze wie das Zipf-Mandelbrot-Gesetz über den Zusammenhang zwischen Häufigkeitsrang und Frequenz lexikalischer Einheiten, funktionale Gesetze wie das Menzerathsche Gesetz über den Zusammenhang der Länge eines sprachlichen Konstrukts und der Länge seiner unmittelbaren Komponenten, oder Entwicklungsgesetze wie das Piotrovskiy-Altmann-Gesetz zur Bestimmung der Verwendungsproportion sprachlicher Einheiten aus diachroner Perspektive.[36] Deren Erklärungskraft entfaltet sich erst bei der (ggf. separaten) Analyse zusammenhängender und ungekürzter Texte, da die relevanten Messgrößen (Wort-, Morphem- oder Phoneminventar, Satz- und Teilsatzlängen usw.) stets das Resultat individueller Textgenerierungsprozesse sind. Sinclair (2005) argumentiert dementsprechend gegen den Aufbau von Korpora aus Textfragmenten, weil Autorenpräferenzen für bestimmte Sprachkonstruktionen nicht zuletzt in Abhängigkeit von der jeweiligen Textposition getroffen werden, und fordert konsequent: „Samples of language for a corpus should wherever possible consist of entire documents or transcriptions of complete speech events, or should get as close to this target as possible“.

Neben der physischen kommt der logisch-formalen Integrität eine bedeutende Rolle zu. Um Korpusinhalte effektiv maschinell weiterverarbeiten bzw. auswerten zu können, müssen sie einheitlich formatiert sein, d.h. einem verbindlichen Textmodell entsprechen. Ziel ist dabei eine möglichst originalgetreue Speicherung der Rohdaten unter Erhalt ihrer ursprünglichen Struktur, typografischer Hervorhebungen usw. Zu diesem Zweck werden zumeist auf XML

[36] Vgl. hierzu die grundlegenden Überlegungen in Köhler (1986, 2005) sowie den Überblick über weiterführende Arbeiten in Köhler/Altmann/Piotrovskii (Hg.) (2005). Biemann (2007) stellt in diesem Zusammenhang ein zufallsbasiertes Textgenerierungsmodell für die Erforschung statistischer Sprachregularitäten vor.

(Extensible Markup Language) basierende Formate[37] eingesetzt, beispielsweise der Corpus Encoding Standard for XML (XCES),[38] die Standards der Text Encoding Initiative (TEI)[39] oder das auf beiden Spezifikationen aufbauende Textmodell des Instituts für Deutsche Sprache (IDS).[40] XML-kodierte Inhalte erfüllen dabei die Kriterien der Wohlgeformtheit (d.h. der konsistenten Verschachtelung aller enthaltenen Elemente) sowie der Gültigkeit (d.h. der Einhaltung vordefinierter struktureller Regeln). Alternativ können Rohdaten auch rein datenbankbasiert vorgehalten werden; vgl. hierzu z.B. die Dokumentation der Leipzig Corpora Collection (LCC) in (Biemann et al. 2007).

2.1.1 Umfang und Zusammensetzung von Sprachkorpora

Betrachten wir digitale Sprachkorpora als zweckmäßiges Instrumentarium für die Untersuchung quantifizierbarer Eigenschaften von Sprachverhalten, so stellt sich unmittelbar die Frage nach der erforderlichen Größe (Quantität) der Primärdaten – also der erfassten Rohtexte bzw. Ton-/Videoaufnahmen – sowie der Art (Qualität) der darüber hinaus einzubeziehenden Sekundärdaten. Hinsichtlich des quantitativen Aspekts liegt die begründete Vermutung nahe, dass methodisch valide empirische Aussagen zur intendierten Grundgesamtheit (also dem tatsächlichen Sprachgebrauch) von möglichst umfangreichen Sprachkorpora profitieren. Hunston (2008, S. 160) argumentiert entsprechend: „If a range of topics and writers is to be included in the corpus, it must be of a sufficient size to allow this. Thus, representativeness and size are connected".[41] Insbesondere niedrigfrequente Phänomene, wie z.B. erste Erscheinungen bei Sprachänderungen, morphologische Prozesse, syntaktische Innovationen, Neologismen etc., lassen sich erfahrungsgemäß erst unter Zuhilfenahme sehr großer Sprachdatensammlungen bewerten. Dies gilt in besonderem Maße, wenn nicht nur nach Einzelphänomenen, sondern nach tendenziell nochmals selteneren Phänomenkombinationen (Wortpaaren, Lemma-Wortart-Paaren, Redewendungen, spezifischen syntaktischen Grup-

[37] Hinsichtlich der Eignung von XML-Technologien zur Strukturierung linguistisch relevanter Informationsquellen siehe z.B. Farrar (2006); Lobin (2000); Witt (2002).

[38] Siehe hierzu einführend Ide/Bonhomme/Romary (2000) sowie http://xces.org.

[39] Mit der aktuellen Version TEI-P5 (Burnard/Bauman (Hg.) 2013) werden beispielsweise die DWDS-Korpora (Digitales Wörterbuch der deutschen Sprache des 20. Jahrhunderts) ausgezeichnet; vgl. Lemnitzer et al. (2013).

[40] Die aktuelle Konzeption des Textmodells für das Deutsche Referenzkorpus (DeReKo) beschreiben Lüngen/Sperberg-McQueen (2012); siehe auch www.ids-mannheim.de/kl/projekte/korpora/textmodell.html.

[41] Hier gilt es anzumerken, dass das Konzept der Repräsentativität im Sinne von „repräsentativ für eine bestimmte (vordefinierte) Fragestellung" zu verstehen ist (s.u.).

pierungen etc.) recherchiert werden soll; diesbezügliche Implikationen für die gegenstandsbedingt notorisch mit komplexen Mustern arbeitende Grammatikforschung diskutieren z.B. Bubenhofer et al. (2014), Dürscheid et al. (2011), Štícha (2008) oder Strecker (2011).

Die Gefahr bei der Verwendung zu kleiner Datensammlungen liegt gleichermaßen in der Überbewertung von Zufallsfunden wie im Übersehen seltener Phänomene. Geyken (2011, S. 123) definiert als Mindestgrenze für die Erstellung aussagekräftiger Wortprofile auf Basis von Lemma-Wortart-Paaren eine Belegmenge von 500-1.000 Fundstellen für Lemmata. Am Beispiel der DWDS-Korpora folgert er daraus, dass Korpusgrößen von 100 Millionen laufenden Wortformen für verlässliche statistische Aussagen nicht ausreichen. Ein (phänomenbezogen einfacheres) Beispiel aus dem Online-Wortschatz-Informationssystem Deutsch (OWID) untermauert diese Größenordnung: Abbildung 2 dokumentiert die zeitliche Verteilung der Gebrauchshäufigkeiten des Neologismus „casten“.[42] Bis Ende der Neunzigerjahre liegt die ermittelte relative Frequenz des Wortes im Deutschen bei unter 0,05 Instanzen pro Million Wörter. Korpussammlungen mit weniger als 20 Millionen laufenden Wortformen kämen bei diesem Beispiel als sichere Belegquelle rein rechnerisch keinesfalls in Betracht. Andererseits ließe sich daraus auch keine negative Evidenz ableiten, d.h. es könnte nicht im Umkehrschluss gefolgert werden, dass „casten“ im deutschen Sprachgebrauch der Neunzigerjahre keine Rolle gespielt hätte. Noch viel weniger ließen sich mit derartigen Korpusvolumen eine verlässliche Entwicklung der Verwendung nachzeichnen oder soziologisch-demografische Muster unter Zuhilfenahme außersprachlicher Sekundärdaten aufdecken.[43]

Entscheidend für das Sammeln und Auswerten von Fakten ist demnach die Verfügbarkeit einer quantitativ angemessenen Datengrundlage:[44] „More data are better data“. Dieser in (Church/Mercer 1993) formulierte Anspruch an em-

[42] Quelle: Online-Wortschatz-Informationssystem Deutsch (OWID); www.owid.de. Das zugrunde liegende Untersuchungskorpus basiert auf Zeitungstexten des Deutschen Referenzkorpus (DeReKo)/Archiv der Korpora geschriebener Gegenwartssprache 2014-II (Release vom 11.09.2014) des IDS Mannheim.

[43] Vgl. hierzu auch die Untersuchungen von Mark Davies unter https://corpus.byu.edu/compare.asp sowie verschiedene Beiträge in Gippert/Gehrke (2015) zur Bedeutung von Korpusgrößen für die Sprachforschung am Beispiel historischer Korpora. Zur erforderlichen Stichprobengröße für linguistische Untersuchungen vgl. z.B. Gries (2008b).

[44] Neben der Korpusgröße können bei derartigen Untersuchungen aus statistischer Perspektive grundsätzlich weitere Parameter für die Güte der getroffenen Aussagen relevant sein, beispielsweise die verwendete Stichprobengröße (*sample size*) bei zahlenmäßig umfangreicheren Treffermengen oder die Methode der Stichprobenziehung; vgl. hierzu auch Abschnitt 2.4. Formeln und Berechnungen zur Stichprobengröße finden sich z.B. in Gries (2008b, S. 130 ff.).

pirische Sprachuntersuchungen ist mittlerweile zu einem grundlegenden Leitmotiv der Korpuslinguistik geworden: „Die Herausforderung besteht nun darin, herauszufinden, inwieweit es möglich ist, die Erkenntnisse, die man durch die Analyse eines Sprachausschnitts gewonnen hat, auf andere Ausschnitte zu extrapolieren. Je größer und vielfältiger die Korpora sind [...] desto mehr können sie diese verallgemeinernde Funktion erfüllen.“ (Belica/Steyer 2008, S. 10). Der Trend bei linguistisch motivierten Datensammlungen geht eindeutig hin zu immer eindrucksvolleren Tokenzahlen.

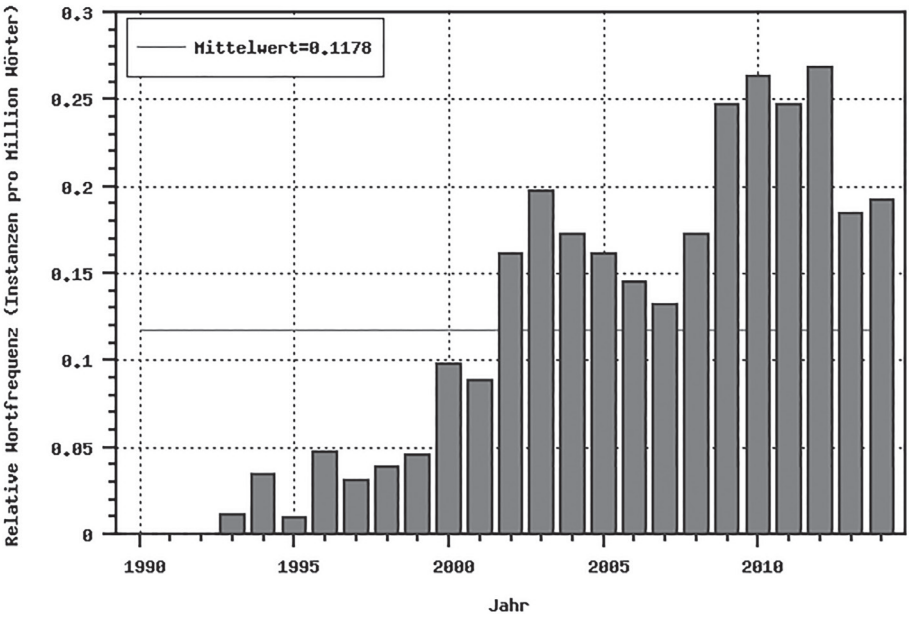


Abb. 2: Zeitliche Verteilung der Gebrauchshäufigkeiten des Neologismus „casten“

Der positive Zusammenhang zwischen ansteigenden Korpusgrößen für Retrieval-, Test- oder Trainingszwecke sowie ansteigender Güte der daraus gewonnenen empirisch-statistischen Erkenntnisse erscheint grundsätzlich stichhaltig. Daneben zeigen neuere Studien allerdings auch, dass die bloße Zunahme an Primärdaten nicht per se und für alle linguistischen Fragestellungen bzw. Forschungsszenarien die Aussagequalität erhöht. Insbesondere angesichts der „Multifunktionalität“ (McEnery 2003) umfangreicher Korpora, also deren intendierten Nutzung für ganz unterschiedlich ausgerichtete Fragestellungen, gilt es die Qualität und Angemessenheit der gesammelten Daten vor jeder Auswertung kritisch zu reflektieren und ggf. die auszuwertenden Stichproben einzuschränken. Die Zufallsauswahl konkurriert an dieser Stelle mit der

willkürlichen sowie der bewussten Stichprobenauswahl. In Sinne randomisierter Korpusausschnitte argumentieren z.B. Gries (2008b) oder Zesch/Gurevych (2010); eine vorteilhafte Evaluation sehr großer Textsammlungen für linguistisch motivierte Untersuchungen findet sich z.B. in Banko/Brill (2001).[45]

Für das Korpusdesign ist es aus methodisch-methodologischer Perspektive hilfreich, zwischen den kontrovers diskutierten Konzepten der Repräsentativität und der Ausgewogenheit zu unterscheiden. Beide wurden in der Vergangenheit als potenzielle Grundlage für Komposition und Organisation von Korpussammlungen herangezogen, wobei sich das Ideal der Repräsentativität nur begrenzt umsetzen lässt.[46] Wenn Sprache bzw. Sprachverhalten nämlich aufgrund ihres inhärent unendlichen Charakters als Grundgesamtheit gar nicht fassbar sind, dann können selbst die umfangreichsten Sprachkorpora niemals universell repräsentativ für eine Gesamtsprache sein, sondern bestenfalls im Hinblick auf spezielle Fragestellungen und Dimensionen des Sprachgebrauchs. Stattdessen bietet es sich beim Aufbau linguistischer Untersuchungskorpora an, die heterogene sprachliche Realität durch Einbeziehung verschiedenartiger – und ihrerseits dann wiederum möglichst umfangreicher – Ausschnitte zumindest tendenziell ausgewogen abzubilden. Das bedeutet konkret, unterschiedliche Manifestationsformen (z.B. hinsichtlich Medium, Region, Fachdomäne, Zeitraum etc.) der zu untersuchenden Sprache gebührend zu berücksichtigen, das Korpus also annäherungsweise als eine Art Sprach-Mikrokosmos mit breiter Streuung anzulegen. Solche nicht einseitig auf Zeitungstexten, Belletristik oder Webseiten aufgebauten Datensammlungen heißen in der Korpuslinguistik auch „geschichtet" oder „strati-

[45] Dass unangemessene Methoden der Stichprobenziehung die grundsätzlichen Vorteile sehr großer Stichproben unterlaufen können, ist abseits des linguistischen Gegenstandsbereichs seit Jahrzehnten wohlbekannt. So erwies sich bereits 1936 eine im US-amerikanischen Präsidentschaftswahlkampf (Landon vs. Roosevelt) durchgeführte Umfrage unter mehr als zwei Millionen Wahlberechtigten als spektakulär unzutreffend, während die Analyse einer mit 5.000 Probanden vergleichsweise kleinen, aber repräsentativen Stichprobe verlässlich den späteren Gewinner vorhersagte.

[46] Biber (1993a, S. 243) definiert Repräsentativität als Abdeckung sämtlicher für spätere Untersuchungen relevanten Variabilitätsfaktoren (*sampling strata*), also beispielsweise Texttyp, Thema oder situationelle Parameter bis hin zu linguistischen Faktoren (Wortklasse, Satzglied etc.): „Representativeness refers to the extent to which a sample includes the full range of variability in a population". Eine Abbildung exakter Proportionen erscheint bei multifunktionalen Korpora allerdings kaum realistisch: "The bottom-line in corpus design, however, is that the parameters of a fully representative corpus cannot be determined at the outset". (Biber 1993a, S. 255f.). Kennedy (2008, S. 62) fasst in diesem Sinne die zentrale Kritik an der Idee einer Repräsentativität von Sprachstichproben zusammen: „It is not easy to be confident that a sample of texts can be thoroughly representative of all possible genres or even of a particular genre or subject field or topic".

fiziert". Bickel et al. (2009) sprechen in diesem Zusammenhang von der Abdeckung einer „reduzierten Grundgesamtheit" durch Erfüllung eines „Kriterienkatalogs für gewichtete Vielseitigkeit".

Vor diesem Hintergrund stellt die angemessene Proportionierung der jeweiligen Anteile (Strata) für jede Korpuszusammenstellung eine wissenschaftsmethodische Herausforderung dar und lässt sich kaum allgemeinverbindlich beziffern. In Abhängigkeit vom konkreten Untersuchungsinteresse und den damit verbundenen inner- bzw. außersprachlichen Implikationen gilt es dabei gleichermaßen, sämtliche eventuell relevanten Sprachgebrauchsbereiche in einer Typologisierungsbasis zu erfassen, ungeeignete Sprachdatentypen auszuklammern, sowie Über- bzw. Untergewichtungen der konstituierenden Korpusquellen zu vermeiden.[47] Soll beispielsweise ein Korpus zur Analyse der Verständlichkeit schriftsprachlicher technischer Instruktionen konstituiert werden, kann auf Verschriftlichungen frei formulierter Reden und Gespräche ebenso verzichtet werden wie auf disparate Textsorten (Pressetexte, narrative Abhandlungen etc.). Zur Abdeckung der inhaltlich-thematischen Anforderungen – in dem genannten Beispiel also der Domäne „Technik" – sollte ein breites Spektrum an Montage-, Bedienungs- und Reparaturanleitungen für unterschiedliche Einsatzbereiche (also z.B. aus der Informationstechnik, Bautechnik, Verkehrstechnik etc.) integriert werden, wohingegen Kochrezepte oder Gesundheitsbücher entfallen dürfen. Gegebenenfalls helfen quantifizierende Übersichten wie Auflagezahlen, Bestsellerlisten, Ausleih- und Zugriffsstatistiken o.Ä. bei der Beurteilung von Popularität und Reichweite und damit bei der Entscheidung über die Aufnahme in ein Untersuchungskorpus bzw. über die interne Gewichtung.[48] Um die inhärente stilistisch-grammatikalische Merkmalsvielfalt der gespeicherten Textinhalte – d.h. der Ergebnisse individueller Sprachhandlungen – nicht durch Idiosynkrasien zu verzerren, gilt es

[47] Erfahrungsgemäß kommen bei der Ausarbeitung von Merkmalssystematiken für die Textsorten-Klassifikation verschiedenartige textinterne bzw. textexterne Kriterien, Faktoren und Methoden zum Einsatz; zur Einführung siehe z.B. Adamzik (Hg.) (2000), Leech (2006) oder Sinclair (2005). Gleiches gilt für die letztendliche Gewichtung der einzubeziehenden Sprachbereiche (vgl. z.B. Biber 1993a; Evert 2006 und Nelson 2012), bei der zudem grundsätzlich unterschiedliche Zähleinheiten zur Auswahl stehen (Texte, Textseiten, Sätze, Wörter etc.). Gries (2008b, S. 31) spricht in diesem Zusammenhang von einem „theoretischen Ideal, weil wir die Teile und ihre Proportionen in der Grundgesamtheit nicht kennen". Zur Adäquatheit ausgewogener Korpora sowie exemplarischen Ansätzen (z.B. dem Einsatz von „Life-Logging" zur Quantifizierung von Korpusbestandteilen auf der Basis empirischer Daten zu Sprachproduktion und -gebrauch) vgl. auch z.B. Bubenhofer et al. (2014) und Wattam et al. (2013).

[48] Dabei gilt es zu beachten, dass bei der Einbeziehung von Auflagestatistiken o.Ä. die Rezipientenzahl als (ggf. schwer verifizierbarer) Einflussfaktor auf zukünftiges Sprachverhalten genutzt wird. Bestsellerautoren oder auflagenstarken Zeitschriften wird somit ein prägenderer Einfluss als Nischenpublikationen zugeschrieben.

weiterhin ein möglichst breites Autorenspektrum zu berücksichtigen. Möchte man mediale, geografische, historische oder soziokulturelle Entstehungsbedingungen in die Auswertung einbeziehen, hat die Typologisierungsbasis auch vergleichbar dimensionierte Sprachdaten für sämtliche Ausprägungen der entsprechenden Metadaten zu beinhalten – die Verwendung überregionaler Tageszeitungstexte zur Beschreibung lokaler Sprachvariationen etwa dürfte in diesem Sinne als wenig aussichtsreicher Ansatz gelten.

2.1.2 Sekundärdaten

Der Blick auf umfangreiche heterogene Datenbestände jeglicher Art findet häufig maßgeblich durch die „Brille" der Sekundärdaten statt; Sprachkorpora bilden in dieser Hinsicht keine Ausnahme. Für linguistische Untersuchungen aufbereitete Korpussammlungen grenzen sich von reinen Textarchiven[49] dadurch ab, dass sie über die primären Sprachinhalte (Texte bzw. transkribierte Ton-oder Videoaufnahmen) hinaus potenziell zwei weitere Informationstypen enthalten. Diese lassen sich einteilen in deskriptiv-analytische Annotationen linguistisch relevanter Merkmale der Sprachäußerungen sowie inner- und außersprachliche Metadaten auf Text- bzw. Korpusebene.

Annotationen basieren auf einer vorhergehenden Segmentierung der Primärdaten in kleinere sprachliche Einheiten, also in Sätze, Phrasen, Wörter, Morpheme, Phoneme etc., denen die linguistische Zusatzinformation zugeordnet wird. Eine initiale Zerlegung besteht zumeist in einer Isolierung der Wortformen (engl. *token*) und wird angesichts des enormen Umfangs moderner Korpora unter Ausnutzung computerlinguistischer Methoden vorzugsweise maschinell durchgeführt. Hierfür sind wiederum verschiedene Vorverarbeitungsschritte durchzuführen: Bereinigung der Rohdaten von Steuerzeichen (z.B. Zeilenumbrüchen), Behandlung von Trennungsstrichen sowie sonstiger Interpunktions- und Sonderzeichen, temporäre Entfernung struktureller Auszeichnungen (Markup) oder typografischer Hervorhebungen (z.B. Sperrsatz) usw. Problematisch gestaltet sich erfahrungsgemäß die Identifikation und ggf. Normalisierung ungewöhnlicher Neubildungen (z.B. Onomatopoetika) sowie komplexer Token, also z.B. von Bindestrichkomposita, Internet-/Mailadressen, Abkürzungen oder numerischen Darstellungsvarianten.[50] Hier ar-

[49] Als Beispiele für den deutschsprachigen Raum seien das Projekt Gutenberg-DE (mit derzeit ca. 6.000 vorrangig literarischen Werken; vgl. http://gutenberg.spiegel.de) sowie die deutsche Wikisource (mit derzeit über 30.000 urheberrechtsfreien Werken unterschiedlichster Herkunft und Thematik, die oft anderweitig nicht im Internet zugänglich sind; vgl. http://de.wikisource.org) genannt.

[50] Beispiele für Fälle, in denen (nicht nur maschinelle) Analysen mehrere plausible Vorschläge für eine Tokenisierung liefern können, sind beispielsweise *(Dienst-)Personal, Jeanne d'Arc,*

beiten einschlägige Algorithmen mit wissensbasierten Hilfsmitteln, etwa Abkürzungslisten, regelbasierten Ansätzen oder statistischen Lernverfahren unter Nutzung eines bereits (manuell) korrekt tokenisierten Referenzkorpus (engl. *gold standard*). Ähnliches gilt für die Erkennung von Satzgrenzen. Diese wird dadurch kompliziert, dass eigentlich satztrennende Interpunktionszeichen auch zu anderen Zwecken eingesetzt werden können, etwa in Abkürzungen oder zum Ausdruck von Bewertungen und Stimmungen (Emoticons).

Nach der erfolgreichen Tokenisierung folgt in der korpuslinguistischen Verarbeitungskette üblicherweise die Anreicherung um aussagekräftige Annotationen (engl. *tagging*) derjenigen Merkmale, die für spätere sprachwissenschaftliche Fragestellungen und linguistisch informierte Phänomenanalysen potenziell relevant sind. Hierzu zählen beispielsweise:

- (morpho-)syntaktische Strukturen, etwa zur Aufdeckung grammatischer Regularitäten und Variationen, von Wortbildungsprozessen oder von Thema-Rhema-Beziehungen;
- semantische Beschreibungen und Relationen, etwa zur Analyse von Wortschatzentwicklung, zur Differenzierung von Verwendungskontexten oder zum Aufbau lexikalisch-semantischer Wortnetze;
- phonetische Merkmale, etwa für die Dialektforschung oder zur Unterscheidung unterschiedlicher Lesarten eines Lexems anhand der Intonation;
- temporale oder sprecherspezifische Referenzen, Sprechakte, rhetorische Relationen etc., z.B. für die Analyse von Diskursphänomenen.

Auch hier kommen angesichts der zu verarbeitetenden Datenvolumen vorrangig maschinelle und semi-maschinelle Verfahren zum Einsatz. Diese erledigen ihre Aufgaben wahlweise unter Verwendung regel- oder lexikonbasierter, statistischer oder hybrider Methoden. In jedem Fall setzen sie – notabene ebenso wie menschliche Annotatoren – einen gewissen theoretischen Rahmen voraus; hierzu gehören beispielsweise Vorgaben zur Abgrenzung von Wortklassen oder zum Inventar semantischer Relationen. Dabei gilt es aus methodischer Perspektive, sich der in Tognini-Bonelli (2001, S. 17) dargestellten Unterscheidung zwischen korpusbasierten (*corpus-based*) und korpusgeleiteten (*corpus-driven*) Untersuchungen bewusst zu sein. Für beide Ansätze existieren wohlbegründete wissenschaftspraktische bzw. erkenntnistheoretische Motivationen; beide Ansätze verfolgen das Ziel, durch explorative Korpusanalyse empirisch fundierte Erkenntnisse über natürliche Sprache zu gewinnen und Regularitäten zu beschreiben. Der Hauptunterschied besteht darin, dass die korpusbasierte Arbeitsweise vorab aufgestellte Hypothesen und Theorien ein-

CDU/CSU, 1.30h oder *tina-taler@webmail*; vgl. hierzu z.B. Mikheev (2002).

bezieht und diese ggf. statistisch überprüft. Korpusgeleitete Untersuchungen dagegen folgen einem strikt induktiven Paradigma. Sie bauen – soweit irgendwie möglich – ohne vorgegebenen theoretischen Rahmen „in the presence of the evidence" allein auf unmittelbar beobachtbaren Phänomenen auf und leiten allein davon generalisierende Sprachgebrauchsregeln sowie weiterführende theoretische Konstrukte induktiv ab. Nicht quantitativ begründete Annahmen und Theorien, intuitive Urteile usw. dürfen erst zu einem möglichst späten Zeitpunkt in die Untersuchungen einfließen; Sinclair (1991) spricht in diesem Zusammenhang vom Prinzip der minimalen Annahme (*minimum assumption*). Auf diese Weise soll vermieden werden, dass Sprachanalysen Kategorisierungen und Regularitäten zutage fördern, die bereits in die Konzeption des Untersuchungsdesigns eingeflossen sind.[51]

Entsprechende Anforderungen haben einen unmittelbaren Einfluss auf das jeweils verwendbare Datenmaterial. Streng genommen widerspricht etwa die auf diversen theoretisch-methodischen Vorannahmen aufbauende Annotation linguistischer Beschreibungsebenen der für korpusgeleitete Forschung zentralen Maxime, Sprachdaten vorbehaltslos und theorieneutral zu betrachten. Ein Tagger, der vorab mit Wortklassen und anderen grammatischen Kategorien trainiert wurde, arbeitet eben notwendigerweise nicht strikt induktiv. Ein alternativer, korpusgeleiteter Ansatz zur Aufdeckung grammatischer Regularitäten findet sich in „Approaching Grammar" (Keibel et al. 2011). Unter weitestgehendem Verzicht auf einen externen sprachwissenschaftlichen Überbau und allein durch die statistische Analyse von Kollokationen auf Ebene der Primärdaten ermittelt er für die untersuchten Sprachdaten Kategorien mit identischen Verwendungsmerkmalen. Diese können zwar zumeist nicht als allgemeingültige (d.h. sprachweite) Wortklassen gelten, stehen im betreffenden Korpussample aber prototypisch für traditionelle Wortklassen und deren morphosyntaktische Eigenschaften. Gegenstand zukünftiger Forschungsarbeit bleibt, ob und wie sich ein solcher vorannahmsloser Ansatz auf andere linguistische Beschreibungsebenen übertragen lässt. Auf jeden Fall unterstreicht das Beispiel die Verpflichtung linguistischer Untersuchungen, die Problematik a priori postulierter Kategorisierungen anzuerkennen und im Kontext der Korpusannotation die jeweils eingeflossenen methodischen Vorannahmen in spätere Bewertungen einzubeziehen. Praktische Abhilfe zur Vermeidung einseitiger theoretischer Einflüsse kann im Übrigen bis zu einer gewissen Grenze

[51] Bubenhofer (2009, S. 100) fasst diese Kritik folgendermaßen zusammen: „Korpora mit ganz bestimmten Theorien als Prämissen zu befragen, birgt die Gefahr, in den Daten nur die Strukturen zu finden, die mit der Theorie kompatibel sind und blind gegenüber Evidenzen zu sein, die quer zu einer Theorie stehen." Dass beide Ansätze auch nutzbringend komplementär eingesetzt werden können, demonstriert Storjohann (2012a).

auch durch die parallele Annotation derselben Ebene unter Zuhilfenahme verschiedener Ansätze und Werkzeuge sowie einen anschließenden systematischen Vergleich der Annotationsresultate hinsichtlich konvergierender bzw. divergierender Beurteilungen geschaffen werden.[52]

Unabhängig von diesen grundlegenden methodischen Aspekten lässt sich konstatieren: Computerlinguistisch etabliert und im praktischen Einsatz weit verbreitet sind maschinelle Verfahren für die Erkennung der Grundformen (Lemmata) sowie für die Zuweisung von Wortklasseninformationen (engl. *part of speech*, deshalb auch *POS-Tagging* genannt) und morphosyntaktischen Kategorien (Kasus, Tempus, Numerus, Genus etc.). Softwarebasierte Ansätze annotieren dabei mittlerweile bis zu ca. 98% der behandelten Wortformen korrekt entsprechend der Vorgaben – und dabei weitestgehend konsistent, einer elementaren Vorraussetzung für die effiziente Weiterverarbeitung. Die anspruchsvolle Analyse syntaktischer Strukturen (engl. *parsing*), also die Annotation von phrasalen Kategorien (NP, VP, PP etc.) sowie anderen Konstituenz- und Dependenzrelationen kann ebenfalls unter Zuhilfenahme computerlinguistischer Programme durchgeführt werden. Die eingesetzten Syntaxparser verwenden je nach theoretischem Ansatz spezifische Strategien und Algorithmen (z.B. deterministisch vs. nicht-deterministisch, kontextsensitiv vs. kontextfrei, top-down vs. bottom-up, breadth-first vs. depth-first etc.), deren Fehlerraten in Abhängigkeit von Qualität und Komplexität der Primärdaten zwischen ca. 5 und 20% variieren. Eine manuelle Nachbearbeitung bzw. die Einbeziehung interaktiver Entscheidungsprozesse durch semi-automatische Verfahren sind deshalb in bestimmten Fällen angezeigt und kommen insbesondere beim Aufbau von Baumbanken (engl. *tree banks*) zum Einsatz. Baumbanken als Sonderformen linguistischer Korpora spezialisieren sich primär auf die Analyse von Syntaxbäumen und erreichen dabei – bei im Vergleich zu aktuellen Referenz- oder Monitorkorpora zumeist deutlich kleineren Datenmengen – beachtliche Gütequoten. Ebenfalls teilautomatisiert – wenn auch mit z.T. erheblich stärkerer manueller Unterstützung – lassen sich semantische Informationen (Disambiguierung unterschiedlicher Lesarten, Zuweisung semantischer Rollen und Relationen etc.), Diskursphänomene (Koreferenzen, Kospezifikationen etc.) oder phonetisch-phonologische Eigenschaften (Intonation, prosodische bzw. suprasegmentale Merkmale etc.) annotieren.[53]

[52] Vgl. z.B. die in Belica et al. (2011) beschriebenen Studien zum parallelen Einsatz unterschiedlicher Tagger, die Bewertung der Verlässlichkeit und Brauchbarkeit grammatischer Annotationen in Bubenhofer et al. (2014, S. 110 ff.) sowie das in Zesch/Horsmann (2016) präsentierte anpassbare Tagging-Framework FlexTag.

[53] Eine Übersicht über forschungsrelevante Auszeichnungssprachen, Annotationsebenen und die zugehörige Projektliteratur bieten z.B. Leech (2005), Stührenberg (2012) und Zinsmeister (2010); für das Potsdamer Referenzkorpus siehe Stede (Hg.) (2016). Zu Problemen bei der au-

Die unterschiedlichen Beschreibungsebenen werden in sogenannten Tagsets definiert und dokumentiert. Naturgemäß existiert hierbei eine gewisse Bandbreite hinsichtlich der Beschaffenheit und Granularität des Beschreibungsinventars. Für den deutschsprachigen Bereich etabliert ist das „Stuttgart-Tübingen TagSet" (STTS).[54] Es unterscheidet 11 Hauptwortarten und insgesamt 54 Wortartentags, während die englischsprachige Penn Treebank mit 36 und das Brown-Korpus mit über 80 POS-Tags arbeiten. Tagsets dokumentieren stets einen konsistenten Bestand der für die Auszeichnung zur Verfügung stehenden Elemente (engl. *tags*), spezifizieren aber nicht deren technische Realisierung in der physischen Datenstruktur.

Für diese Realisierung wird zwischen den beiden Varianten „inline" und „standoff" unterschieden. Im ersten Fall finden sich die hinzugefügten Annotationsdaten in die Originaltexte eingebettet, was einerseits eine unmittelbare Begutachtung durch menschliche Nutzer zumindest grundsätzlich ermöglicht. Andererseits werden die Originaltexte, die als Referenz unbedingt erhalten oder zumindest zuverlässig rekonstruierbar bleiben müssen, dabei zwangsläufig manipuliert. Außerdem leidet die Lesbarkeit inline annotierter Texte spätestens bei komplexeren Tagsets und der parallelen Auszeichnung verschiedener Annotationsebenen (engl. *multi-layer annotations* bzw. *multi-level annotations*) erheblich. In vielen Fällen verhindern konkurrierende Hierarchien und Überschneidungen den wohlgeformten Einsatz von Inline-Markup überhaupt. Ein Beispiel hierfür ist die gleichzeitige Annotation von Silben- und Morphemgrenzen auf Tokenlevel (vgl. Tab. 1), bei der die Überschneidungen zwischen beiden Beschreibungsebenen eine XML-konforme Speicherung sowie ein standardisiertes maschinelles Postprocessing beeinträchtigen. Durusau/O'Donnell (2002) identifizieren bereits für zwei konkurrierende Annotationslayer eine Zahl von 13 möglichen Beziehungstypen zwischen Elementen der beiden Ebenen („klassische" Überlappung, gemeinsame Start- bzw. Endposition, Identität, Inklusion usw.). Lüngen/Witt (2008) generalisieren diesen Ansatz für drei und mehr Annotationsebenen und konstatieren eine „kombinatorische Explosion" der potenziell abzufragenden Elementbeziehungen (409 bei drei Annotationsebenen, 23.917 bei vier Annotationsebenen usw.).

tomatischen Lesartendisambiguierung und der daraus resultierenden „semantischen Blindheit" vieler Korpusabfragesysteme vgl. Beißwenger/Storrer (2011).

[54] STTS definiert ein kleines sowie ein großes Tagset, letzteres umfasst neben Wortklassen auch weitere morphologische und lexikalische Kategorien; vgl. z.B. Telljohann et al. (2013) und Schiller et al. (1999). STTS wird u.a. von der freien Tagging-Software *TreeTagger* verwendet.

Als Lösung bietet sich hier die logische und technische Trennung von Primär- und Sekundärdaten in Form einer „standoff"-Annotation an; in der Praxis haben sich auch Mischformen bewährt.[55] Dabei werden in den Originaltexten maximal grundlegende strukturelle Informationen wie Text-, Absatz- oder Satzgrenzen ausgezeichnet, während darauf operierende linguistische Sekundärdaten separiert vorgehalten werden. XML unterstützt zu diesem Zweck die flankierenden Standards XLink und XPointer. Diese erlauben durch die Referenz auf Identitätsattribute externe Verweise auf beliebige Segmente in den Primärdaten sowie darauf aufbauend die gegebenenfalls mehrfache Zuweisung ergänzender Angaben.

Token	**Nicht wohlgeformte „inline"-Annotation im Primärtext**
Ämter	`<token lemma="Amt"><morphem><silbe>Äm</silbe>` `<silbe>t</morphem><morphem>er</morphem></silbe></token>`
lustig	`<token lemma="lustig"><morphem><silbe>lus</silbe>` `<silbe>t</morphem><morphem>ig</morphem></silbe></token>`

Tab. 1: Exemplarische Inline-Annotation von Silben und Morphemen

Tabelle 2 illustriert, wie durch diese Aufhebung der physischen Einheit von Originaltext und Annotation Überschneidungen zwischen Silben- und Morphemgrenzen XML-konform realisiert werden können. Im Primärtext wurden dabei aus Gründen der Anschaulichkeit die Tokengrenzen markiert, was grundsätzlich ebenso ausgelagert werden könnte wie die Kennzeichnung der feingranularen Zusatzinformationen.

[55] Vgl. z.B. die Diskussionen in Naumann (2003) und Burghardt/Wolff (2009) sowie die Empfehlungen der Text Encoding Initiative (TEI) zur Anbindung multipler Standoff-Annotationen in Burnard/Bauman (Hg.) (2013). Eine Darstellung konkurrierender Standoff-Annotationen anhand realer Korpusbelege findet sich in Abschnitt 2.4.

XML- Primärtext	„standoff"-Annotation in separater Datei
`<token id="t1">Ämter</token>`	`<token xlink:href="xptr(substring (//token[t1]))" lemma="Amt"> <morphem id="m1" xlink:href="xptr(substring (//token[t1]/text(),1,3)"/> <morphem id="m2" xlink:href="xptr(substring (//token[t1]/text(),4,5)"/> <silbe id="s1" xlink:href="xptr(substring (//token[t1]/text(),1,2)"/> <silbe id="s2" xlink:href="xptr(substring (//token[t1]/text(),3,5)"/> </token>`
`<token id="t2">lustig</token>`	`<token xlink:href="xptr(substring (//token[t2]))" lemma="lustig"> <morphem id="m3" xlink:href="xptr(substring (//token[t2]/text(),1,4)"/> <morphem id="m4" xlink:href="xptr(substring (//token[t2]/text(),5,6)"/> <silbe id="s3" xlink:href="xptr(substring (//token[t2]/text(),1,3)"/> <silbe id="s2" xlink:href="xptr(substring (//token[t2]/text(),4,6)"/> </token>`

Tab. 2: Exemplarische Standoff-Annotation von Silben und Morphemen

Metadaten sind auf Korpus- oder Textebene angesiedelte Hinzufügungen zu den Primär- und Annotationsdaten, die außersprachliche Hintergrundinformationen über deren Inhalt oder Entstehungskontext strukturiert kodieren. Damit sind sie eine elementare Voraussetzung für langfristige Wiederverwendbarkeit und multifunktionale Einsatzszenarien. Grundsätzlich gilt wieder die quantitative Optimierungsmaxime: Je mehr wir über die realen Einsatzfaktoren von Sprachäußerungen zuverlässig in Erfahrung bringen, desto detaillierter können wir anschließend regelhafte Interdependenzen ergründen – „more *meta*-data are better *meta*-data".[56]

[56] Aus informatischer Perspektive handelt es sich zweifellos auch bei Annotationen um Metadaten; ebenso werden Metadaten oft als Inline-Annotationen in Korpustexte eingefügt. Für eine unmissverständliche gegenseitige Abgrenzung dieser beiden in der Korpuslinguistik

Metadaten sollen Sprachressourcen idealerweise derart umfassend und standardisiert beschreiben, dass sie zur Entscheidung darüber eingesetzt werden können, ob deren Inhalte für das jeweilige Untersuchungsspektrum wesentlich und ausreichend erschlossen sind. Insbesondere helfen sie bei der Komposition ausgewogener Teilkorpora und befördern dadurch eine systematische empirische Forschung.[57] Darüber hinaus beinhalten sie optional informatisch relevante Details hinsichtlich der im Zuge des Korpusaufbaus erfolgten Bearbeitungs- bzw. Annotationsschritte.

Die durch Metadaten ausgedrückten kontextuellen Informationen lassen sich unter dem Aspekt ihres Bezugsrahmens sowie ihrer hauptsächlichen Verwendung im Untersuchungskontext in folgende Kategorien einteilen:

– Angaben zum Entstehungskontext: Hierzu zählen Sprache, Medium, Textsorte/Gattung/Genre, Autor/Sprecher, Publikations-/Aufnahmeort, Entstehungszeit, Adressaten usw. Für Spezialuntersuchungen können auch soziologische Daten der Sprachproduzenten erkenntnisfördernd sein, also z.B. Geburts- bzw. Heimatort/-region, Geschlecht, Alter, soziale Herkunft als demografische Diversifikationskriterien. All diese wort- und satzübergreifenden Metadaten helfen bei der Einteilung und Gewichtung von Subkorpora. Außerdem stellen sie als Varianzfaktoren ergänzend zum Primärtext eine eigenständige Quelle für statistische Untersuchungen – etwa multivariate Analyseverfahren – von Sprachwandelphänomenen, Standardnähe u.Ä. dar.
– Inhaltliche Charakterisierungen: Zu diesem Zweck bietet es sich aus Gründen der effizienten maschinellen Weiterverarbeitung an, keine Freitexte (*abstracts*), sondern kurzgefasste, prägnante thematische Klassifizierungen von Topic bzw. Domäne einzusetzen. Diese können in Form einer Schlagwortliste (*keywords*) vorliegen oder Teil einer (hierarchischen) Taxonomie sein und unterstützen bei Bedarf ebenfalls das Design maßgeschneiderter Untersuchungskorpora.
– Technisch-administrative Angaben: In diese Kategorie fallen Lokalisierung, Format und Zeichensatz der Primärdaten, explizite Versionsangabe der verwendeten Tagger und Parser, Elemente und Attribute der Tagsets, Annotationsrichtlinien (z.B. welche Phänomene wurden mit welcher Granularität

häufig verwendeten Bezeichnungen erscheint der inhaltliche Aspekt sinnvoll: Annotationen kodieren i.d.R. linguistische Merkmale unterhalb der Textebene, Metadaten umfassen in erster Linie außersprachliche Spezifika auf und oberhalb der Textebene.

[57] Vgl. Burnard (2005): „Without metadata, corpus linguistics would be virtually impossible […]; without it, we have no way of distinguishing or grouping the component texts which make up a large heterogeneous corpus, nor even of talking about the properties of a homogeneous one."

annotiert?), Angaben zu menschlichen Revisoren bzw. Annotatoren, Brauchbarkeit der Auszeichnungen (z.B. via Inter-Annotator Agreement relativ zu einer Referenzressource), Auflistungen durchgeführter oder ausstehender Korrekturen, Bearbeitungs- und Zugriffsstatus usw. Solche Metadaten bilden die Basis für eine Beurteilung, ob Korpusinhalte allgemein verwendbar oder nur eingeschränkt nutzbar sind, sowie für die Auswahl der benötigten Werkzeuge für Analyse und Weiterverarbeitung.

Qualität und Nachhaltigkeit korpuslinguistischer Forschung hängen nicht zuletzt von der dauerhaften Gewährleistung einer computergestützten Vergleichbarkeit heterogener Sprachressourcen ab. Hierzu erscheint eine normierte und persistente Vorgehensweise bei der Anreicherung von Primärinhalten um Metadaten als wünschenswert.[58] Neben den bereits erwähnten Standards XCES und TEI, die entsprechende Kodierungsrichtlinien für die Kopfbereiche (*header*) von Korpustexten spezifizieren, existieren zu diesem Zweck verschiedene nationale und internationale Ansätze, Frameworks und Gremien.[59] Hervorzuheben ist die Component Metadata Initiative (CMDI), die seit 2008 im Umfeld des CLARIN-Projekts[60] unter Mitwirkung maßgeblicher Institutionen vorangetrieben wird. CMDI bietet ein modular aufgebautes und dadurch beliebig anpassbares Inventar an Metadatentypen (*categories*) speziell zur Abdeckung der Anforderungen unterschiedlicher linguistischer Teildisziplinen. Sofern für einen konkreten Anwendungsfall noch keine eingeführte Kategorie existiert, lässt sich eine solche individuell definieren und öffentlich bereitstellen. Auf diese Weise ist die gezielte Adaption an spezifische Datenerfordernisse korpusgestützter Projekte möglich. Insbesondere wird sichergestellt, dass starr vorgegebene Kategorien nicht sachfremd genutzt werden müssen und sich in der Folge nicht mehr eindeutig auswerten lassen.

CMDI-Metadatenkategorien sind zu Komponenten (*components*) kombinierbar und über die CLARIN Component Registry[61] editier- und abfragbar. Die

[58] Vgl. NISO (2004, S. 2): „Metadata is key to ensuring that resources will survive and continue to be accessible into the future." Zur Nachhaltigkeit zählt übrigens auch der Aspekt des Auffindens: Eine praktische Anwendung von Spezialkorpora mit Metadaten für das semantische Retrieval wird z.B. in Schneider (2018) vorgestellt.

[59] Ein aktueller Überblick über linguistische Annotations- und Metadatenstandards findet sich z.B. in CLARIN-D AP5 (2012) sowie unter www.computerlinguistik.org/portal/portal.html?s=Standardisierung.

[60] Vgl. Trippel et al. (2012) sowie www.clarin.eu/content/component-metadata.

[61] Vgl. http://catalog.clarin.eu/ds/ComponentRegistry; für die unmittelbare Bearbeitung der Metadatendateien lassen sich neben konventionellen XML-Werkzeugen spezielle Online-/Offline-Browser und -Editoren nutzen.

Anbindung an Primärdaten erfolgt unter Nutzung sogenannter Digital Object Identifiers (DOI, vgl. hierzu Kahn/Wilensky 2006), sofern die Originaltexte entsprechend eindeutig registriert und lokalisierbar sind. Das nachfolgende Beispiel illustriert in XML-Syntax den flexiblen CMDI-Ansatz unter Einbeziehung bereits existierender Metadatenressourcen. Es ist der CMDI-Spezifikation der Leipzig Corpora Collection (LCC) entnommen und definiert Syntax und Inventar für die Einbettung von Genre-Informationen in den Metadatenblock:

```
<CMD_Component CardinalityMax="1" CardinalityMin="0" name="Genres">
  <CMD_Element Multilingual="false" DisplayPriority="1" Cardinality
   Max="unbounded" CardinalityMin="1" ValueScheme="string"
   ConceptLink="http://www.isocat.org/datcat/ DC-2470"
   name="Genre"/>
</CMD_Component>
```

Die CMDI-Komponente „Genres“ verweist dabei auf die Kategorie „Genre“ der ISO-Data Category Registry (ISOcat). In diesem am Max-Planck-Institut für Psycholinguistik (MPI) in Nijmegen angesiedelten Online-Katalog sind sämtliche zugelassenen Inhaltsausprägungen (z.B. „discourse“, „drama“, „newspaper article“, „poetry, singing“) erfasst und dokumentiert. Über unser rudimentäres Beispiel hinaus sind auch komplexere Konstruktionen möglich, etwa die passgenaue Spezifikation von korpusrelevanten Personenangaben mit normierten Subelementen für Namen, Alter, Geschlecht, Herkunftsland usw. In ISOcat finden sich für solche Fälle weiterverwendbare Kategorien aus bekannten Metadateninitiativen, etwa der ISLE Meta Data Initiative (IMDI) oder der Dublin Core Metadata Initiative (DCMI).[62] Letztere wurde Mitte der Neunzigerjahre ursprünglich für den Bibliotheksbereich gegründet und über die Jahre auch für linguistische bzw. interdisziplinäre Anwendungszwecke eingesetzt. Allerdings hat sich gezeigt, dass die in Dublin Core fest vordefinierten 15 Metadatentypen für die vielfältigen von der Korpusdokumentation abzudeckenden Anforderungen nicht ausreichen. So bietet DCMI für die Kodierung zeitlicher Datumsstempel lediglich das recht unspezifische Element „date“. Eine verbindliche Festlegung darüber, was genau damit ausgedrückt wird (Zeitpunkt der Erstellung, der Aufnahme in das Korpus, der letzten Änderung, der Annotation etc.) sowie eine Option zur eindeutigen parallelen Kodierung mehrerer charakteristischer Zeitpunkte sind nicht vorgesehen. Hier bietet CMDI einen deutlich mächtigeren Ansatz; existierende Dublin Core-Beschreibungen lassen sich bei Kenntnis ihrer Erstellungsrichtlinien nach CMDI konvertieren.

[62] Vgl. https://tla.mpi.nl/imdi-metadata/ bzw. http://dublincore.org.

Generell werden korpusbezogene Metadatenelemente zwecks algorithmisierter Weiterverarbeitung vorzugsweise kurz und prägnant – also in den allermeisten Fällen numerisch oder schlagwortbasiert – sowie unter Nutzung kontrollierter Inventare inhaltlich belegt. In unterschiedlichem Maße lassen sie sich unter Heranziehung international anerkannter Normen formatieren, z.B.:

- Sprachangaben gemäß ISO-Standard 639 für Sprachkürzel (*language codes*) wie „de" für „deutsch" oder „en" für „englisch"; der in der Entwicklung befindliche vierstellige Substandard 639-6 bietet potenziell auch ausreichend Raum für dialektale Varianten.
- Geografische Angaben gemäß ISO-Standard 3166 für Länderkürzel (*country codes*) sowie feinkörnigere Regionsangaben z.B. gemäß NUTS (*Nomenclature des unités territoriales statistiques*), einer Systematik des Europäischen Amts für Statistik (EUROSTAT).
- Zeitliche Angaben im numerischen internationalen Datumsformat gemäß ISO-Standard 8601 in der Notation JJJJ-MM-TT hh:mm:ss, also z.B. „2016-04-27 20:30:00" für den 27. April 2016 AD halb neun Uhr abends.
- Inhaltliche Klassifizierungen unter Nutzung des Inventars von Wortnetzen oder Thesauri, also beispielsweise von GermaNet oder OpenThesaurus für das Deutsche.[63] Strukturell normiert ISO-Standard 2788 Äquivalenz-, Assoziations- und hierarchische Relationstypen, z.B. SYN (*synonym*) für Synonymie sowie BT (*broader term*) und NT (*narrower term*) für Hyperonymie bzw. Hyponymie, und kann dadurch für die Einordnung einzelner Inhaltsbeschreibungen in größere Kontexte eingesetzt werden.

Über die genannten Beispiele hinaus existieren weitere, vornehmlich technische Standards für die inhaltliche Ausgestaltung einzelner Metadatenelemente, etwa für die Kodierung von Textformat, Zeichensatz usw. Bei anderen Elementtypen gestaltet sich eine Normierung des Inventars schwieriger, da im praktischen Einsatz vielschichtige, vom Projektkontext abhängige Erfordernisse in den Fokus rücken. Hierzu zählen etwa Angaben zu Textsorte, sozialer Herkunft oder Adressatenkreis. Je nach Untersuchungsinteresse und -methode haben diese Metadatentypen ganz unterschiedliche Phänomenaspekte und Ausprägungen in variabler Granularität abzudecken, so dass universal verbindliche Vorgaben rasch an ihre Grenzen stoßen. Dessen ungeachtet existieren auch hier vereinzelte regelhafte Klassifikationsansätze, beispielsweise durch die Einbeziehung journalistischer Darstellungsformen („Nachricht", „Porträt", „Essay", „Kommentar", „Interview") für die Textsortenklassifikation in zeitungsbasierten Korpora.

[63] Vgl. www.sfs.uni-tuebingen.de/GermaNet/ bzw. www.openthesaurus.de.

2.2 Deutschsprachige Korpora im internationalen Kontext

Hinsichtlich der Erforschung der deutschen Sprache auf Basis digitaler Korpora kann konstatiert werden, dass diese speziell seit den 1990er Jahren einen bemerkenswerten Aufschwung genommen hat. Während bis dahin häufig technische, institutionelle oder rechtliche Problematiken die substantielle Nutzung linguistisch aufbereiteter Korpussammlungen erschwerten, lässt sich in der Folgezeit eine zunehmende Verfügbarkeit korpuslinguistischer Werkzeuge und Sprachressourcen beobachten. Und während frühe Korpussammlungen vorrangig das Englische fokussierten, existieren mittlerweile umfangreiche Datensets für die Dokumentation weiterer Nationalsprachen und Varietäten; allein die Leipzig Corpora Collection (LCC) als Basis des Wortschatz-Portals an der Universität Leipzig deckt rund 200 Sprachen ab.

Für die korpusgestützte Erforschung der deutschen Sprache ist es darüber hinaus essentiell, dass neben den Primärdaten auch adaptierte computerlinguistische Werkzeuge zur Verfügung stehen. Hierzu gehören insbesondere Tagger und Parser für die Analyse geschriebener oder gesprochener Rohdaten unter Berücksichtigung der einzelsprachlichen Besonderheiten. Flankiert wird dieser Ausbau von Infrastrukturen für die systematische empirische Auswertung umfangreicher Sprachkorpora durch eine interdisziplinäre Kooperation geistes- und kulturwissenschaftlicher Forschungsinstitutionen unter dem Dach der *digital humanities* bzw. *eHumanities* (Schreibman/Siemens/Unsworth (Hg.) 2016) sowie durch die Bündelung existierender Ressourcen. Ein positives Signal haben in diesem Zusammenhang die nationalen und internationalen eHumanties-Initiativen und -Forschungsverbünde gesetzt, beispielsweise TextGrid, CLARIN (Common Language Resources and Technology Infrastructure) oder DARIAH (Digital Research Infrastucture for the Arts and Humanities).[64]

Die nachfolgende Auflistung verortet den Stand des Ausbaus deutschsprachiger wissenschaftlicher Korpusprojekte im internationalen Kontext. Dabei kann und soll es keinesfalls um eine exhaustive Dokumentation sämtlicher welt-

[64] TextGrid (www.textgrid.de) ist ein seit 2006 vom Bundesministerium für Bildung und Forschung (BMBF) gefördertes Verbundprojekt zum Aufbau einer digitalen Forschungsinfrastruktur für die Humanwissenschaften; seit 2012 bildet ein eingetragener Verein (e.V.) die Basis für die nachhaltige Pflege des Angebots. CLARIN (www.clarin.eu) wird seit 2008 unter initialer Förderung durch die Europäische Kommission vorangetrieben und ermöglicht Forschern aus den „Digital Humanitites" einen vereinfachten Zugriff auf europaweit verteilte sprachwissenschaftliche Datenquellen und Service-Zentren (Ketzan/Schuster 2012; Heyer et al. 2015). Auch DARIAH (www.dariah.eu, Henrich/Gradl 2013) ist ein europäisch ausgelegtes Infrastrukturprojekt für die Geisteswissenschaften, dessen Vorbereitungsphase 2011 erfolgreich abgeschlossen wurde und das u.a. Ergebnisse des TextGrid-Projekts integriert.

weiter Korpusprojekte gehen, hierfür bieten Online-Verzeichnisse[65] ohnehin den aktuelleren Rahmen. Vielmehr lassen sich auf diese Weise – fünf Jahrzehnte nachdem in Gestalt des Brown-Korpus erstmals elektronische Textsammlungen für linguistische Untersuchungen kompiliert wurden – stichhaltige Vorstellungen über Umfang und Komplexität verfügbarer Korpora vermitteln sowie Anforderungen an deren Verwaltungssysteme realistisch einschätzen. Wir beschränken uns auf die nach unseren Recherchen umfangreichsten, synchron ausgerichteten Korpusinitiativen für europäische Sprachen; oft sind das sogenannte Nationalkorpora, die als Grundlage referenzieller sprachbeschreibender Untersuchungen dienen. Neben den Primärdaten bieten die aufgelisteten Korpora – die zumeist über unterschiedlich mächtige Online-Schnittstellen recherchierbar sind – stets ein Basisinventar textspezifischer Metadaten und mindestens eine linguistisch motivierte Annotationsebene[66] an:

– Bulgarisch: Das seit 2009 kompilierte „Bulgarian National Corpus (BulNC)" an der „Bulgarian Academy of Science" (http://search.dcl.bas.bg) umfasst ca. 1,2 Milliarden laufende Wortformen in ca. 240.000 Texten. Der Schwerpunkt liegt auf geschriebener Sprache und Internetdokumenten, außerdem enthält die Sammlung einen großen Anteil an übersetzten Texten. Für die Dokumentklassifizierung kommen 27 Metadatenkategorien wie Autor, Jahr, Genre, Domäne etc. zum Einsatz. Die Mehrebenen-Annotation erfolgte unter Verwendung sprachspezifischer Werkzeuge (Satzgrenzenerkenner und Tokenisierer, lexikonbasierter Lemmatisierer, POS-Tagger auf Basis von Support Vector Machines, Finite State Chunker, Wordnet-Bedeutungsannotation).
– Dänisch: Das Kopenhagener „KorpusDK" (https://ordnet.dk/korpusdk/) beinhaltet momentan knapp 60 Millionen laufende Wortformen aus Zeitungen, Periodika, Büchern und sonstigen Quellen ab 1983. Die Texte enthalten neben den Standardinformationen (z.B. Publikationsjahr, Titel) – soweit ermittelbar – autorenspezifische Metadaten (Alter, Geschlecht) sowie Angaben zum Texttyp. Die Constraint Grammar-basierte morphosyntaktische Auszeichnung erfolgte unter Einsatz des *DanPars*-Taggers.

---

[65] Anlaufstellen im Internet sind beispielsweise die Kataloge der European Language Resources Association ELRA (http://catalog.elra.info) oder des Linguistic Data Consortiums LDC (www.ldc.upenn.edu/Catalog/), die Korpusliste der CLARIN-D-Zentren (www.clarin-d.net/de/auffinden), das Ressourcenverzeichnis des Computerlinguistik-Portals (www.computerlinguistik.org/portal/portal.html?s=Ressourcen), das Semtracks Corpora Directory (www.semtracks.org/web/index.php?id=Corpora%20Directory) oder die Liste der mit Sketch Engine recherchierbaren Korpora (www.sketchengine.eu/user-guide/user-manual/corpora/by-language/).

[66] Weiterführende Informationen zu den hierfür verwendeten Werkzeugen finden sich auf den jeweils angegebenen Projektseiten.

- Englisch: Die Inhalte der prominenten BNC-, Brown-, Cobuild-, und Gigaword-Korpora wurden bereits im einleitenden Kapitel 1 vorgestellt. Daneben bietet seit 2013 das länderübergreifende „Corpus of Global Web-based English (GloWbE)“ ca. 1,9 Milliarden laufende Wortformen primär aus Internetquellen. Neben textspezifischen Eigenschaften wie Ursprungsland, Publikationsjahr oder Genre wurden hier mit dem *CLAWS*-Tagger morphosyntaktische Sekundärdaten sowie Synonymrelationen ausgezeichnet. Ebenfalls an der Brigham Young University (https://corpus.byu.edu) finden sich das mehrsprachige Korpus „News on the Web (NOW)“ mit 6 Milliarden Wortformen sowie seit 2017 das „iWeb: The Intelligent Web-based Corpus“ mit 14 Milliarden Wortformen und explizit markierten dialektalen Zuordnungen (US/CA/UK/IE/AU/NZ).
- Französisch: Das 2006 kompilierte und am „Centre for Translation Studies“ der Universität Leeds abfragbare Korpus „I-FR“ (http://corpus.leeds.ac.uk/internet.html) umfasst ca. 200 Millionen laufende Wortformen aus Internetquellen. Tokenisierung, Lemmatisierung und POS-Tagging wurden mit *TreeTagger* durchgeführt; daneben enthält das Korpus maschinell generierte thematische Klassifikationen und geografische und Quellenangaben. Mit knapp 10 Milliarden Wortformen deutlich umfangreicher und ebenfalls mit *TreeTagger* analysiert ist das Internetkorpus „frTenTen: Corpus of the French Web 2012“ (www.sketchengine.eu/frtenten-french-corpus/).[67]
- Griechisch: Seit 2000 ist das „Hellenic National Corpus (HNC)/ILSP Corpus“ am Athener „Institute for Language and Speech Processing (ILSP)“ (www.ilsp.gr/en/) mit knapp 50 Millionen laufenden Wortformen in einem relationalen Datenbanksystem online. Die ausschließlich schriftsprachlichen Inhalte (Buchpublikationen, Zeitungen/Zeitschriften, Broschüren, Internetquellen) wurden mit *HNCedit* morphosyntaktisch annotiert und enthalten auf Dokumentebene normierte Metadaten wie Titel, Autor, Medium oder Genre sowie eine inhaltliche Klassifizierung.
- Italienisch: Das 2012 an der Universität von Pisa aufgebaute „PAISÀ“-Korpus (www.corpusitaliano.it) enthält 250 Millionen laufende Wortformen. Basierend auf hochfrequenten Wortlisten wurden unter Einsatz von Bootstrap- und Retrieval-Tools ca. 380.000 frei verfügbare Internetquellen erfasst und im CoNLL-Format mit Metadaten etikettiert. Die morphosyntaktische Analyse übernahm der *ILC-POS-Tagger*; hierarchische Dependenzrelationen ermittelte der *DeSR Dependency Parser* entsprechend des *ISST-TANL Dependency Tagsets*.

[67] Das French Web Corpus sei an dieser Stelle exemplarisch genannt; inzwischen existieren für sämtliche in unserer Liste behandelten Sprachen ähnliche via Sketch Engine abfragbare Internet-Sammlungen.

- Litauisch: Im „Corpus of the Contemporary Lithuanian Language (CCLL)“ am „Centre of Computational Linguistics (CCL)“ der Universität von Kaunas (http://tekstynas.vdu.lt/tekstynas/) finden sich ca. 160 Millionen laufende Wortformen aus medial klassifizierten Printdokumenten ab 1990. Unter Anwendung statistischer Hidden Markov Modelle (HMM) wurde das Korpus morphosyntaktisch annotiert.
- Kroatisch: Das „Kroatische Nationalkorpus (Hrvatski Nacionalni Korpus, HNK)“ an der Universität von Zagreb (www.hnk.ffzg.hr) exisitiert seit 1998 und speichert über 200 Millionen laufende Wortformen in einer relationalen Datenbank, stratifiziert nach Medium (Zeitung, Zeitschrift, Populärliteratur, Korrespondenz usw.) und Genre (Politik, Wirtschaft, Sport usw.). Die morphosyntaktischen Annotationen verwenden ein kroatisches *MSD Tagset*.
- Niederländisch: Zwischen 2008 und 2011 wurde an Hochschulen in Nijmegen, Gent, Leuven, Tilburg, Twente und Utrecht das „SoNaR“-Korpus (STEVIN Nederlandstalig Referentiecorpus, http://lands.let.ru.nl/projects/SoNaR/) in der Nachfolge des ebenfalls niederländisch-flämischen Pilotprojekts „Dutch Corpus Initiative (D-Coi)“ zusammengestellt. Es umfasst mit Metadaten angereicherte Volltexte (entstanden ab Mitte der 1950er Jahre) unterschiedlicher Art und Herkunft (Literatur, Journalistik, Instruktionstexte, administrative Texte etc.) mit insgesamt ca. 500 Millionen Textwörtern. Neben einer syntaktischen Annotation wurden vier semantische Annotationsebenen (Named-Entities, Koreferenzbeziehungen, semantische Rollen und räumlich-zeitliche Beziehungen) erstellt.
- Norwegisch: Das 1999 entstandene „Oslo Corpus of Tagged Norwegian Texts“ an der Universität von Oslo (www.tekstlab.uio.no/norsk/bokmaal/english.html) enthält ca. 20 Millionen laufende Wortformen. Das Textspektrum reicht von Zeitungen/Zeitschriften über Unterhaltungsliteratur bis hin zu Fachtexten; das Metadatenformat entspricht dem Standard der *IMS Corpus Workbench*. Morphologische und syntaktische Annotationen wurden mit Hilfe zweier speziell entwickelter Tagger (Multitagger und dependenzgrammatischer disambiguierender Tagger) hinzugefügt.
- Polnisch: Das von der „Polish Academy of Sciences“ seit 2007 koordinierte „National Corpus of Polish (NKJP)“ (http://nkjp.pl) umfasst 1,8 Milliarden laufende Wortformen, von denen 300 Millionen in ein ausgewogenes und 1,2 Millionen in ein manuell annotiertes Subkorpus einfließen. Automatisch wurden drei Annotationslevel (morphologisch, syntaktisch, Named-Entities) mit den regelbasierten Tools *Spejd* und *Sprout* generiert. Sämtliche

Buch- und Zeitungstexte wurden mit Metadaten (Autor, Titel, Erscheinungsjahr usw.) versehen.

– Portugiesisch: Das „Centro de Linguística" der Universität Lissabon stellt das „Reference Corpus of Contemporary Portuguese (CRPC)" (http://alfclul.clul.ul.pt/CQPweb/) mit ca. 310 Millionen laufenden Wortformen bereit. Der Schwerpunkt liegt auf geschriebenen Texten ab 1970; ein kleines Subkorpus gesprochener Sprache wurde mit *EXMARaLDA* aligniert. Die Inhalte stammen aus portugiesischen und außereuropäischen Quellen und wurden um geografische, thematische sowie autorenspezifische Metadaten ergänzt. Für die Tokenisierung kam der *LX tokenizer*, für die Lemmatisierung eine portugiesische Version von *MBLEM* und für die Wortarterkennung der *MBT*-Tagger zum Einsatz.
– Rumänisch: Das 2012 kompilierte „Romanian Balanced Annotated Corpus (ROMBAC)" an der Rumänischen Akademie in Bukarest (Download via http://metashare.elda.org) umfasst über 40 Millionen laufende Wortformen. Es ist hinsichtlich fünf möglicher Ausprägungen des Parameters „Genre" (journalistisch, medizinisch, juristisch, biografisch, fiktional) ausgewogen. Linguistische Annotationen (Token, Lemma, POS, Satzkonstituenten) wurden unter Einsatz der *TTL*-Plattform erstellt, wobei die morphosyntaktische Annotation auf Hidden Markov Modellen (HMM) und die syntaktische Annotation auf Shallow Parsing beruht. ROMBAC bildet die Grundlage eines zukünftigen Referenzkorpus namens CoRoLa, das ca. 500 Millionen Wortformen aufnehmen soll. Geplant ist eine Anreicherung um weitere Annotations- und Metadatentypen; ein Teil des Korpus soll als syntaktisch annotierte Baumbank vorgehalten werden.
– Russisch: Das an der Moskauer „Russian Academy of Sciences" koordinierte „Russian National Corpus (RNC)" (http://ruscorpora.ru) ist seit 2003 im Aufbau und umfasst momentan über 300 Millionen laufende Wortformen. Der Schwerpunkt liegt auf geschriebener Sprache (Zeitungen/Zeitschriften, Fachtexte, Belletristik, Poesie, Korrespondenz), des Weiteren wird ein multimediales Subkorpus für Spontansprache aufgebaut. Neben einer genrebasierten Textklassifikation enthalten die Metadaten Publikationsjahr, regionale Zuordnung und Autorenspezifika. Maschinell generiert wurden morphosyntaktische Annotationen sowie eine lexikonbasierte semantische Annotation unter Verwendung einer Taxonomie; für Teilkorpora sind Akzent, rhythmische Gliederung u.a. kodiert.
– Schwedisch: Die seit 1975 an der Universität von Göteborg angesiedelte „Språkbanken" (https://spraakbanken.gu.se) enthält in einer mit Hilfe des KORP-Frontends recherchierbaren Sammlung ca. 13 Milliarden lau-

fende Wortformen. Modernes Schwedisch ist mit ca. 2 Milliarden Wortformen vertreten, vornehmlich aus literarischen und journalistischen Quellen, aber auch Weblogs oder das schwedische Europarl-Korpus; textspezifische Metadaten sind im CMDI/ISOCat-Format kodiert. Mit Hilfe von Eigenentwicklungen und freien Tools wurden morphosyntaktische Kategorien maschinell annotiert, für die Auszeichnung von Dependenzrelationen kam der *MaltParser* zum Einsatz. Als lexikalische Ressource für semantische Annotationen dient der *SALDO*-Thesaurus.

– Slowenisch: Im vom nationalen Wissenschaftsministerium verantworteten Korpus „GigaFIDA" (www.gigafida.net) kann nach ca. 1,2 Milliarden laufenden Wortformen recherchiert werden. Es enthält seit 1990 publizierte Texte aus Sachbüchern, Unterhaltungsliteratur, Zeitungen/Zeitschriften, Online-Portalen, parlamentarischen Reden usw. Soweit verfügbar, wurden zu den üblichen textspezifischen Metadaten soziologische Angaben über den Autor hinzugefügt. Die maschinelle morphosyntaktische Annotation wurde vermittels des statistischen Taggers *Obeliks* durchgeführt.

– Spanisch: Das im Rahmen des „Spanish FrameNet (SFN)" zusammengestellte „Corpus del Español Actual (CEA)" (http://spanishfn.org/tools/cea/english) umfasst 540 Millionen laufende Wortformen. Es besteht im Wesentlichen aus den spanischen Beiträgen des Europarl-Korpus (1996-2010), des Wikicorpus (2006) und des MultiUN-Korpus der Vereinten Nationen (2000-2009) mit ihren entsprechenden Metadaten. Geparst und morphosyntaktisch annotiert wurde es an der Universität von Barcelona unter Einbeziehung lexikalischer Ressourcen und Finite State-Werkzeugen zur Disambiguierung.

– Tschechisch: Das Tschechische Nationalkorpus (Cesky Národní Korpus, CNK) an der Universität Prag (https://korpus.cz) versammelt in seinen synchronischen Textkorpora über vier Milliarden laufende Wortformen (Release 6 vom Dezember 2017). Der überwiegende Anteil stammt aus Zeitungen und Zeitschriften ab 1990, kleinere Subkorpora enthalten Belletristik, Korrespondenz u.a. Textspezifische Metadaten wie Titel, Autor oder Publikationsjahr wurden um eine auf Register/Domäne basierende Klassifizierung ergänzt, die die Grundlage für die quantitative Korpusgewichtung bildet. Morphosyntaktische Kategorien annotierte ein Head-Driven-Chartparser, für die Disambiguierung wurden probabilistische Methoden implementiert.

– Türkisch: Das an der Universität von Mersin entwickelte Türkische Nationalkorpus „TNC" (www.tnc.org.tr) enthält 50 Millionen laufende Wortfor-

– men aus vornehmlich geschriebenen Quellen (2% gesprochene Sprache) ab 1990. Das Korpus ist auf der Basis von Publikationsjahr, Domäne und Medium ausgewogen, jeder Text ist darüber hinaus mit autorenspezifischen Metadaten versehen. Für das POS-Tagging wurde ein regelbasierter Ansatz mit einer probabilistischen Analyse kombiniert.
– Ungarisch: Seit 1998 ist das ausgewogene „Hungarian National Corpus (HNC)" (http://corpus.nytud.hu/mnsz/) an der Ungarischen Akademie der Wissenschaften (HAS) in Budapest beheimatet. Es enthält knapp 190 Millionen laufende Wortformen, unterteilt in je fünf Genre- und Regionspezifische Subkorpora. Für morphologische Analysen (Lemma, POS, Flexionsinformationen) wurde der speziell entwickelte *Humor*-Parser eingesetzt, die Disambiguierung übernahm der statistische *TnT* (*Trigrams'n'Tags*)-Tagger. Als aktuelles Nachfolgeprojekt weist das „Hungarian Gigaword Corpus (HGC)" eine ähnlich konzeptionierte Stratifikation auf; für die nun ca. 1,5 Milliarden, im WaCky-Format annotierten Wortformen dienen *Manatee* und *Bonito* als Korpusverwaltungs- bzw. Abfragesysteme.

Ergänzend zu diesen ausgewählten Projekten existieren eine Vielzahl weiterer einzelsprachlicher und sprachübergreifender Korpussammlungen sowie Initiativen mit spezifischen Fokussierungen, beispielsweise das bereits erwähnte parallele Europarl-Korpus (Koehn 2005) mit frei verwertbaren Parlamentsdokumenten in 21 europäischen Sprachen, morphosyntaktisch annotiert mit unterschiedlichen Tagsets. Andere umfangreiche und mehrsprachige Ressourcen sind das auf Webseiten-Crawling basierende COW-Projekt („Corpora from the Web", Schäfer/Bildhauer 2012; Schäfer 2016) oder das WaCky-Projekt („Web as Corpus kool ynitiative", Baroni et al. 2009). Die COW-Datenbasis unterteilt sich in deutsche, englische, französische, niederländische, schwedische und spanische Subkorpora; DECOW16 enthält ca. 20 Milliarden Wortformen, annotiert u.a. mit SMOR, Marmot, IMS Mate und dem Stanford Named Entity Recognizer. WaCky berücksichtigt deutsche, englische, französische und italienische Inhalte; es enthält für jede Einzelsprache jeweils zwischen einer und zwei Milliarden laufende Wortformen, die – überwiegend unter Nutzung von *TreeTagger* – segmentiert, lemmatisiert und morphosyntaktisch annotiert vorliegen.

Korpuslinguistische Untersuchungen zum Deutschen können auf eine Reihe qualitativ und quantitativ hochwertiger Ressourcen zurückgreifen. Neben Baumbanken wie dem Saarbrücker „Negra"-Korpus, „TIGER" am Stuttgarter „Institut für Maschinelle Sprachverarbeitung (IMS)" oder den Baumbanken am Tübinger Arbeitsbereich „Allgemeine Sprachwissenschaft und Computer-

linguistik" mit vergleichsweise geringem Primärdatenvolumen[68] – dafür allerdings mit manuell bzw. semi-automatisch annotierten syntaktischen Strukturen – und diversen Spezialkorpora lassen sich für empirische Studien insbesondere die nachfolgenden großen Korpussammlungen nutzen:

1) Deutsches Referenzkorpus (DeReKo): Das am Institut für Deutsche Sprache (IDS) in Mannheim beheimatete DeReKo (www.ids-mannheim.de/kl/projekte/korpora/) ist die weltweit größte wissenschaftlich motivierte Korpussammlung für deutsche Schriftsprache mit 2018 mehr als 40 Milliarden laufenden Wortformen. Der Korpusname basiert auf einem 2002 abgeschlossenen Kooperationsprojekt zum Aufbau einer repräsentativen Datenbasis des Deutschen und ist heutzutage insofern missverständlich, als DeReKo seither als bewusst unausgewogene „Urstichprobe" für die individuelle Zusammenstellung virtueller Untersuchungskorpora fortgeführt wird. Journalistische Texte (u.a. Monitorkorpora diverser Zeitungen und Zeitschriften aus deutschsprachigen Ländern) nehmen einen breiten Raum ein. Daneben enthält die Sammlung eine breite Auswahl belletristischer, fachsprachlicher und populärwissenschaftlicher Inhalte, die unter Zuhilfenahme einer Themen-Taxonomie inhaltlich klassifiziert wurden. Für linguistische Analysen von besonderem Interesse ist die parallele Annotation der Primärinhalte mit unterschiedlichen morphosyntaktischen Taggern.
2) Leipzig Corpora Collection (LCC): In der Abteilung „Automatische Sprachverarbeitung (ASV)" am Institut für Informatik der Universität Leipzig wird seit Mitte der Neunzigerjahre unter der Projektbezeichnung „Deutscher Wortschatz" (http://wortschatz.uni-leipzig.de) eine Textdatenbank aufgebaut, die mittlerweile als LCC außer dem Deutschen (mit derzeit ca. 5 Milliarden laufenden Wortformen in ca. 300 Millionen Sätzen) auch eine Vielzahl weiterer Sprachen abdeckt. Für jede Einzelsprache existieren auf Satzlevel gesampelte Subkorpora mit gestaffelten Größen. Inhaltlich liegt der Schwerpunkt auf Zeitungstexten, per Bootstrapping zufällig ausgewählten öffentlich zugänglichen Webseiten sowie Wikipedia-Material. Diese Quellen werden maschinell unter Einsatz eines statistischen, nicht-überwachten und nicht-lernenden Textklassifikators analysiert; da-

[68] Vgl. www.coli.uni-saarland.de/projects/sfb378/negra-corpus/ (Negra mit 355.000 Textwörtern), www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/tiger.html (TIGER mit 900.000 Textwörtern), www.sfs.uni-tuebingen.de/de/ascl/ressourcen/corpora.html (z.B. TüBa-D/S mit 360.000 Textwörtern, TüBa-D/Z in Release 10.0 mit 1,7 Millionen Textwörtern). Für die webbasierte Recherche und Visualisierung von Suchergebnissen auf solchen Baumbanken existiert mit TÜNDRA (Martens 2013) eine innerhalb der CLARIN-Infrastruktur frei verfügbare Applikation.

rüber hinaus existieren Werkzeuge für die morphologische Analyse und Grundformreduktion, für die Erkennung von Eigennamen oder von semantischen Wortähnlichkeiten.

3) DWDS-Korpora: Im Rahmen des Projekts „Digitales Wörterbuch der deutschen Sprache (DWDS)“ (www.dwds.de) wird an der Berlin-Brandenburgischen Akademie der Wissenschaften seit 2000 eine digitale Korpussammlung als empirische Grundlage des Wortinformationssystems aufgebaut. Ein Kernkorpus des 20. Jahrhunderts umfasst ca. 120 Millionen laufende Wortformen, für das 21. Jahrhundert entsteht eine vergleichbare Ressource. Beide Kernkorpora enthalten festgelegte Anteile verschiedener Textsorten (Belletristik, Gebrauchsliteratur, Wissenschaft, Journalistische Prosa), die gleichmäßig über den Erfassungszeitraum gestreut sind. Ergänzend werden historische und gegenwartssprachliche Spezial- und Zeitungskorpora mit einem Umfang von insgesamt ca. 13 Milliarden laufenden Wortformen (Stand 2018) vorgehalten. Für die morphologische Analyse sowie die Erkennung semantischer Lesarten kommt ein lexikon- und regelbasiertes System zum Einsatz, darüber hinaus wurden ein Finite-State-Eigennamenerkenner, ein statistischer Wortarten-Tagger sowie ein regelbasierter Bottom-Up-Dependenzparser implementiert.

Insgesamt dokumentiert, im Vergleich mit den ca. 1 Million Wortformen im Brown-Korpus der 1960er Jahre, bereits der rein zahlenmäßige Fortschritt den gewachsenen Stellenwert und Anspruch natürlichsprachlicher Korpora als Arbeitsgrundlage der empirisch arbeitenden Sprachwissenschaft. Bemerkenswert an obiger Übersicht ist weiterhin, dass insbesondere eine Reihe vermeintlich „kleinerer“ Sprachen wie Bulgarisch, Schwedisch, Slowenisch oder Tschechisch auf quantitativ herausragende, annotierte Korpussammlungen mit jeweils über einer Milliarde Textwörtern zurückgreifen können. Darüber hinaus wird die ausgesprochen vorteilhafte Situation der zum Deutschen arbeitenden Korpuslinguisten deutlich. Diesen steht eine respektable Auswahl aussagekräftiger Datenquellen mit Tokenzahlen im höheren Milliardenbereich und einem breiten Spektrum an Sekundärdaten zur Verfügung, darunter die mit Abstand größte wissenschaftlich motivierte Einzelressource DeReKo. Auch wenn aus rechtlichen Gründen nicht immer der Gesamtbestand aller Korpussammlungen ohne Einschränkung abfragbar ist, so bieten diese Dateninventare doch ideale Voraussetzungen für eine authentische Beschreibung selbst seltener bzw. komplexer Sprachphänomene.

Allerdings veranschaulicht unsere Übersicht auch die aus den Ressourcengrößen unmittelbar erwachsenden technologischen Herausforderungen an Korpusabfragesysteme. Diese steigen naheliegenderweise stetig mit der Aus-

weitung der Primärdaten, deren durchschnittlicher Token-Umfang sich innerhalb weniger Jahrzehnte um den Faktor 1.000 erhöht hat. DeReKo mit über 40 Milliarden laufenden Wortformen entspricht – bei einer durchschnittlichen Füllung von 400 Wörtern pro Seite – einem Printvolumen von mehr als 100 Millionen Buchseiten. Aneinandergereiht ließe sich mit einer aus einem solchen Bestand generierten horizontalen Zeichenkette der Äquatorkreis der Erde mehr als fünfzehnmal abdecken. Da Korpussysteme im Gegensatz zu traditionellen Volltextsuchwerkzeugen nicht nur Fundstellen für Wortsuchen auflisten, sondern darüber hinaus *KWIC* (*key word in context*)-Ansichten generieren, Konkordanzen/Kollokationen berechnen oder frequenz- bzw. verteilungsbasierte Statistiken bereitstellen sollen, begründen derartige Korpus-größen im Milliarden-Token-Bereich einen erheblichen hard- und softwaretechnischen Skalierungsbedarf.

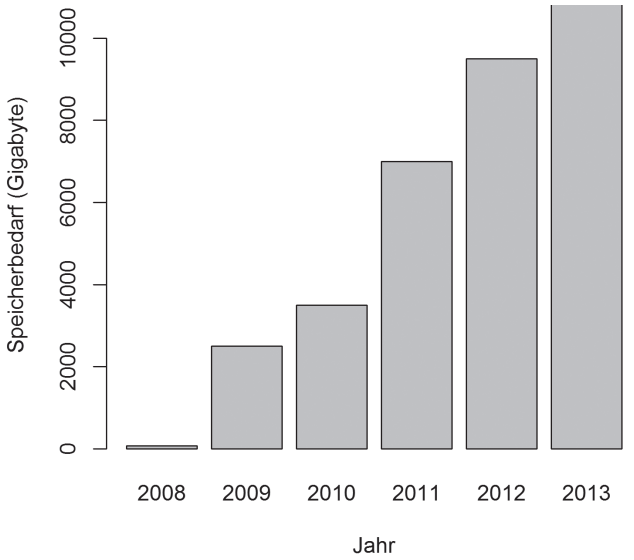


Abb. 3: Entwicklung des DeReKo-Speicherbedarfs zwischen 2008 und 2013

Die Abfrage- und Analyseanforderungen primärer Korpusinhalte wachsen noch weitestgehend proportional zur Erhöhung der Tokenzahl.[69] Einen weitaus gravierenderen Einfluss auf die Verarbeitungsperformanz nehmen hinge-

[69] Als Randnotiz sei erwähnt, dass einige Korpusretrievalsysteme Primärdaten filtern, indem sie „unpassende" (im Sinne von „ungrammatisch" bzw. „nicht sprachrelevant") Sätze aussortieren. Kriterien hierfür sind beispielsweise die Satzlänge, die Anzahl von Einzelbuchstaben innerhalb eines Satzes oder fehlende Satzzeichen. Aus streng wissenschaftlicher Perspektive sollten solche Verfahren vorzugsweise nicht vorab, sondern zur Laufzeit erfolgen, da sie dann parametrisierbar und ggf. abschaltbar sind.

gen Umfang und Komplexität der insgesamt relevanten Datenbasis, namentlich der Sekundärdaten. Hierzu tragen einerseits – zumeist außersprachliche – Metadaten auf Text- und Korpusebene bei, hauptsächlich jedoch die textinterne Auszeichnung diverser linguistischer Phänomene. Exemplarisch illustriert Abbildung 3 die Entwicklung des DeReKo-Speicherbedarfs zwischen 2008 und 2013: Innerhalb eines halben Jahrzehnts steigerte sich das Volumen sämtlicher Primär- und Sekundärdaten von 75 GB auf über 10 TB, d.h. etwa um den Faktor 150. Im selben Zeitraum stieg die Anzahl der gespeicherten laufenden Wortformen allerdings „lediglich" von knapp 3,5 Milliarden auf ca. 6 Milliarden – mithin in einer weitaus geringfügigeren Größenordnung. Die massive Zunahme des Speicherbedarfs erklärt sich in erster Linie durch die in diesem Zeitraum hinzugekommenen parallelen morphosyntaktischen Annotationen.

Im Vergleich zu anderen datenintensiven Szenarien, etwa in den Bereichen E-Commerce, Telekommunikation oder Soziale Netzwerke, bei denen Datenmengen leicht den Petabyte-Bereich erreichen, erscheinen die Anforderung annotierter Textkorpora auf den ersten Blick vergleichsweise moderat. Allerdings gilt es zu berücksichtigen, dass Textdaten bezüglich des reinen Speichervolumens zwar durchgängig anspruchsloser auftreten als etwa multimediale Ton- oder Videodaten, aber im Gegenzug hinsichtlich der Recherche äußerst komplexe interne Strukturen aufweisen. Die korpustechnologischen Herausforderungen liegen also weniger in der Bereitstellung des Speicherplatzes, sondern in der Handhabung der Beziehungen zwischen den Datensätzen. Eine Milliarde gespeicherte Wortformen etwa implizieren bei einer damit einhergehenden Annotation morphologischer, syntaktischer oder semantischer Merkmale nicht allein eine Zunahme der Speichermenge um jeweils eine weitere Milliarde Lemmaformen, Wortarten-Tags oder Bedeutungsangaben. Für die Rechercheorganisation bedeutsamer erscheint vielmehr die massive Ausweitung potenziell abfragerelevanter Muster und Relationen zwischen auf unterschiedlichen linguistischen Beschreibungsebenen angesiedelten Parametern.

Soll beispielsweise die Variation verschiedener Genitivendungen im deutschen Wortschatz empirisch analysiert werden, wäre die kombinierte Abfrage folgender sprachimmanenter, gebrauchsbasierter oder außersprachlicher Einzelparameter potenziell aufschlussreich: „Finde häufig verwendete (Parameter: Korpusfrequenz), maskuline (Parameter: Genus), fachsprachliche (Parameter: lexikalische Integration) Komposita mit Fugen-*s* (Parameter: morphologische Komplexität), die unmittelbar vor oder nach (Parameter: Position im Satz) einer Genitiv-Präposition (Parameter: Wortart) stehen und auf *-es* enden

(Parameter: Tokenendung).“ Sofern, wie etwa im Falle von DeReKo, die Primärdaten sogar parallel durch verschiedenartige Tagger aufbereitet wurden und dadurch für einzelne linguistische Beschreibungsebenen konkurrierende Sekundärdaten vorliegen, könnte darüber hinaus eine Einbeziehung der jeweiligen Annotationsspezifika wünschenswert erscheinen, also etwa die Fokussierung auf übereinstimmend bzw. nicht-übereinstimmend klassifizierte Phänomene.

Eine weitere Zuspitzung erfährt die musterbasierte Abfragekomplexität, wenn zusätzliche Segmentierungen unter- und oberhalb der Wortebene hinzukommen. Allein die Hinzunahme der morphologischen Ebene erhöht die Menge abfragbarer Muster um Kombinationen mit Präfixen, Fugenelementen, Suffixen etc.; daneben sind beispielsweise Segmentierungen von Lauten, Silben oder Teilsätzen/Phrasen möglich. Unser obiges Beispiel dürfte dann um Parameter wie „mit Anlautbbetonung“, „mit mehr als vier Silben“, „mit Fugen-*s*“ oder „innerhalb einer Präpositionalphrase“ ergänzt werden. Da alle diese in den Sekundärdaten kodierten Informationen grundsätzlich an beliebiger Position in eine musterbasierte Korpusabfrage einfließen können, führt das Hinzufügen zusätzlicher Ebenen zur Datenbasis zu einem nicht mehr linearen, sondern exponentiellen Wachstum potenziell abfragerelevanter Phänomenkombinationen. Korpora mit mehreren Milliarden laufenden Wortformen stoßen in diesen Fällen rasch in Komplexitätsdimensionen vor, die programmierte Werkzeuge vor immense Herausforderungen stellen.

2.3 Recherche in ausgewählten Korpussammlungen

Für die algorithmisierte Exploration natürlichsprachlicher Korpora lassen sich unterschiedlich mächtige, im konkreten Einsatz miteinander kombinierbare Strategien unterscheiden:

– Die Einzelwortsuche fahndet nach Belegen für zusammenhängende Zeichenketten. Zumeist versteht man darunter das Auffinden einzelner Token (*läuft*) oder Lemmata (*laufen*), gegebenenfalls unter Einsatz von Platzhalterzeichen (*l?uf**) für die Trunkierung und damit Ausweitung des Suchmusters. Aus informatisch-formaler Perspektive fallen auch Recherchen nach anderen einfachen diskreten Elementen (Symbolen) – etwa Wortklassen- oder Phrasenbezeichnern (*Verb* bzw. *Verbalphrase*) – in diese Kategorie.

– Die Mehrwortsuche erlaubt das Spezifizieren mehrerer Einzelsymbole (z.B. *laufen* AND *Schule* oder *Verb* AND *Adjektiv*) als Suchmuster und liefert Belege für deren gemeinsames Auftreten innerhalb einer vordefinierten übergeordneten Struktur (Satz, Text etc.). Neben der Konjunktion (*AND*)

ist auch der Einsatz anderer logischer Operatoren wie *OR* (Disjunktion) oder *NOT* (Negation) möglich. Abfolge und Abstand der Einzelsymbole dürfen unspezifiziert bleiben, was im Einzelfall zu Lasten der Trennschärfe zwischen erwünschten und unerwünschten Ergebnissen geht.

– Die Phrasensuche hingegen arbeitet mit exakten linearen Abfolgen und unmittelbaren Abständen: Die Suchanfrage „*auf dem Laufenden*" etwa findet ausschließlich Vorkommen genau dieser Redewendung.
– Die komplexe musterbasierte Suche (engl. *complex pattern matching*) nutzt reguläre Ausdrücke für die präzise Spezifizierung abstrakter Strukturen. Beispielsweise passt die Suchanfrage (*Artikel*)? ((*Adjektiv*)* *Substantiv*) auf alle Sätze oder Phrasen, in denen ein Substantiv nach null oder beliebig vielen Adjektiven folgt und diese wiederum hinter einem Artikel stehen. Wird ein einheitliches Segmentierungskriterium (z.B. Wörter) beibehalten, so gestaltet sich auch die Mischung linguistischer Beschreibungsebenen formal problemlos, natürlichsprachlich ausformuliert z.B. als „Finde Belege für das unmittelbare Vorkommen des Substantivs *Wiese* nach einem Adjektiv mit der Lemmaangabe *grün* sowie möglicherweise weiteren Adjektiven".
– Die hierarchisch geschachtelte Suche schließlich kombiniert unterschiedliche Segmentierungsebenen – Laute, Morpheme, Wörter, Phrasen etc. – innerhalb einer Abfrage (z.B. Nominalphrase nach infinitem Verb mit Lemma *befehlen*). Auf diese Weise lassen sich Wortformen in beinahe beliebig komplex ausgebauten syntaktischen Strukturen oder Wörter mit ausgewählten Anlauten, Fugenelementen etc. auffinden. Hinzu kommen potenzielle Verschränkungen mit korpus- oder textspezifischen Sekundärdaten.

Ebenso variabel wie die Formulierung der Suchmuster gestalten sich in Abhängigkeit vom konkreten linguistischen Untersuchungsinteresse die Aufbereitung und Präsentation von Suchergebnissen. Zu nennen sind insbesondere folgende Varianten:

– Die Konkordanzanzeige präsentiert die gefundenen Wörter bzw. Wortfolgen in ihren unmittelbaren lokalen Kontexten. Diese können satzübergreifend aus einer bestimmten Anzahl linker und rechter Nachbarwörter (positionsgebundene n-Gramme, engl. *key word in context* = *KWIC*) bestehen oder auch aus kompletten syntaktischen Strukturen (z.B. Sätze).
– Die Annotationsanzeige integriert phänomenrelevante Annotationen in die Kontextansicht. Die Metadaten – etwa Angaben zu morphologischen, syntaktischen oder semantischen Kategorien der Belegwörter – werden dabei entweder im Inline-Verfahren in den Primärtext integriert oder tabellarisch angezeigt (vgl. das Beispiel in Abschnitt 6.3), was Begrenzun-

gen aufgrund der dadurch rasch ansteigenden Anzeigekomplexität zur Folge hat.

– Die statistische Anzeige präsentiert deskriptive frequenz- bzw. verteilungsbasierte Regularitäten der Fundstellen und verdichtet diese ggf. zu Tabellen oder Grafiken (vgl. Abschnitt 6.3). In diese Kategorie fallen auch statistische Spezialauswertungen, etwa die Berechnung diskontinuierlicher n-Tupel als Kookkurrenzen höherer Ordnung (vgl. Belica 2011). Derartige Statistiken dienen als empirische Grundlage für linguistische Interpretation und Plausibilitätsprüfungen sowie für weitere explorative Phänomenanalysen.
– Die lexikografische Anzeige eignet sich in erster Linie für Einzelwort- oder Phrasensuchen. Sie kumuliert vielfältige Informationen der Fundstellen (Schreibvarianten, typische Verwendungskontexte, grammatische Eigenschaften, Etymologie, semantische Relationen, absolute oder relative Frequenzwerte etc.) und vermittelt damit sprachliches Wissen ähnlich wie traditionelle Wörterbücher, allerdings ohne deren inhärente Beschränkung der Lemmastrecke.

Bei der Abschätzung der technologischen Anforderungen an eine Korpusrechercheschnittstelle gilt es zu beachten, dass abstrakte Suchmuster aus Primär- und Sekundärdaten sowie die Kalkulation von Verteilungen o.Ä. für umfangreiche Trefferlisten grundsätzlich deutlich rechenintensiver ausfallen als einfache Belegsuchen oder das Auffinden von Kollokationen, Häufigkeitsklassenzuordnungen etc. einzelner Treffer. Neben methodischen Konsequenzen impliziert dieser Umstand auch unterschiedliche Toleranzgrenzen in der Mensch-Maschine-Interaktion: Während die lexikografisch orientierte Präsentation von Einzeltreffern ohne nennenswerte Zeitverzögerung erwartet (und auf zeitgemäßen Systemen auch realisiert) wird, sind für multidimensionale linguistische Analysen durchaus deutlich längere Suchzeiten hinnehmbar.

Zu entscheiden ist weiterhin, ob die Ergebnispräsentation musterbasiert erfolgen soll oder ob für die statistische Beurteilung ein bestimmter Bezugsrahmen wie Tokenanzahl, Satzanzahl etc. erwartet wird. Dies betrifft etwa diejenigen Fälle, in denen ein Phänomen mehrmals im selben Satz vorkommt, also beispielsweise das Suchmuster „Adjektiv an beliebiger Position vor Substantiv“ in *Kinder spielen auf der großen grünen Wiese*. Das gesuchte Muster tritt im Satz zweimal auf (*großen* vor *Wiese*, *grünen* vor *Wiese*); als numerisches Ergebnis ist in Abhängigkeit von der konkreten Untersuchungsfrage also entweder „2“ (Anzahl der Muster) oder „1“ (Anzahl der Belegsätze) möglich. Sollen absolute oder relative Gebrauchshäufigkeiten berechnet werden, muss das Retrievalsystem die gewünschten Häufigkeitsmaße kennen und einbeziehen. Mög-

liche Varianten sind hier Anzahl der Wortformen oder aber Anzahl der Sätze im Gesamtkorpus, die entsprechenden Retrievalergebnisse zur weiteren statistischen Auswertung lauten dann: „2 Treffer in einem Korpus aus x Token" bzw. „1 Treffer in einem Korpus aus y Sätzen".

Für die Realisierung all dieser Erfordernisse setzen die drei großen deutschsprachigen Korpusportale DeReKo/COSMAS, LCC/Wortschatz und DWDS auf unterschiedliche korpustechnologische Design- und Implementierungsstrategien. Diese bilden nicht immer sämtliche Such- und Präsentationsvarianten in Gänze ab. In der Summe ergibt sich jedoch ein mutmaßlich repräsentatives Gesamtbild zum derzeitigen „Stand der Kunst".[70]

2.3.1 DeReKo/COSMAS

Die Anfänge des Deutschen Referenzkorpus (DeReKo) am Mannheimer Institut für Deutsche Sprache (IDS) liegen in den Sechzigerjahren des 20. Jahrhunderts. Seinerzeit von Paul Grebe und Ulrich Engel als vergleichsweise bescheidene Sammlung elektronischer Texte – anfangs noch auf Lochkarten – initiiert, wurde die Ressource über die nachfolgenden Jahrzehnte hinweg auch unter den Bezeichnungen „Mannheimer Korpora", „IDS-Korpora", „COSMAS-Korpora" oder „Archiv der Korpora geschriebener Gegenwartssprache am IDS" bekannt.[71] Der Umfang erhöhte sich fortlaufend:

– Das Mannheimer Korpus I von 1967 enthielt ca. 2,2 Millionen laufende Wortformen in 293 Texten aus Belletristik, Trivialliteratur, wissenschaftlicher Literatur und Publikumspresse.

[70] Daneben existiert eine Vielzahl weiterer Korpusrecherchetools unterschiedlicher Mächtigkeit und Fokussierung, deren umfassende Evaluation für Korpora im Viel-Milliarden-Token-Bereich allerdings z.T. noch aussteht. Für Baumbanken sind hier tgrep, tgrep2 oder tregex zu nennen (z.B. Rohde 2005), weiterhin VIQTORYA (Steiner/Kallmeyer 2002), TIGERSearch (Lezius 2002), Erweiterungen der Corpus Workbench (CWB; Evert/Hardie 2011, 2015), CorpusSearch Randall 2009 oder FSQ (Kepser 2003). Für andere annotierte Formate wurden beispielsweise LPath (Lai/Bird 2005), die NITE Query Language (NQL; Carletta et al. 2005) oder SPLICR (Rehm et al. 2009) als experimentelle Plattform für die integrierte Abfrage heterogener Sprachdaten konzipiert. Im Rahmen des SFB 632 stellt ANNIS (ANNotation of Information Structure; Dipper et al. 2004; Krause/Zeldes 2014) eine datenbankbasierte Lösung für Korpora im PAULA-Format (Potsdamer AUstauschformat Linguistischer Annotationen) bereit, incl. der Abfragesprache ANNIS Query Language (AQL) und einer grafischen Oberfläche (https://korpling.german.hu-berlin.de/annis3-snapshot/). Auch das freie, ursprünglich als Klon des kommerziellen Intex-Produkts entwickelte Unitex Paumier 2003, Poliqarp (http://poliqarp.sourceforge.net; Janus/Przepiórkowski 2007) oder die von Sketch Engine (Kilgarriff et al. 2014) genutzte Abfragesprache CQL (Jakubíček et al. 2010) sowie die Manatee Corpus Engine (Rychlý 2007) lassen sich für komplex formulierte Korpusrecherchen nutzen.

[71] Vgl. Kupietz et al. (2010, 2014); Kupietz/Keibel (2009).

- Als 1991 die Recherchesoftware COSMAS (Corpus Search, Management and Analysis System) ihren Betrieb aufnahm, hatte sich die Zahl der am IDS vorgehaltenen Korpuswörter bereits auf ca. 20 Millionen verzehnfacht.
- Bis zum Abschluss des vom Land Baden-Württemberg finanzierten Kooperationsprojekts DeReKo-I im März 2002 wurden ca. 993 Millionen Textwörter archiviert.
- 2003, zum Start des Retrievalsystems COSMAS II, umfasste DeReKo ca. 2 Milliarden Wortformen.
- 2013 ist DeReKo bei über 6 Milliarden Textwörtern angelangt. Die meisten Inhalte sind online abfragbar, ein Teilbereich aus lizenzrechtlichen Gründen jedoch nur IDS-intern.
- Bis 2018 vervielfachte sich der Umfang aufgrund zahlreicher Neuakquisitionen auf ca. 42 Milliarden Textwörter (vgl. Abb. 4 sowie Kupietz/Lüngen 2014; Kupietz/Schmidt 2015), womit sich DeReKo zur weltweit umfangreichsten mehrfach-annotierten Sammlung deutschsprachiger Texte für sprachwissenschaftliche Untersuchungen entwickelt hat.

Die angemessene Dokumentation des öffentlichen Schriftsprachgebrauchs erfolgt durch eine Abdeckung unterschiedlicher Textsorten, Genres und Domänen. Hierzu gehören laufende Jahrgänge zahlreicher Zeitschriften und Zeitungen, Fachtexte, literarische Texte, Redenprotokolle usw. Dabei soll DeReKo in seiner Gesamtheit weder ausgewogen noch repräsentativ sein. Vielmehr strebt es eine Streuung bezüglich potenziell relevanter Strata sowie die Maximierung seiner Größe an und erlaubt in der konkreten Korpusnutzungsphase eine flexible temporäre Zusammenstellung virtueller Subkorpora. Damit kann es als „Urstichprobe“ (Perkuhn et al. 2012, S. 49) zur Generierung passgenauer Analysegrundlagen für spezielle wissenschaftliche Fragestellungen bezeichnet werden.

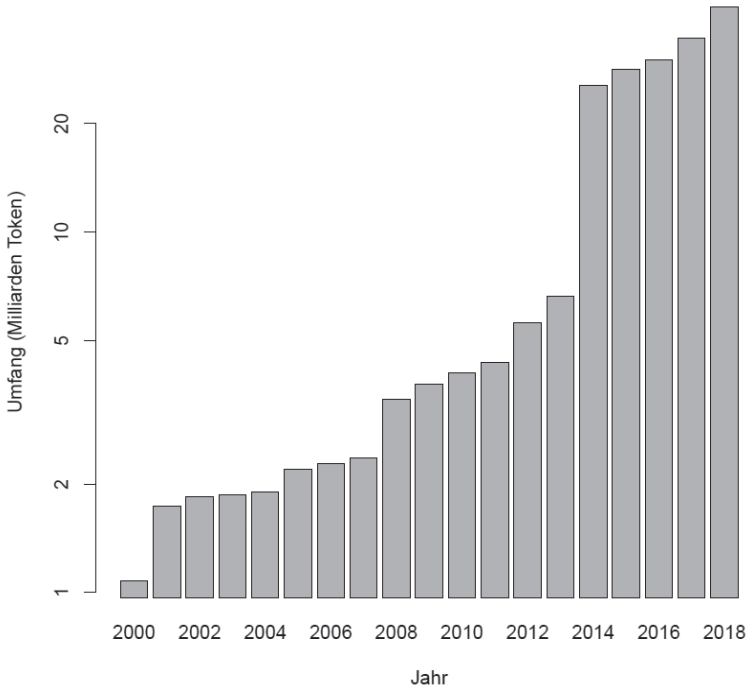


Abb. 4: Entwicklung des DeReKo-Gesamtumfangs (logarithmische Darstellung)

DeReKo belegt Sprachgebrauch aus Gegenwart und jüngerer Vergangenheit ab ca. 1955, wobei Materialien für den Zeitraum seit der Computerisierung in den 1980er/90er Jahren deutlich überwiegen. Sämtliche Texte werden von den ihren Rechteinhabern (Autoren, Verlage) explizit lizenziert. Anschließend erfolgt eine ungekürzte und zusammenhängende Speicherung, so dass sich auf Textebene auftretende empirische Phänomene (z.B. Zipf-Mandelbrot-Gesetz, Menzerathsches Gesetz, Piotrovskiy-Altmann-Gesetz) später fachgerecht untersuchen lassen. Die Konversion aus den diversen Originalformaten geschieht über spezielle Zwischenformate. Dabei finden eine maschinelle sowie partiell auch eine manuelle Qualitätskontrolle statt, die allerdings orthografische Fehler in den Originaltexten bewusst nicht korrigieren. Nach Abschluss dieser Aufbereitungsphase entsprechen sämtliche Sprachbelege einem einheitlichen Strukturformat (*IDS-XCES* bzw. ab 2014 *IDS-TEI P5*; vgl. Lüngen/Sperberg-McQueen 2012), das Inhalt und Struktur der Primärtexte originalgetreu abbildet. Die Gesamtstruktur ist hierarchisch organisiert: DeReKo setzt sich aus diversen Teilkorpora zusammen, deren Einteilung auf Metaangaben wie Quelle und/oder Jahr basiert, und die in sogenannten Archiven („Archiv der geschrie-

benen Korpora“ als Hauptarchiv) organisiert sind. Ein Korpus wiederum besteht aus einem oder mehreren Dokumenten; ein Dokument umfasst potenziell mehrere Texte, etwa Zeitungsartikel oder Buchkapitel.

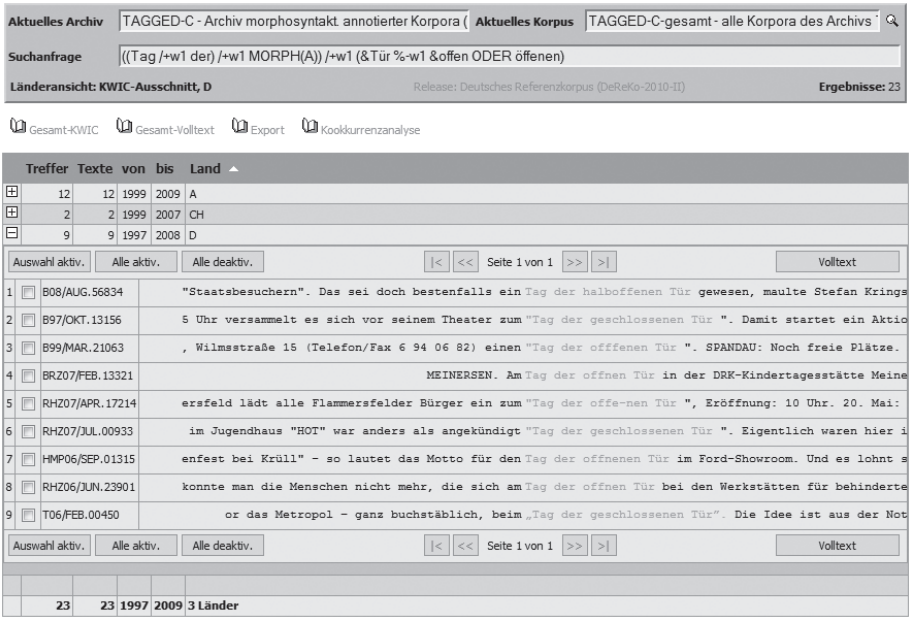


Abb. 5: COSMAS-Abfragespezifikation sowie nach Publikationsland geordnete KWIC-Anzeige

Als Ausgangspunkt für empirische Sprachstudien werden DeReKo-Inhalte mit umfangreichen Sekundärdaten angereichert. Hierzu zählen textspezifische Metadaten wie Publikationsjahr oder -ort, mithilfe einer Taxonomie strukturierte Angaben zur Textsorte sowie ebenfalls taxonomisch organisierte thematische Klassifikationen.[72] Seit Mitte der 1990er Jahre erfolgt eine maschinelle Anreicherung um linguistisch motivierte Standoff-Annotationen, anfangs mit dem *Source Tagger* der Firma Logos sowie den Tools *gercg* und *gertwol* der Firma Lingsoft. Mittlerweile sind die Primärinhalte in unterschiedlichem Ausmaß mit Tagging-Werkzeugen (z.B. *Connexor Machinese Phrase Tagger* und *TreeTagger*, teilweise und für interne Testzwecke auch *Xerox Incremental Parser*) morphosyntaktisch annotiert.[73] Gemeinsam nehmen Primär- und

[72] Vgl. Klosa et al. (2012) sowie Keibel/Belica (2007); Weiß (2005).

[73] Vgl. Belica et al. (2011). Die parallele Annotation mit Hilfe unterschiedlicher Werkzeuge ermöglicht einen systematischen Vergleich der Ergebnisse und Besonderheiten der automatischen Tagger/Parser und soll darauf aufbauend möglichst theorieneutrale bzw. -übergreifende Untersuchungen gestatten. Zur eingesetzten Annotationssoftware vgl. auch Stadler (2014, S. 13 ff.) sowie die jeweiligen Dokumentationen unter www.connexor.com bzw. www.

Sekundärdaten Speicherplatz im zweistelligen Terabyte-Bereich ein, wobei sich das Datenvolumen durch die kontinuierliche Akquise zusätzlicher Korpusinhalte zukünftig weiter erhöhen wird. Neben dem fortlaufenden Monitoring von Periodika sind die Erfassung weiterer Mediengattungen (z.B. Online-Inhalte oder verschriftete mündliche Kommunikation) ebenso wie die Anreicherung um zusätzliche Annotationsebenen geplant.

DeReKo ist lizenzrechtlich bedingt größtenteils nicht per Download verfügbar. Der Online-Zugang für derzeit über 30.000 registrierte Nutzer erfolgt über das Recherchesystem COSMAS II mit eigener Abfragesprache. Dort lassen sich einfache Suchanfragen nach Wortformen oder Wortbestandteilen ebenso wie nach Grundformen (Lemmata), Wortklassen (hier hilft auf Wunsch ein „Morph-Assistent“) oder grammatikalischen Kategorien (z.B. Numerus, Kasus, Tempus, sofern im recherchierten Teilkorpus annotiert) formulieren. Die dem Retrieval zugrunde liegende Indizierung erfolgt unter Verwendung des Open Source-Systems „MG (Managing Gigabytes)“.[74] Mit Hilfe der logischen Operatoren „und“/„oder“/„nicht“ sowie festlegbarer Wort-, Satz- bzw. Absatzabstände („/+w1“ für ein Wort, „/+s2“ für zwei Sätze etc.) oder Positionsangaben (z.B. „am Satzanfang“) lassen sich auch komplexe Konstrukte aufspüren. Die Recherche liefert zunächst selektierbare Wortformenlisten für die einzelnen Suchbestandteile. Die finalen Ergebnisse lassen sich zeitlich, geografisch oder thematisch sortieren (vgl. Abb. 5). Als Präsentationsformat ist entweder eine Volltextansicht oder KWIC wählbar, beide mit variablen linken und rechten Kontextgrößen und -maßen (Buchstabe, Wort, Satz oder Absatz); Treffer können alphabetisch, chronologisch oder nach Kookkurrenzstärken angeordnet werden. Exportformate sind ASCII oder RTF, ergänzend zum Kontext und den Quellennachweisen der Treffer exportiert COSMAS II auch die zur Abfrage generierten Expansionslisten.

Funktionsweise und Mächtigkeit der COSMAS II-Schnittstelle dokumentiert folgende Suchanfrage:

```
((Tag /+w1 der) /+w1 MORPH(A)) /+w1 (&Tür %-w1 &offen ODER öffenen)
```

Dieser komplexe Suchausdruck liefert Belege für die Wortgruppe „*Tag der*“, gefolgt von einem Adjektiv, wiederum gefolgt von einer beliebigen Deklinationsform von „*Tür*“. Die Grundform des Adjektivs darf nicht „*offen*“ sein, ebenso wird der mutmaßliche Tippfehler „*öffenen*“ ausgeschlossen.

---

cis.uni-muenchen.de/~schmid/tools/TreeTagger/. Nicht alle Annotationsergebnisse sind öffentlich verfügbar.

[74] Zur Online-Recherche mit COSMAS II vgl. Bodmer (2005). Die Indizierungs-, Abfrage- und Kompressionstechniken von MG dokumentiert Witten et al. (1999).

Eine integrierte Abfragehistorie erlaubt die Überprüfung oder Modifizierung bereits gestellter Korpusabfragen. Außerdem können häufig verwendete Gebrauchsmuster durch eine angeschlossene Kookkurrenzanalyse aufgedeckt werden. Reguläre Ausdrücke lassen sich in COSMAS II – sicherlich nicht zuletzt aufgrund der hohen DeReKo-Datenmenge, der Indizierungsproblematik bei nicht explizit realisierten Textstrings und der damit verbundenen langen Antwortzeiten für hochfrequente formale Suchmuster – nicht unmittelbar für das Pattern Matching einsetzen.

2.3.2 Deutscher Wortschatz/Leipzig Corpora Collection

Das Leipziger Wortschatz-Portal dient der Erschließung von öffentlich zugänglichen Korpusbelegen und statistischen Analysen der in der Datenbasis versammelten über 250 Einzelsprachen. Es basiert auf Methoden und Algorithmen, die seit Anfang der 1990er Jahre im Projekt „Deutscher Wortschatz" zunächst für das Deutsche und sukzessive auch für andere Sprachen erarbeitet wurden. 2006 erfolgte die Einführung der Bezeichnung „Leipzig Corpora Collection (LCC)" für den Gesamtdatenbestand, der in sprach-, genre- und zeitspezifische Subkorpora aufgeteilt ist.[75] Potenzielle Einsatzgebiete sind empirisch fundierte Studien zu linguistischen Phänomenen, sprachtypologische Betrachtungen, monolinguale oder kontrastive Wörterbucharbeit sowie NLP-Anwendungen wie Wissensextraktion, Identifizierung semantischer Relationen, Maschinelle Übersetzung oder die Unterstützung von Suchmaschinen beim Information Retrieval. Zur Abdeckung dieser Aufgaben wird das Korpusportal kontinuierlich inhaltlich und technologisch erweitert. Die nachfolgenden Kennzahlen beziehen sich auf den deutschsprachigen Teilbestand:

- Seit 1996 erfolgt die relationierte Speicherung sämtlicher Primär- und Sekundärinhalte in einem Datenbankmanagementsystem, wobei die Satzebene als maximale Fragmentgröße dient.
- 1998 umfasste die deutschsprachige Wortschatz-Datenbank ca. 3 Millionen Sätze. Bei einer durchschnittlichen Satzlänge von 15 Wörtern entsprach dies ca. 50 Millionen laufenden Wortformen. Als hauptsächliche Quellen dienten Zeitungstexte.
- 2003 umfasste der durch kontinuierliches Zeitungsmonitoring sowie Web-Crawling in deutschen und schweizerischen Domänen erweiterte Datenbestand bereits ca. 35 Millionen Sätze mit ca. 600 Millionen Wortformen.

[75] Vgl. Quasthoff (1998); Quasthoff et al. (2015) zum Wortschatz-Projekt sowie Biemann et al. (2007); Goldhahn et al. (2012); Kuras et al. (2018); Quasthoff et al. (2006); Richter et al. (2006) zu Aufbau und Abfrage der LCC-Inhalte.

– Nachdem seit 2007 neben Zeitungs- und Webtexten auch Wikipedia-Inhalte regelmäßig erfasst werden, versammeln die deutschsprachigen LCC-Korpora 2014 insgesamt ca. 5 Milliarden laufende Wortformen in über 300 Millionen Sätzen.
– Bis 2016 hat sich das Korpusvolumen auf ca. 10 Milliarden Wortformen in ca. 650 Millionen Sätzen verdoppelt, mit weiterhin stark ansteigender Tendenz.

Auf die initiale Textakquise folgt eine standardisierte mehrstufige Datenaufbereitung (*corpus processing toolchain*). Diese beginnt mit der Konvertierung des Quellformats (zumeist HTML) in Klartext (*plain text*) mit Hilfe des selbstentwickelten Tools „HTML2TEXT" unter Ausfilterung von Markup-Code, eingebetteten Metadaten, Kommentaren und nicht-textuellen Bestandteilen (Bilder, Tabellen etc.). Markup-Positionsinformationen zur Begrenzung von Blockelementen (z.B. von Absätzen) fließen allerdings später in die Satzgrenzenbestimmung ein. Ein statistischer Spracherkenner („LangSepa") analysiert die Verteilung hochfrequenter Stoppwörter und zeichenbasierter n-Gramme zwecks Bestimmung der Dokumentensprache. Darauf aufbauend werden Satzgrenzen – unter Verwendung sprachspezifischer Regeln, Satzendezeichenlisten und Abkürzungslisten – segmentiert; weiterhin findet eine Tokenisierung statt. Abschließend durchlaufen die einzelnen Sätze einen Säuberungsalgorithmus, der unerwünschte Daten nach formalen Kriterien ausfiltert. Dabei werden Dubletten ebenso entfernt wie fremdsprachige Sätze. Nicht erwünscht sind weiterhin Sätze, die bestimmte andere Auffälligkeiten aufweisen: mehr als sechs Einzelzeichen (Hinweis auf Sperrsatz, d.h. die Verwendung von Leerzeichen zur H e r v o r h e b u n g), mehr als eine vorgegebene Punkt-, Komma- oder Leerzeichenanzahl (Hinweis auf tabellarische Daten) usw. Als „non standard language" (Quasthoff et al. 2006, S. 1800) bzw. für die weitere statistische Auswertung uninteressant werden auch Sätze eingestuft, die multiple Satzzeichenverwendung oder bestimmte Sequenzen von Sonderzeichen sowie überlange Großbuchstaben- und Ziffernreihungen aufweisen. Sämtliche LCC-Werkzeuge für die Korpusvorverarbeitung sind unter einer Creative Commons-Lizenz als Download verfügbar.

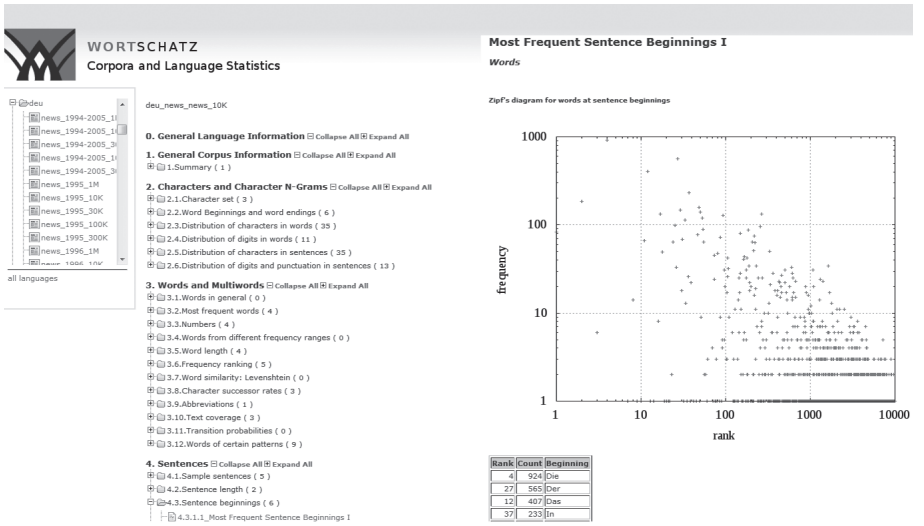


Abb. 6: Visualisierung korpusspezifischer Statistiken auf http://cls.informatik.uni-leipzig.de

Die LCC konzentriert sich auf moderne Gegenwartssprache, die ersten erfassten deutschsprachigen Texte datieren von 1994. Neben den fortlaufend archivierten und dadurch in der Sammlung prominentesten Genres (Web, News, Wikipedia) enthält die Sammlung eine Auswahl an Handbüchern sowie freie Literatur aus dem Projekt Gutenberg-DE. Neben Sprache und Genre existieren weitere Sekundärdaten, die etwa durch maschinelle Sachgebietsbestimmungen, Wortklassen-Tagging und Grundformanalyse, die Markierung von Eigennamen oder die Berechnung semantischer Ähnlichkeiten generiert werden und unmittelbar in die Online-Präsentation des Wortschatz-Portals einfließen. Ferner wird für jedes Subkorpus ein ausführliches Set statistischer Auswertungen durchgeführt, die über den einzelsprachlichen Nutzwert hinaus eine Basis für detaillierte Sprachvergleiche darstellen. Zu den ermittelten Parametern zählen zeichenbasierte Statistiken (zu Wortanfängen und -endungen, zur Verteilung von Buchstaben, Ziffern oder Sonderzeichen in Wörtern oder Sätzen etc.), Wort- und Wortgruppenstatistiken (Frequenzen, Längen, Levenshtein-Distanz für Wortähnlichkeit usw.), Satzanalysen (Längen, signifikante Anfänge und Endungen, Satzähnlichkeiten usw.), Kookkurrenzberechnungen auf Wort- und Satzebene sowie statistische Analysen der Textquellen (Anzahl, Größen, Eigenschaften). Abbildung 6 illustriert die Visualisierung einzelner statistischer Werte und Verteilungen.

Automatisierte Analysen des LCC-Datenmaterials stützen sich auf eine konsistente Speicherung sämtlicher Korpusinhalte in einem relationalen Datenbankmanagementsystem (*mySQL*) mit wohldefinierten Schnittstellen. Der Sammlung liegt ein sprachübergreifendes, homogenes Systemdesign zugrunde; Primär- und Sekundärdaten sowie Statistiken für jedes einzelsprachliche Subkorpus folgen einer vorgegebenen Tabellen- und Indexstruktur. Zu den Besonderheiten des physischen Datenbankschemas zählen die Speicherung von Positionsangaben einzelner Wortformen im Satz sowie die Nutzung eindeutiger numerischer Identifikatoren als Ersatz für konkrete Textwörter. Letztere werden, gemeinsam mit Frequenzangaben, in einer separaten Lookup-Tabelle referenziert und bei Bedarf über ihren Primärschlüssel in Queries einbezogen. Die Nutzung von ausschließlich numerischen – anstelle von alphanumerischen – Tabellen- bzw. Indexattributen für vollständig balancierte B-Bäume (engl. *B-trees*) soll vergleichsweise effiziente Indexstrukturen und mithin optimierte Abfragezeiten befördern.

In der LCC-Relationierung bleiben strukturelle Hierarchien und andere Beziehungstypen im zwischen Satz- und Korpusebene angesiedelten Bereich bewusst unabgebildet. Informationen zur Textstrukturierung in Absätze oder Kapitel werden ebensowenig erfasst wie satzübergreifende linguistische Phänomene. Diese Entscheidung beruht in erster Linie auf urheberrechtlichen Erwägungen. Die Segmentierung in maximal satzgroße Fragmente einerseits sowie die komplette Auslassung vereinzelter Sätze aufgrund der oben beschriebenen Restriktionen während der Datenaufbereitung andererseits machen eine Rekonstruktion kompletter Originaltexte de facto unerreichbar. Auf diese Weise begegnet die LCC potenziellen Copyright-Restriktionen und kann unter einer Creative Commons-Lizenz zur Verfügung gestellt werden. Diverse wort-, satz- und korpusbezogene Phänomene lassen sich nichtsdestotrotz verlässlich beschreiben. In wenigen Fällen allerdings limitiert die gewählte Strategie die empirische Aussagekraft der Korpussammlung: Beispielsweise lassen sich auf Häufigkeitsverteilungen oder Längenbestimmungen basierende Gesetzmäßigkeiten grundsätzlich nur für intakte Objekte nachweisen. Deshalb gilt es, bei der Interpretation quantitativer Messungen den Wegfall der Textebene und die Vorab-Extraktion unerwünschter Sätze zu beachten. Weiterhin bleiben transphrastische Untersuchungen – für die ganzheitliche Analyse von Diskursphänomenen, textgrammatischen Phänomenen o.Ä. – vom Anwendungsspektrum ausgenommen.

LCC-Korpusinhalte lassen sich über drei Zugangswege nutzen: (i) per Download, (ii) mit Hilfe serverseitig implementierter Webdienste, (iii) über das Wortschatz-Portal. Für die erstgenannte Variante werden pro Subkorpus mehrere

Ausschnitte in gestaffelten Normgrößen und mit komplementärer Satzauswahl angeboten. Diese Downloads enthalten, sofern der Gesamtumfang des jeweiligen Subkorpus dies zulässt, 10.000, 30.000, 100.000, 300.000 bzw. 1 Millionen zufällig ausgewählte Sätze; bei entsprechenden Subkorpusgrößen sind auch 3 oder 10 Millionen Sätze verfügbar. Als Exportformat stehen ANSI-Text sowie *mySQL*-Exportfiles zur Auswahl, die nach dem Download in eine Datenbank geladen oder mit korpuslinguistischen Werkzeugen ausgewertet werden können. Alternativ hierzu agieren die LCC-Webdienste durch eine Umkehr des Datenflusses: Sie transportieren nicht die kompletten Korpusdaten zum Nutzer, sondern kommunizieren einen konkreten Nutzungswunsch an die Datenbank und liefern via SOAP-Schnittstelle passgenaue Ergebnisse. Die angebotenen Dienste sind in drei Rangstufen unterteilt. Dienste der Stufe 1 stellen grundlegende linguistische Abfragen zur Verfügung, also z.B. die Bestimmung von Grundform, Wortklasse, Frequenz oder Synonymen zu einer Wortform, die Berechnung statistisch signifikanter Wortnachbarn oder die zufallsbasierte Auswahl von Belegsätzen. Dienste der Stufe 2 umfassen komplexere Retrievaloperationen und Analysen, beispielsweise Kookkurrenzschnitte für mehrere Eingabewörter, die aufgrund des höheren Rechenzeitaufwands registrierten Nutzern vorbehalten sind. Dienste der Stufe 3 schließlich umfassen experimentelle oder besonders rechenaufwändige Abfragen und werden üblicherweise in Absprache mit externen Kooperationspartnern eingerichtet.

Das eigentliche Wortschatz-Portal präsentiert sich unter Nutzungsaspekten eher wörterbuchorientiert. Recherchiert wird in monolingualen, nach Genre und Zeitraum kompilierten Korpora, für das Deutsche also beispielsweise in „News 1994-2000“ oder „Wikipedia 2012“. Ergänzend existiert für das Deutsche eine automatisiert berechnete Lemmastrecke tagesaktueller Termini („Wörter des Tages“), zu denen Belegstellen aus Tageszeitungen und Newsdiensten, Assoziationsgraphen sowie Häufigkeitsvergleiche bereitgestellt werden. Für das interaktive Online-Retrieval steht keine spezialisierte Korpusabfragesprache, sondern eine Freitexteingabe zur Verfügung. Deren Inhalt wird automatisiert in SQL-Queries integriert, Einzelwörter können dabei mit oder ohne Berücksichtigung von Groß-/Kleinschreibung nachgeschlagen werden. Platzhalterzeichen expandiert das Retrievalsystem gegen interne Wortlisten, flektierte Formen werden auf morphologische Stamm- bzw. Grundformen zurückgeführt. Neben der Einzelwortsuche ist auch die Recherche nach phraseologischen Wortverbindungen (*Radio hören, geht auf das Konto von*) möglich. Komplexere musterbasierte Suchausdrücke lassen sich über das standardisierte Online-Formular nicht spezifizieren. Es ist folglich keine kombinierte Suche nach morphosyntaktischen Kategorien, Wortklassen etc. möglich, ebensowenig eine Verkettung mehrerer Suchbegriffe mit logischen Operatoren.

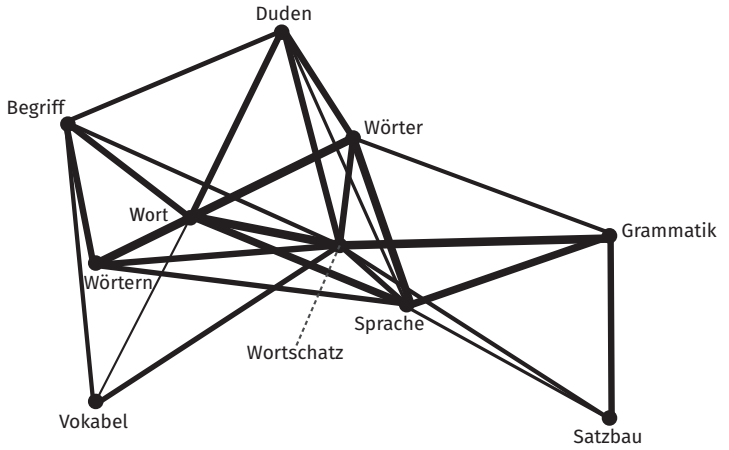


Abb. 7: Visualisierung von Nachbarschaftskookkurrenzen durch das Wortschatz-Portal

Die Darstellung der Suchergebnisse erfolgt primär aus lexikografischer Perspektive, d.h. in einem synoptischen Artikelformat mit Querverweisen. Die einzelnen wortbezogenen Informationen (Frequenzdaten im Artikelkopf, weiterhin Textsorten- und Fachgebietsklassifizierungen, grammatische Angaben, Synonyme, Unter- und Oberbegriffe, Dornseiff-Bedeutungsgruppen etc.) werden kumuliert und gemeinsam mit exemplarischen Textbelegen (weitere Beispielsätze lassen sich per Mausklick aus der Korpusbasis extrahieren) sowie Kookkurrenzen aufgelistet. Letztere erscheinen darüber hinaus visualisiert in Form eines Assoziationsgraphs bzw. einer „Semantic Map“ (siehe Abb. 7). Insgesamt bedient das Wortschatz-Portal täglich ca. 40.000 Abfragen deutscher Wörter.

2.3.3 DWDS

Die Textsammlungen der Berlin-Brandenburgischen Akademie der Wissenschaften (BBAW) stellen das empirische Fundament des online abfragbaren Digitalen Wörterbuchs der deutschen Sprache (DWDS) dar.[76] Hinsichtlich ihrer Größenordnung entsprechen die DWDS-Kernkorpora dem British National Corpus (BNC), und auch ihre Stratifizierung orientiert sich an ähnlichen Prinzipien. So wird durch eine Gewichtung der enthaltenen Textsorten ein ausgewogener Querschnitt der deutschen Gegenwartssprache angestrebt. Jeweils

[76] Daneben baut das DWDS-Wortinformationssystem auf weiteren lexikalischen Quellen auf, etwa dem Wörterbuch der deutschen Gegenwartssprache (WDG), dem Etymologischen Wörterbuch des Deutschen (EtymWB) sowie dem Deutschen Wörterbuch (DWB) von Jacob und Wilhelm Grimm; vgl. Geyken/Klein (2010) und Geyken (2007).

ein gutes Viertel der Inhalte stammt aus Belletristik und Zeitungstexten, je ca. 20% verteilen sich auf Fach- bzw. Gebrauchstexte, ca. 5% basieren auf transkribierter Rede. Weiterhin wurde darauf geachtet, dass sämtliche Dekaden seit 1900 möglichst gleichmäßig in den Quellen vertreten sind. Im Unterschied zum BNC oder der LCC enthalten die DWDS-Korpora keine Textauszüge, sondern speichern auch bei umfangreichen Werken (z.B. Romanen) vorrangig vollständige Dokumente.

Das digitale DWDS-Archiv wird seit 2000, mit initialer Unterstützung der Deutschen Forschungsgemeinschaft (DFG), kontinuierlich ausgebaut. Die Textressourcen umfassen drei Hauptbestandteile:

– Das themenübergreifende, ausgewogene Kernkorpus des 20. Jahrhunderts enthält über 120 Millionen laufende Wortformen (incl. ca. 20 Millionen Interpunktionen und alphanumerischen Zeichenketten). Diese verteilen sich auf ca. 7 Millionen Sätze in knapp 80.000 Dokumenten.
– Eine in Konzeption und Quellenauswahl vergleichbare Ressource für das 21. Jahrhundert befindet sich im Entstehen und kann in Teilen ebenfalls bereits online abgefragt werden.
– Weitere Referenz-, Spezial- und Zeitungskorpora – z.B. das Korpus des deutschen Textarchivs (DTA), ein in Zusammenarbeit mit der Humboldt-Universität zu Berlin aufgebautes DDR-Korpus, das „Berliner Wendekorpus“ u.a. – mit einem Umfang von zusammengenommen ca. 13 Milliarden laufenden Wortformen ergänzen die Kernkorpora. Dabei stehen weniger Ausgewogenheit oder gar Repräsentativität der Sprachquellen als vielmehr die Abdeckung spezifischer Sprachumstände im Vordergrund des Archivdesigns.

Die DWDS-Korpora werden unter Beachtung der TEI-Richtlinien XML-basiert kodiert. Die formale Aufbereitung der Primärdaten geht einher mit einer Anreicherung um dokumentspezifische Metadaten wie Titel, Autor, Publikationsdatum oder Textsorte. Weiterhin werden die Textquellen durch größtenteils im Projektumfeld entstandene Software um linguistisch motivierte Annotationen angereichert. Das „TAGH“-Morphologiesystem[77] zerlegt dabei Komposita unter Nutzung gewichteter endlicher Transduktoren, führt flektierte Wortformen auf ihre Grundformen zurück und ermittelt semantische Lesarten von Substantiven. Darauf aufbauend integriert ein statistischer Wortarten-Tagger Hidden Markov Modelle zur Disambiguierung und Zuweisung von Wortklassen. Weiterhin werden Eigennamen regelbasiert ausgezeichnet sowie syntaktische Strukturen durch einen Dependenzparser („SynCoP“) annotiert.

[77] Vgl. Geyken/Hanneforth (2005) sowie http://tagh.de.

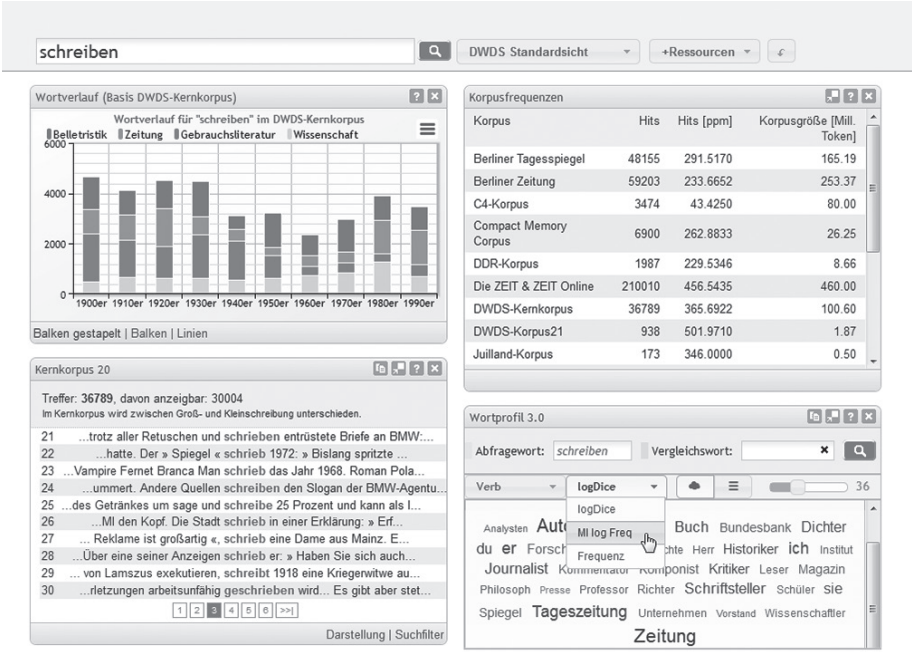


Abb. 8: Statistische Module („Panels“) und KWIC des DWDS-Portals

Im DWDS-Wortinformationssystem darf für nicht-kommerzielle Zwecke frei recherchiert werden. Aufgrund der mit einzelnen Textgebern abgeschlossenen Lizenzabkommen ist gegebenenfalls eine vorherige Registrierung notwendig, um urheberrechtlich geschützte Kontextbelege einzusehen. Derzeit verwaltet das System mehrere Zehntausend personifizierte Benutzerkennungen. Für die Online-Recherche kommt die linguistische Suchmaschine „DDC-Concordance“[78] zum Einsatz, auf deren Technik beispielsweise auch DWDS-Partnerprojekte wie das Schweizer Textkorpus oder das Online-Portal der Wochenzeitung „Die Zeit“ aufbauen. DDC erlaubt eine Indizierung von Textwörtern mit linguistisch motivierten Annotationen wie Lemma oder Wortklasse, weiterhin können XML-kodierte textspezifische Metadaten als Retrievalparameter einbezogen werden. Die Abfragesprache unterstützt Phrasen- und Einzelwortsuchen. Für die Verknüpfung von Einzelsymbolen bietet sie einen Abstandsoperator sowie die logischen Operatoren *AND* (&&), *OR* (||) und *NOT* (!); Platzhalterzeichen dienen der Links- oder Rechts-Trunkierung. Per modifizierbarer Voreinstellung werden flektierte Formen inkludiert, d.h. die Recherche nach *geht* findet auch *gehen, ging, gegangen* usw., allerdings prä-

[78] Vgl. Sokirko (2003); Informationen zum gleichnamigen Open Source-Projekt bietet www.ddc-concordance.org.

sentiert das Retrievalfrontend keine explizite Wortformenliste. Wortklassenwerte lassen sich als alleinige Suchparameter oder im Verbund mit Zusatzspezifikationen angeben: Die Spezifikation *$p=ART* findet sämtliche Artikelwörter in der Korpusbasis; „*$p=ART with @der*“ schränkt die Abfrage auf das Artikelwort *der* ein; „*$p=ART with @der #0 $p=ADJA #0 $p=NE*“ liefert Belege für das Suchmuster „Artikelwort *der* unmittelbar vor einem beliebig flektierten Adjektiv und einem Eigennamen“, also beispielsweise „*der überragende Philipp*“ oder „*der kleinen Schweiz*“. Weitere grammatische Kategorien wie Numerus, Kasus oder Tempus lassen sich nicht in die Abfragespezifikation einbauen, ebensowenig wie syntaktische Strukturen.

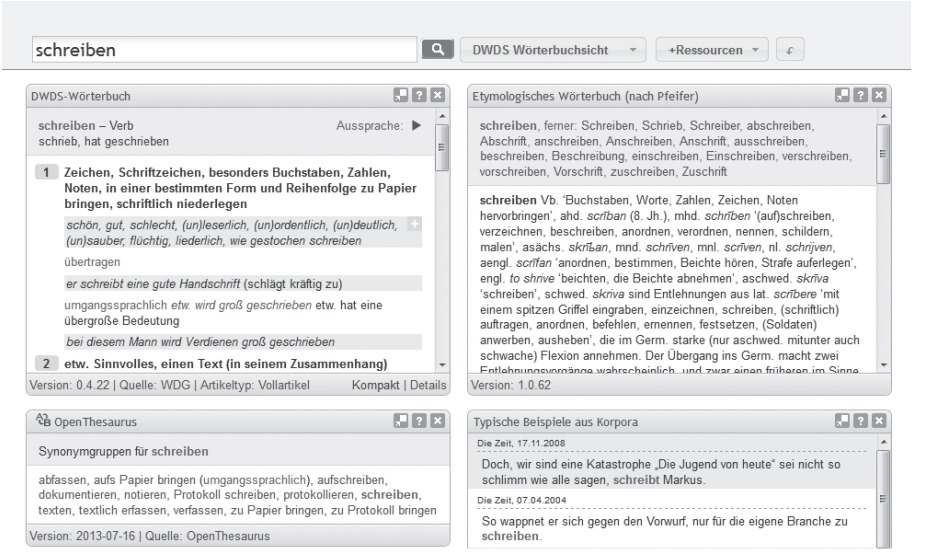


Abb. 9: Wörterbuch-Module des DWDS-Portals

Die Portaloberfläche ist als modulares System konzipiert; die DWDS-Standardansicht kombiniert wörterbuch- und korpusgestützte Analysen in sogenannten „Panels“. Diese beinhalten die zur aktuell gestellten Abfrage gehörigen Recherchen in den angeschlossenen Wörterbüchern, weiterhin für Einzelwörter ein statistisch generiertes Wortprofil mit signifikanten syntagmatischen Beziehungen als Schlagwortwolke oder in tabellarischer Form. KWIC-Ansichten sind nach Publikationsjahr, Textsorte, Autor und Titel filterbar und nach Datum oder Satzerst- bzw. Satzendstellung – jedoch nicht nach Kontextwörtern – sortierbar. Benutzer können die Standardsicht an ihre Bedürfnisse anpassen, indem sie gezielt einzelne Lexika, Korpora oder statistische Ressour-

cen in die Panel-Oberfläche integrieren. Für statistische Auswertungen bieten sich dabei neben dem Wortprofil auch die auf dem Kernkorpus basierende Wortverlaufsanalyse – ein Diagramm der Textsortenverteilungen und Vorkommenshäufigkeiten des Suchmusters über die zurückliegenden Dekaden – sowie Übersichten zu den einzelnen Korpusfrequenzen mit absoluten und relativen Häufigkeitsdaten an (vgl. Abb. 8).

Weitere vordefinierte Panelkombinationen bieten nutzungsspezifische Aggregationen von Rechercheergebnissen in den Kern- und Ergänzungskorpora. Eine speziell für lexikografische Studien konzipierte „Wörterbuchansicht" (vgl. Abb. 9) kombiniert die lexikalischen DWDS-Quellen mit Synonymgruppen des OpenThesaurus sowie einer auf empirischen und qualitativen Kriterien beruhenden Belegliste („Typische Beispiele"). Zu den Auswahlregeln dieser Belegliste zählen beispielsweise die Präferierung ausgesuchter Publikationen aus einer Setzliste, die möglichst ausgewogene Berücksichtigung unterschiedlicher Textsorten und Dekaden, sowie die Anwendung von Ausschlussmerkmalen wie überlange oder nicht vollständig mit dem DWDS-Dependenzparser analysierbare Sätze.

2.4 Multidimensionale Suchkriterien

Natürliche Sprache kann als multidimensionaler Forschungsgegenstand angesehen werden in dem Sinne, dass bei ihrer Produktion und Rezeption diverse sich potenziell wechselseitig beeinflussende Systembedürfnisse wirken, komplexe Konstruktionen und Konstituenten zum Einsatz kommen, sowie divergierende Perspektiven zu berücksichtigen sind. Um diesem Umstand Rechnung zu tragen, verwenden Sprachwissenschaftler sowohl bei der Beschreibung von Sprache als abstraktem System wie auch bei der konkreten Analyse von Sprachgebrauch verschiedenartige, in Teilen aufeinander aufbauende formale Ebenen. Hierzu zählen insbesondere:

- die Ebene der Lautstrukturen (Phonemebene),
- Ebenen des internen Wortaufbaus (z.B. Silben- oder Morphemebene),
- die Wortebene,
- wortübergreifende Strukturen (Wortgruppen, Phrasen etc.),
- die Satzebene sowie
- die Textebene.

Die Hierarchie dieser Beschreibungsebenen geht davon aus, dass elementare sprachliche Einheiten jeweils komplexere Einheiten höherer Ordnung konstituieren. Auf diese Weise lassen sich natürlichsprachliche Äußerungen segmentieren und paradigmatische Teil-Ganzes-Beziehungen (Meronymien) zwi-

schen Sprachkonstrukten modellieren: Silben können in Laute zerlegt werden und konstituieren ihrerseits Wörter, die wiederum in Phrasen- oder Satzstrukturen einfließen. Neben dem Prinzip der Konstituenz kommt – beispielsweise auf Satzebene – auch das Prinzip der Dependenz zum Einsatz, um mehr oder weniger ausgeprägte Anhängigkeiten zwischen Einheiten (etwa zwischen einem Verb und seinen Ergänzungen) zu darzustellen.

Die Beschreibungsebenen sind für diverse linguistische Disziplinen interessant: Text- und Diskurslinguisten untersuchen primär satzübergreifende Konstrukte, Grammatiker beschäftigen sich üblicherweise mit der Morphem-, Wort-, Wortgruppen- oder Satzebene,[79] Phonologen betrachten Lautstrukturen usw. Linguistisch interessant sind darüber hinaus Phänomene der Bedeutungsebene, also semantische Beziehungen wie syntagmatische Verträglichkeitsbeziehungen, paradigmatische Relationen (Synonymie, Antonymie etc.) oder Thema-Rhema-Verbindungen zur Beschreibung von Informationsstrukturen. Rhetorische Strukturen in Texten werden durch Ansätze wie die Rhetorical Structure Theory (RST; vgl. Mann/Thompson 1988) abgebildet.

Für viele dieser Segmente und Strukturen existieren computerlinguistische Ressourcen (Parser, Tagger, Lexika), die natürlichsprachliche Primärinhalte maschinell um entsprechende Annotationen anreichern. Zunehmend – z.B. zur methodenübergreifenden Analyse von Annotationsresultaten oder zur Evaluation der Leistungsfähigkeit einzelner Werkzeuge – entstehen dabei mehrfach annotierte Sprachdaten. Als Konsequenz daraus haben sich die Rahmenbedingungen für das Korpusretrieval in den letzten Jahren massiv geändert. Moderne Korpussysteme müssen mit dem Umstand umgehen, dass nicht nur einzelne Annotationsebenen, sondern multiple Segmentierungen und konkurrierender Tagger-Output (etwa bei der Tokensierung oder Satzgrenzenerkennung) für einzelne Beschreibungsebenen verwaltet und recherchierbar gemacht werden sollen. Hinzu kommen vielschichtige Metadaten zu Aspekten wie Publikationsort oder -zeit, die ebenfalls für multivariate Untersuchungen zu Aufbau und Funktion sprachlicher Äußerungen relevant sind (vgl. Abschnitt 2.1.2). Alle diese Sekundärdaten und deren hierarchischen Strukturen gilt es folglich bei der Konzeption von Abfragesystemen zu berücksichtigen. Die Zielsetzung empirisch ausgerichteter Sprachforschung besteht dann darin, sprachliche Phänomene als quantifizierbare Vorkommen in der multidimensionalen Datenstruktur zu formulieren und ggf. signifikante Häufigkeitsverteilungen ermittelbar zu machen.

[79] Tatsächlich nimmt der Bereich der Morphosyntax derzeit eine besonders prominente Stellung bei der Annotation und Analyse von Korpusinhalten ein, vgl. z.B. Truskkina (2004), Müller/Meurers (2006) oder die Übersicht zu Abfragemöglichkeiten existierender Systeme in Duffner/Näf (2006).

Konkrete Herausforderungen bei der gezielten Abfrage mehrfach segmentierter, hierarchisierter und annotierter Sprachbelege sollen nachfolgend exemplarisch aufgezeigt werden. Als Beispiel dient folgender Satz aus der DeReKo-Korpussammlung (Textsigle A00/SEP.64317):

> Im Gegenteil: Zum Kultobjekt aufgestiegen, feiert es heute sogar eine Renaissance. [St. Galler Tagblatt vom 21.09.2000, Ressort: TB-ABI (Abk.); Alte Rezepte zu neuem Leben erweckt]

Textspezifische Metadaten für diesen Beleg lassen sich entlang ausgewählter Dimensionen hinzufügen, z.B.:

- Medium: Publikumspresse
- Domäne: Kultur/Unterhaltung
- Region: Ostschweiz
- Jahr: 2000
- Autor: männlich

Satz-, phrasen- oder wortgebundene Sekundärdaten erfordern bereits komplexere, üblicherweise XML-basierte Darstellungsnotationen. Eine automatische Analyse mit dem *Connexor Machinese Phrase Tagger* etwa resultiert in folgender Standoff-Annotation:

```
<sentence>
       <token pos="4592" len="1">
              <text>I</text>
              <lemma>in</lemma>
              <tags syntax="@PREMARK" morpho="PREP"/>
       </token>
       <token pos="4593" len="1">
              <text>m</text>
              <lemma>das</lemma>
              <tags syntax="@PREMOD" morpho="DET"/>
       </token>
  <np>
       <token pos="4595" len="9">
              <text>Gegenteil</text>
              <lemma>gegenteil</lemma>
              <tags syntax="@NH" morpho="N"/>
       </token>
```

```
</np>
      <token pos="4604" len="1">
            <text>:</text>
            <lemma>:</lemma>
            <tags/>
      </token>
      <token pos="4606" len="2">
            <text>Zu</text>
            <lemma>zu</lemma>
            <tags syntax="@PREMARK" morpho="PREP"/>
      </token>
      <token pos="4608" len="1">
            <text>m</text>
            <lemma>das</lemma>
            <tags syntax="@PREMOD" morpho="DET"/>
      </token>
<np>
      <token pos="4610" len="10">
            <text>Kultobjekt</text>
            <lemma>kult objekt</lemma>
            <tags syntax="@NH" morpho="N"/>
      </token>
</np>
      <token pos="4621" len="12">
            <text>aufgestiegen</text>
            <lemma>aufsteigen</lemma>
            <tags syntax="@MAIN" morpho="V" sub1="PCP"
            sub2="PERF"/>
      </token>
      <token pos="4633" len="1">
            <text>,</text>
            <lemma>,</lemma>
            <tags/>
      </token>
      <token pos="4635" len="6">
            <text>feiert</text>
            <lemma>feiern</lemma>
            <tags syntax="@MAIN" morpho="V" sub1="IND"
            sub2="PRES"/>
      </token>
      <token pos="4642" len="2">
```

```
            <text>es</text>
            <lemma>es</lemma>
            <tags syntax="@NH" morpho="PRON"/>
      </token>
      <token pos="4645" len="5">
            <text>heute</text>
            <lemma>heute</lemma>
            <tags syntax="@ADVL" morpho="ADV"/>
      </token>
      <token pos="4651" len="5">
            <text>sogar</text>
            <lemma>sogar</lemma>
            <tags syntax="@ADVL" morpho="ADV"/>
      </token>
      <token pos="4657" len="4">
            <text>eine</text>
            <lemma>eine</lemma>
            <tags syntax="@PREMOD" morpho="DET"/>
      </token>
  <np>
      <token pos="4662" len="11">
            <text>Renaissance</text>
            <lemma>renaissance</lemma>
            <tags syntax="@NH" morpho="N"/>
      </token>
  </np>
      <token pos="4673" len="1">
            <text>.</text>
            <lemma>.</lemma>
            <tags/>
      </token>
</sentence>
```

Die vorstehende Notation lässt sich weitestgehend tabellarisch zusammenfassen. Zur Förderung der Übersichtlichkeit ignorieren wir dabei die im XML-Output enthaltenen Positions- und Längenangaben einzelner Wörter; letztere bleiben implizit in den Token-Werten enthalten. Ebenfalls rekonstruierbar bleiben Informationen zur Stellung der Wörter im Satz (z.B. lineare Abfolge, Satzerst- oder Endposition). Auffallend ist die Segmentierung der initialen Wort-

form „*Im*“ in zwei separate Token sowie darauf basierend die Bestimmung der zugehörigen Grundformen „*in*“ und „*das*“. Die Erkennung syntaktischer Konstruktionen erfolgt ausgesprochen rudimentär, als einzige phrasale Kategorie markiert Connexor die Nominalphrase (NP). NP-Chunks beinhalten explizit lediglich das Nomen, implizit sind auch die premodifizierenden Elemente (PREMARK/PREMOD) hinzuzurechnen.

Token	Grundform	Wortklasse	Subkategorien	Syntax
I	in	PREP		PREMARK
m	das	DET		PREMOD
Gegenteil	gegenteil	N		NH
:	:	P		
Zu	zu	PREP		PREMARK
m	das	DET		PREMOD
Kultobjekt	kult objekt	N		NH
aufgestiegen	aufsteigen	V	PCP PERF	MAIN
,	,	P		
feiert	feiern	V	IND PRES	MAIN
es	es	PRON		NH
heute	heute	ADV		ADVL
sogar	sogar	ADV		ADVL
eine	eine	DET		PREMOD
Renaissance	renaissance	N		NH
.	.	P		

Tab. 3: Zusammenfassung der Connexor-Annotation

Die parallele Annotation des Beispielsatzes mit *TreeTagger* produziert folgendes Ergebnis, das seinerseits in Tabelle 4 zusammengefasst wird:

```
<sentence>
  <lexeme id="955" pos="4592" len="2">
    <surface-form>Im</surface-form>
    <sense id="0">
      <base-form>im</base-form>
      <part-of-speech conf="1.000000">APPRART</part-of-speech>
    </sense>
```

```
  </lexeme>
  <lexeme id="956" pos="4595" len="9">
    <surface-form>Gegenteil</surface-form>
    <sense id="0">
      <base-form>Gegenteil</base-form>
      <part-of-speech conf="1.000000">NN</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="957" pos="4604" len="1">
    <surface-form>:</surface-form>
    <sense id="0">
      <base-form>:</base-form>
      <part-of-speech conf="1.000000">$.</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="958" pos="4606" len="3">
    <surface-form>Zum</surface-form>
    <sense id="0">
      <base-form>zum</base-form>
      <part-of-speech conf="1.000000">APPRART</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="959" pos="4610" len="10">
    <surface-form>Kultobjekt</surface-form>
    <sense id="0">
      <base-form>Kultobjekt</base-form>
      <part-of-speech conf="1.000000">NN</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="960" pos="4621" len="12">
    <surface-form>aufgestiegen</surface-form>
    <sense id="0">
      <base-form>aufsteigen</base-form>
      <part-of-speech conf="1.000000">VVPP</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="961" pos="4633" len="1">
    <surface-form>,</surface-form>
    <sense id="0">
      <base-form>,</base-form>
      <part-of-speech conf="1.000000">$,</part-of-speech>
```

```
    </sense>
  </lexeme>
  <lexeme id="962" pos="4635" len="6">
    <surface-form>feiert</surface-form>
    <sense id="0">
      <base-form>feiern</base-form>
      <part-of-speech conf="0.999680">VVFIN</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="963" pos="4642" len="2">
    <surface-form>es</surface-form>
    <sense id="0">
      <base-form>es</base-form>
      <part-of-speech conf="1.000000">PPER</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="964" pos="4645" len="5">
    <surface-form>heute</surface-form>
    <sense id="0">
      <base-form>heute</base-form>
      <part-of-speech conf="1.000000">ADV</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="965" pos="4651" len="5">
    <surface-form>sogar</surface-form>
    <sense id="0">
      <base-form>sogar</base-form>
      <part-of-speech conf="1.000000">ADV</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="966" pos="4657" len="4">
    <surface-form>eine</surface-form>
    <sense id="0">
      <base-form>eine</base-form>
      <part-of-speech conf="0.994514">ART</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="967" pos="4662" len="11">
    <surface-form>Renaissance</surface-form>
    <sense id="0">
      <base-form>Renaissance</base-form>
```

```
      <part-of-speech conf="1.000000">NN</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="968" pos="4673" len="1">
    <surface-form>.</surface-form>
    <sense id="0">
      <base-form>.</base-form>
      <part-of-speech conf="1.000000">$.</part-of-speech>
    </sense>
  </lexeme>
</sentence>
```

Token	Grundform	Wortklasse	Konfidenzwert
Im	im	APPRART	1.000000
Gegenteil	Gegenteil	NN	1.000000
:	:	$.	1.000000
Zum	zum	APPRART	1.000000
Kultobjekt	Kultobjekt	NN	1.000000
aufgestiegen	aufsteigen	VVPP	1.000000
,	,	$,	1.000000
feiert	feiern	VVFIN	0.999680
es	es	PPER	1.000000
heute	heute	ADV	1.000000
sogar	sogar	ADV	1.000000
eine	eine	ART	0.994514
Renaissance	Renaissance	NN	1.000000
.	.	$.	1.000000

Tab. 4: Zusammenfassung der TreeTagger-Annotation

Abgesehen von Tagset-bedingten Abweichungen in der Notation der Wortklassen (die STTS-Tags bezeichnen morphosyntaktische, syntaktische und gelegentlich auch semantische Merkmale), der ergänzenden Angabe eines Konfidenzwertes für die Wortklassenzuordnung sowie der nicht durchgeführten syntaktischen Wortgruppenanalyse fällt insbesondere die differierende Tokenisierung/Lemmatisierung der Präposition „*Im*“ ins Auge. Die Wortform

wird von TreeTagger, im Gegensatz zum Connexor-Output, nicht segmentiert. Damit ergeben sich bereits erste potenzielle Implikationen für vergleichende Korpusabfragen sowie statistisch relevante Korpusmaße (Wortlänge, Tokenzahl, Type-Token-Ratio usw.).

An dieser Stelle ist der Hinweis angebracht, dass in unserer Betrachtung keinesfalls die Beurteilung von Qualität oder Begründung maschinell erstellter Annotationen im Vordergrund stehen soll.[80] Vielmehr soll bewertungsneutral die Komplexität digitaler sprachlicher Strukturen aufgezeigt werden, mit denen sich die empirische Korpusanalyse typischerweise auseinanderzusetzen hat, um im Anschluss daran ein typologisches Anforderungsprofil für ein- oder mehrdimensionale Korpusabfragen aufzustellen.

Auch die dritte hier vorgestellte morphosyntaktische Analyse – diesmal unter Nutzung des *Xerox Incremental Parser* – segmentiert und lemmatisiert die bekannten Satzinhalte im Detail anders als die vorigen Werkzeuge. Darüber hinaus fügt sie eine Vielzahl ergänzender Informationen hinzu:

```
<LUNIT language="German">
<NODE num="6" tag="TOP" start="4592" end="4605" fts="CAT">
  <NODE num="9" tag="PP" start="4592" end="4604" fts="PP SPART
   NDATW DAT START FIRST">
    <NODE num="0" tag="PREP" start="4592" end="4594" fts="CAP
     XIP_CAP P_LOC SG3 NDATW MDATW WEAK SG NEUT MASC DAT START2
     PREP DET START FIRST">
      <TOKEN pos="PREP" start="4592" end="4594" surface="Im">
        <READING lemma="in" pos="PREP" fts="CAP XIP_CAP P_LOC SG3
         NDATW MDATW WEAK SG NEUT MASC DAT START2 PREP DET START
         FIRST"/>
        <READING lemma="in" pos="PREP" fts="CAP XIP_CAP P_LOC SG3
         NDATW WEAK SG NEUT DAT START2 PREP DET START"/>
      </TOKEN>
    </NODE>
    <NODE num="8" tag="NP" start="4595" end="4604" fts="NP SPART
     SG3 NDATS NDATW NACCS NACCW NNOMS NNOMW DAT ACC NOM NOUN
     LAST">
      <NODE num="7" tag="NPA" start="4595" end="4604" fts="NPA SG3
       NDATS NDATW NACCS NACCW NNOMS NNOMW DAT ACC NOM NOUN LAST
       FIRST">
        <NODE num="2" tag="NOUN" start="4595" end="4604" fts="CAP
         XIP_CAP COMMON SG3 NDATS NDATW NACCS NACCW NNOMS NNOMW P3
         WEAK STRONG SG NEUT DAT ACC NOM END2 NOUN NOAMBIGUITY
         LAST FIRST">
```

[80] Zur Brauchbarkeit der hier aufgeführten Annotationen für grammatische Untersuchungen vgl. z.B. Bubenhofer et al. (2014, S. 149 ff.).

```
          <TOKEN pos="NOUN" start="4595" end="4604"
           surface="Gegenteil">
            <READING lemma="Gegenteil" pos="NOUN" fts="CAP XIP_CAP
             COMMON SG3 NDATS NDATW NACCS NACCW NNOMS NNOMW P3
             WEAK STRONG SG NEUT DAT ACC NOM END2 NOUN NOAMBIGUITY
             LAST FIRST"/>
          </TOKEN>
        </NODE>
      </NODE>
    </NODE>
  </NODE>
  <NODE num="4" tag="PUNCT" start="4604" end="4605" fts="COLON
   SENT PUNCT NOAMBIGUITY END LAST">
    <TOKEN pos="PUNCT" start="4604" end="4605" surface=":">
      <READING lemma=":" pos="PUNCT" fts="COLON SENT PUNCT
       NOAMBIGUITY END LAST"/>
    </TOKEN>
  </NODE>
</NODE>
</LUNIT>
<LUNIT language="German">
<NODE num="22" tag="TOP" start="4606" end="4674" fts="CAT">
  <NODE num="30" tag="INS" start="4606" end="4634" fts="INS SPART
   START FIRST">
    <NODE num="28" tag="PP" start="4606" end="4620" fts="PP SPART
     NDATW DAT START FIRST">
      <NODE num="0" tag="PREP" start="4606" end="4609" fts="CAP
       XIP_CAP NUM_CONJ SG3 NDATW MDATW WEAK SG NEUT MASC DAT
       START2 PREP DET START FIRST">
        <TOKEN pos="PREP" start="4606" end="4609" surface="Zum">
          <READING lemma="zu" pos="PREP" fts="CAP XIP_CAP NUM_CONJ
           SG3 NDATW MDATW WEAK SG NEUT MASC DAT START2 PREP DET
           START FIRST"/>
          <READING lemma="zu" pos="PREP" fts="CAP XIP_CAP NUM_CONJ
           SG3 NDATW WEAK SG NEUT DAT START2 PREP DET START"/>
        </TOKEN>
      </NODE>
      <NODE num="26" tag="NP" start="4610" end="4620" fts="NP
       SPART SG3 NDATS NDATW NACCS NACCW NNOMS NNOMW DAT ACC NOM
       NOUN LAST">
        <NODE num="23" tag="NPA" start="4610" end="4620" fts="NPA
         SG3 NDATS NDATW NACCS NACCW NNOMS NNOMW DAT ACC NOM NOUN
         LAST FIRST">
          <NODE num="2" tag="NOUN" start="4610" end="4620"
           fts="COMPD_LEVEL CAP XIP_CAP COMMON SG3 NDATS NDATW
```

```
          NACCS NACCW NNOMS NNOMW P3 WEAK STRONG SG NEUT DAT ACC
          NOM NOUN NOAMBIGUITY LAST FIRST">
           <TOKEN pos="NOUN" start="4610" end="4620"
            surface="Kultobjekt">
             <READING lemma="Kult#Objekt" pos="NOUN" fts=
              "COMPD_LEVEL CAP XIP_CAP COMMON SG3 NDATS NDATW
              NACCS NACCW NNOMS NNOMW P3 WEAK STRONG SG NEUT DAT
              ACC NOM NOUN NOAMBIGUITY LAST FIRST"/>
           </TOKEN>
         </NODE>
       </NODE>
     </NODE>
   </NODE>
   <NODE num="4" tag="VERB" start="4621" end="4633" fts="SPART
    PPAST VERB NOAMBIGUITY">
     <TOKEN pos="VERB" start="4621" end="4633" surface=
      "aufgestiegen">
       <READING lemma="auf=steigen" pos="VERB" fts="SPART PPAST
        VERB NOAMBIGUITY"/>
     </TOKEN>
   </NODE>
   <NODE num="6" tag="PUNCT" start="4633" end="4634" fts=
    "COORD_SENT COMMA PUNCT NOAMBIGUITY LAST">
     <TOKEN pos="PUNCT" start="4633" end="4634" surface=",">
       <READING lemma="," pos="PUNCT" fts="COORD_SENT COMMA PUNCT
        NOAMBIGUITY LAST"/>
     </TOKEN>
   </NODE>
 </NODE>
 <NODE num="29" tag="MC" start="4635" end="4673" fts="MC">
   <NODE num="8" tag="VERB" start="4635" end="4641" fts="REQ_PREF
    V_ONLY PL2 SG3 PRES V1 INDIC FINITE P3 P2 PL SG VERB FIRST">
     <TOKEN pos="VERB" start="4635" end="4641" surface="feiert">
       <READING lemma="feiern" pos="VERB" fts="REQ_PREF V_ONLY
        PL2 SG3 PRES V1 INDIC FINITE P3 P2 PL SG VERB FIRST"/>
       <READING lemma="feiern" pos="VERB" fts="SG3 PRES INDIC
        FINITE P3 SG VERB"/>
       <READING lemma="feiern" pos="VERB" fts="REQ_PREF PL2 PRES
        V1 INDIC FINITE P2 PL VERB"/>
       <READING lemma="feiern" pos="VERB" fts="REQ_PREF SG3 PRES
        V1 INDIC FINITE P3 SG VERB"/>
     </TOKEN>
   </NODE>
   <NODE num="27" tag="NP" start="4642" end="4644" fts="NP SPART
```

```
 SG3 ACC NOM NOUN">
  <NODE num="10" tag="PRON" start="4642" end="4644" fts=
   "IMPERSO PERS SG3 P3 SG NEUT ACC NOM PRON NOAMBIGUITY LAST
   FIRST">
    <TOKEN pos="PRON" start="4642" end="4644" surface="es">
      <READING lemma="es" pos="PRON" fts="IMPERSO PERS SG3 P3
       SG NEUT ACC NOM PRON NOAMBIGUITY LAST FIRST"/>
    </TOKEN>
  </NODE>
</NODE>
<NODE num="12" tag="ADV" start="4645" end="4650" fts="TEMPORAL
 SADV SPART ADV NOAMBIGUITY">
  <TOKEN pos="ADV" start="4645" end="4650" surface="heute">
    <READING lemma="heute" pos="ADV" fts="TEMPORAL SADV SPART
     ADV NOAMBIGUITY"/>
  </TOKEN>
</NODE>
<NODE num="14" tag="ADV" start="4651" end="4656" fts="MOD_ADJ
 SPART ADV NOAMBIGUITY">
  <TOKEN pos="ADV" start="4651" end="4656" surface="sogar">
    <READING lemma="sogar" pos="ADV" fts="MOD_ADJ SPART ADV
     NOAMBIGUITY"/>
  </TOKEN>
</NODE>
<NODE num="25" tag="NP" start="4657" end="4673" fts="NP SPART
 SG3 FACCW FNOMW ACC NOM NOUN LAST">
  <NODE num="16" tag="DET" start="4657" end="4661" fts="INDEF
   SG3 FACCW FNOMW WEAK SG FEM ACC NOM DET NOAMBIGUITY FIRST">
    <TOKEN pos="DET" start="4657" end="4661" surface="eine">
      <READING lemma="ein" pos="DET" fts="INDEF SG3 FACCW
       FNOMW WEAK SG FEM ACC NOM DET NOAMBIGUITY FIRST"/>
    </TOKEN>
  </NODE>
  <NODE num="24" tag="NPA" start="4662" end="4673" fts="NPA
   SG3 FGENS FGENW FDATS FDATW FACCS FACCW FNOMS FNOMW GEN DAT
   ACC NOM NOUN LAST">
    <NODE num="18" tag="NOUN" start="4662" end="4673" fts="CAP
     XIP_CAP COMMON SG3 FACCS FACCW FNOMS FNOMW P3 WEAK STRONG
     SG FEM GEN DAT ACC NOM END2 NOUN NOAMBIGUITY LAST FIRST">
      <TOKEN pos="NOUN" start="4662" end="4673" surface=
       "Renaissance">
        <READING lemma="Renaissance" pos="NOUN" fts="CAP
         XIP_CAP COMMON SG3 FACCS FACCW FNOMS FNOMW P3 WEAK
         STRONG SG FEM GEN DAT ACC NOM END2 NOUN NOAMBIGUITY
         LAST FIRST"/>
```

```
          </TOKEN>
        </NODE>
      </NODE>
    </NODE>
  </NODE>
  <NODE num="20" tag="PUNCT" start="4673" end="4674" fts="SENT
   PUNCT NOAMBIGUITY END LAST">
    <TOKEN pos="PUNCT" start="4673" end="4674" surface=".">
      <READING lemma="." pos="PUNCT" fts="SENT PUNCT NOAMBIGUITY
       END LAST"/>
    </TOKEN>
  </NODE>
</NODE>
</LUNIT>
```

Hinzugekommen sind auf der Tokenebene Bestimmungen diverser Flexionsmerkmale wie Genus, Kasus oder Numerus. Diese erscheinen als Attributwerte in sogenannten Featurelisten (Attributtyp *fts*). Featurelisten sind an Lesarten (Elementtyp *READING*) gebunden, von denen es potenziell mehrere pro Token geben darf. Komposita werden morphologisch zerlegt und lemmatisiert (*Kult#Objekt*), komplexe Verben in ihre Bestandteile segmentiert (*auf=steigen*) usw. Auch wortübergreifend versucht der Parser, hierarchische Strukturen und Zusammenhänge aufzudecken: Präposition und Nomen in „*Im Gegenteil*“ werden als Präpositionalphrase klassifiziert, Artikel und Nomen in „*eine Renaissance*“ als Knoten einer Nominalphrase zusammengefasst – wiederum partiell abweichend vom Connexor-Output. Auswirkungen auf kontextsensitive Korpusrecherchen hat die divergierende Segmentierung auf Satzebene. Die Xerox-Analyse teilt – anders als der Connexor- oder TreeTagger-Output – den Beleg in zwei eigenständige Sätze (vgl. Tab. 5), was beispielsweise die spätere Suche nach satzbezogenen syntaktischen Phänomenen oder die Berechnung von Nachbarschaftskookkurrenzen beeinflusst.

Token	**Grundform**	**Wortklasse**	**Wortgruppen**	**Satz**
Im	In	PREP	PP	Satz 1
Gegenteil	Gegenteil	NOUN	PP NP NPA	Satz 1
:	:	PUNCT		Satz 1
Zum	Zu	PREP	INS PP	Satz 2
Kultobjekt	Kult#Objekt	NOUN	INS PP NP NPA	Satz 2
aufgestiegen	auf=steigen	VERB	INS	Satz 2

Token	Grundform	Wortklasse	Wortgruppen	Satz
,	,	PUNCT	INS	Satz 2
feiert	Feiern	VERB	MC	Satz 2
es	Es	PRON	MC NP	Satz 2
heute	Heute	ADV	MC	Satz 2
sogar	Sogar	ADV	MC	Satz 2
eine	Ein	DET	MC NP	Satz 2
Renaissance	Renaissance	NOUN	MC NP NPA	Satz 2
.	.	PUNCT		Satz 2

Tab. 5: Zusammenfassung der Xerox-Annotation

Ergänzend notiert der Xerox-Parser syntaktische Dependenzen (Subjekt, Objekt etc.) zwischen einzelnen Wortknoten, z.B.:

```
<DEPENDENCY name="DETERM" fts="INDEF">
     <PARAMETER ind="0" num="18" word="Renaissance"/>
     <PARAMETER ind="1" num="16" word="eine"/>
</DEPENDENCY>
<DEPENDENCY name="OBJ" fts="ACC">
     <PARAMETER ind="0" num="8" word="feiert"/>
     <PARAMETER ind="1" num="18" word="Renaissance"/>
</DEPENDENCY>
<DEPENDENCY name="SUBJ" fts="IMPERSO">
    <PARAMETER ind="0" num="8" word="feiert"/>
    <PARAMETER ind="1" num="10" word="es"/>
</DEPENDENCY>
<DEPENDENCY name="VMOD">
    <PARAMETER ind="0" num="8" word="feiert"/>
    <PARAMETER ind="1" num="12" word="heute"/>
</DEPENDENCY>
```

In Anlehnung an die in Abschnitt 2.3 spezifizierten Abfrageszenarien lassen sich für unseren mehrfach annotierten Beispielsatz folgende Abfragetypen, die hier jeweils anhand konkreter Manifestationen veranschaulicht werden, unterscheiden:

1) Suche nach einzelnen diskreten Elementen einer Segmentierungsebene:
   - Token feiert
   - Lemma feiern
   - Wortklasse „PRON“
   - Wortgruppe „NP“
   - Relative Position im Satz (z.B. satzeinleitendes Wort)
2) Suche nach einzelnen Elementen einer Segmentierungsebene unter Verwendung variabler Musterausdrücke (z.B. zur Spezifizierung von Wortbestandteilen wie Präfix, Suffix usw.):
   - mit Platzhalterzeichen: Token *auf** (findet *aufsteigen*)
   - mit regulärem Ausdruck: Lemma *[Z|z]u[m]?* (findet *zu, zum, Zu*)
3) Suche nach mehreren Elementen potenziell unterschiedlichen Typs bzw. unterschiedlicher Segmentierungsebene unter Nutzung logischer Operatoren (Existenzprüfung):
   - Token *Kultobjekt* UND Lemma *feiern*
   - Token *Gegenteil* UND NICHT Token *Kultobjekt*
   - Wortklasse „NOUN“ ODER Wortgruppe „NP“
4) Suche nach mehreren Elementen potenziell unterschiedlichen Typs bzw. unterschiedlicher Segmentierungsebene unter Nutzung von Abstandsoperatoren und hierarchischen Bedingungen:
   - Lemma *in* am Satzanfang, gefolgt von einem Doppelpunkt mit einem maximalem Abstand von zwei Textwörtern
   - Token *Gegenteil* in Wortgruppe „PP“
5) Kombinierte Suche in parallelen Annotationen:
   - Connexor-Lemma *in* in Xerox-Wortgruppe „NP“
6) Einbeziehung kategorialer Dependenzen:
   - Ausgabe der mit dem Lemma *feiern* verbundenen Akkusativobjekte
7) Einbeziehung von Metadaten:
   - Eingrenzung der Suche auf in Süddeutschland erschienene politische Publikumspresse der 1990er Jahre
8) Einbeziehung von Korpusstatistiken:
   - Eingrenzung der Suche auf Verben einer bestimmten Frequenzklasse

Derartige multidimensionale Recherchen in Mehrebenen-Annotationen sowie strukturell übergeordneten Metadaten erfordern elaborierte Speicherungskonzepte und Abfragewerkzeuge. Zu letzteren zählen modular erweiterbare Abfrageoberflächen sowie expressive Abfragesprachen. Diese kommen an der Schnittstelle zwischen Korpusnutzer und Datenrepository zum Einsatz und sind ursächlich von der Komplexität der abzufragenden Annotationslayer abhängig. Bei der Spezifikation der konzeptionellen Anforderungen an Speicherungstechniken steht obendrein, d.h. zusätzlich zur logischen Abbildung sämtlicher abfragerelevanter Datenstrukturen, der Umgang mit extrem großen Datenvolumina im Vordergrund. Hier gilt es, performante Retrievaloperationen auch für stark ansteigende Mengen von Primär- und Sekundärdaten unterschiedlicher Tagsets sicherzustellen. Damit einher geht eine Vervielfachung der potenziell zu analysierenden Relationen zwischen hierarchisch divergierenden Datenebenen.

Vor diesem Hintergrund lassen sich für unsere oben herausgestellten Abfragetypen folgende prototypischen Problematiken hinsichtlich der Erschließung multipler und ggf. konkurrierender Hierarchien unterscheiden:

1) **Divergierende Trennlinien text- oder korpusspezifischer Metadaten:** Im klassischen Information Retrieval dienen deskriptive Metadaten der Einschränkung der für eine Recherche relevanten Datenmenge. Sie definieren die zu berücksichtigenden Primär- bzw. Indexdaten. Idealerweise geht die logische Einschränkung mit einer Reduzierung des zu durchsuchenden physischen Datenvolumens und damit der benötigten Rechenleistung einher. Zwei Verfahrensweisen erscheinen grundsätzlich praktikabel:
   – Segmentierung der Daten: Dieses Verfahren gestaltet sich umso einfacher, je weniger Metadatenkategorien einbezogen werden müssen. Bei steigender Kategorienanzahl wandelt sich der positive Effekt hingegen rasch in einen informatischen Nachteil, weil eine adäquate Segmentierung aufgrund des organisatorischen Overheads nicht mehr praktikabel ist. Das wird am Beispiel unseres obigen Belegs deutlich: Soll ein Korpusbestand – ergänzend zur Recherche nach hierarchisch untergeordneten Strukturen wie Wortverbindungen oder Einzelwörtern – ausschließlich bezüglich des Metadatums „Medium“ eingeschränkt werden, erfolgt naheliegenderweise eine Segmentierung in so viele „Datentöpfe“, wie es Merkmalsausprägungen gibt. Bei fünf Ausprägungen (z.B. „Publikumspresse“, „Belletristik“, „Internet“, „Gesprochenes“ und „Sonstiges“) lassen sich also fünf nicht-überschneidende Datentöpfe einrichten und die weitere Recherche auf die der Suchspezifikation entsprechenden Inhalte beschränken. Eine zusätzliche Merkmals-

kategorie „Region“ (z.B. mit den sieben Ausprägungen „Nordwest“, „Nordost“, „Mittelwest“, „Mittelost“, „Mittelsüd“, „Südwest“, „Südost“) impliziert bereits 5 × 7 = 35 nicht-überschneidende Datentöpfe. Eine ergänzende thematische Diversifizierung (z.B. in die acht Domänen „Politik“, „Wirtschaft“, „Natur“, „Technik“, „Sport“, „Kultur“, „Fiktion“ und „Diverses“) resultiert in 5 × 7 × 8 = 280 Datentöpfen. Soll nun der Korpusbestand weiterhin unter zeitlichem Aspekt (nach Jahr, Dekade o.Ä.) segmentiert werden, resultiert der Versuch einer Übertragung der logischen Unterteilung in physikalische Entsprechungen rasch in einer kaum noch handhabbaren Datenstruktur.

– Explizite Hinzufügung von Metadaten: Im Gegensatz zum vorigen Verfahren werden die zu durchsuchenden Primär- und Annotationsdaten nicht nach Metadaten aufgeteilt, sondern um Angaben zur jeweiligen Merkmalsausprägung angereichert. Bei maximaler Anwendung erhält jedes recherchierbare Element (z.B. Wort) eine Liste der relevanten Metadatenkategorien, z.B. „Medium = Publikumspresse“, „Region = Südwest“, „Thema = Politik“, „Jahr = 1995“. Die logische Einschränkung während der Recherche erfolgt dann unter Nutzung entsprechend implementierter Indizes. Auch hier stellt die Anzahl der einzubeziehenden Metadatenkategorien das informatische Nadelöhr dar: Kombinierte Indizes sind nicht nur speicherplatzintensiv, sondern weisen mit zunehmender Schlüssellänge eine tendenziell abnehmende Performanz auf. Sollen auf Metadaten basierende Bereichsabfragen darüber hinaus auch auf Einheiten unterhalb der Wortebene (Morphem, Silbe, Laut) durchgeführt werden, verschlechtert sich das Kosten-Nutzen-Verhältnis des Verfahrens weiter.

2) **Überschneidung von Annotationslayern:** Die Abfrage von Objekten aus verschiedenen Beschreibungsebenen/Annotationslayern hat mit dem zentralen Umstand umzugehen, dass nicht nur – wie etwa beim Text-Retrieval – eine lineare Zeichenkette nach mehr oder weniger komplexen Suchausdrücken durchsucht werden soll, sondern vertikal geschichtete Ebenen zueinander in Bezug gesetzt werden müssen. Typische Phänomene sind die Inklusion (z.B. die Suche nach einer bestimmten Wortform innerhalb eines spezifischen Phrasentyps) sowie Sonderfälle, bei denen Elemente unterschiedlicher Ebenen gemeinsame Start- bzw. Endpunkte aufweisen (z.B. wenn die Wortform explizit am Anfang der Phrase stehen soll). Weiterhin gehört hierzu die Überprüfung von Identität, also ob eine Nominalphrase etwa ausschließlich aus einem einzelnen Substantiv besteht oder ob die Lemmatisierungen mehrerer Tagger übereinstimmen. Und schließlich gilt es Phänomene der „klassischen“ Überlappung abzudecken, wenn beispiels-

weise Satzgrenzen von verschiedenen Werkzeugen uneinheitlich annotiert wurden. Wiederum stehen zwei Verfahrensweisen zur Auswahl:

- Explizite Kodierung der Bezüge zwischen den Ebenen: In der Datenbasis werden Phänomene wie Inklusion, Identität oder Überlappung, soweit rechercherelevant, exhaustiv zusammengestellt und über eindeutige Schlüsselwerte festgehalten („Wortgruppe XY enthält Wörter a, b, c", „Wort a ist Bestandteil von Satz S1 und Wortgruppe XY" etc.). Dieses Verfahren resultiert in effektiv abfragbaren Datenstrukturen, setzt jedoch eine umfassende Vorabanalyse der für spätere Recherchen benötigten Ebenenbezüge voraus und ist dadurch nur begrenzt flexibel.
- Implizite Bereitstellung der Bezüge über Positionswerte: Eine hinsichtlich der Nutzung für variable Abfrageszenarien mächtigere Lösung gründet sich auf der Kodierung absoluter Positionsangaben der Ebenenkonstrukte. Für sämtliche Segmente der unterschiedlichen Annotationslayer werden Start- und Endposition im Primärtext indiziert, letztere ggf. unter Auswertung von Längenangaben der Annotationswerkzeuge. Die Bezüge lassen sich anschließend in einem iterativen Verfahren aufdecken, also beispielsweise durch eine initiale Ermittlung der Segmentgrenzen für eine bestimmte Wortgruppe und einer darauf aufbauenden Ausfilterung derjenigen Wortformen, die innerhalb dieser Positionsgrenzen liegen.

Die hier aufgezeigten Problematiken bei der Kombination multdimensionaler Suchkriterien exemplifizieren zentrale Herausforderungen moderner und zukünftiger Korpusrecherchesysteme. Nicht das vergleichsweise unproblematische „horizontale" Retrieval von Primärtext-Zeichenketten steht dabei im Vordergrund, sondern die gleichermaßen verlässliche wie performante Integration von Primär-, Annotations- und Metadaten. Eine in Betracht kommende Verfahrensweise soll in den nachfolgenden Kapiteln systematisch implementiert und evaluiert werden.

2.5 Anforderungskatalog für linguistisch motivierte Korpusabfragen

Die bislang beleuchteten Abfragemöglichkeiten etablierter Korpusrecherchesysteme sowie die Herausforderungen bei der Einbeziehung zusätzlicher, multidimensionaler Suchkriterien wecken den Bedarf nach einem umfassenden Anforderungskatalog, der für die Evaluation informatischer Lösungen Anwendung finden könnte. Dieser Wunsch gewinnt dadurch an Bedeutung, dass prototypische Beispielabfragen in der bekannten einschlägigen Litera-

tur bislang zumeist dazu verwendet werden, die individuellen Charakteristika einzelner Systeme zu demonstrieren oder zur Erforschung spezifischer linguistischer Fragestellungen beizutragen. Ein übergreifender „Gold Standard" für abgestuft komplexe Korpusabfragen erscheint deshalb als naheliegendes Desiderat, um potenzielle Retrievalansätze (Speicher- und Indexstrukturen, Methoden, Algorithmen) hinsichtlich ihrer Leistungsfähigkeit tendenziell vergleichbar zu machen.

In diesem Sinne formulieren wir nachfolgend zehn konkrete Abfrageszenarien, die ein breit gefächertes Spektrum korpuslinguistischer Kategorien und Merkmale abdecken. Hierfür berücksichtigen wir das im vorigen Abschnitt dargestellte Inventar an Sekundärdaten, also verschiedenartige Annotationslayer ebenso wie sprachexterne Metadaten. Die Integration von Metadaten flexibilisiert dabei die in manchen existierenden Systemen angebotene Strategie, virtuelle Untersuchungskorpora vorab zu definieren und dadurch die zur Suchzeit abzufragende Datenmenge zu minimieren. Proportionierungen einzelner Strata in opportunistischen Korpora lassen sich durch spontane Metadaten-Einschränkungen deutlich variabler durchführen.

Unsere Beispielabfragen fokussieren technische Gegebenheiten und Herangehensweisen beim Korpusretrieval in unterschiedlichem Maße. Während sich beispielsweise explizit realisierte alphanumerische Inhalte (also Token, Lemmata und Wortklassenbezeichner, aber auch außersprachliche Metadaten wie Medium oder Jahr) vergleichsweise effizient indizieren lassen, rückt dieser Aspekt bei der Verwendung bestimmter Platzhalterzeichen (z.B. zur Links-Trunkierung) und regulärer Ausdrücke tendenziell in den Hintergrund. Hier helfen Indizes aufgrund der zumeist äußerst umfangreichen Menge der auf solche Suchmuster passenden Wortformen nur begrenzt weiter. Folglich gilt es bei einer Evaluierung zusätzlich, passende algorithmisierbare Verfahren für das Pattern Matching zu erproben.[81]

Unser Anforderungskatalog gliedert sich aus informationstechnologischer Perspektive in vier Sektionen:

[81] Dies erfolgt in Kapitel 3; auf die Suchmuster unserer Referenzabfragen geht Kapitel 4 genauer ein.

1) Suchmuster aus diskreten Elementen mit oder ohne Platzhalterzeichen

	Suchmuster	**Beispiel**
Abfrage 1	Token *dabei*	Die verbreitete Zuversicht dürfte dabei den Obstproduzenten gut getan haben.
Abfrage 2	Lemma **bezogen*	Geprobt wird projektbezogen, also immer auf ein Konzert oder einen Auftritt hin.

2) Suchmuster als lineare Verkettung mehrerer Einzelelemente über deren relative Position

	Suchmuster	**Beispiel**
Abfrage 3	Relativsatz mit *was*: Lemma *das* am Satzanfang mit Wortabstand 1 oder 2 vor Nomen (Connexor-Wortklasse „N“) unmittelbar vor einem Komma unmittelbar vor Token *was*	Das einzige Argument, was ich immer wieder höre, ist kein Argument, sondern ein Werturteil.
Abfrage 4	ACI-Konstruktionen: Infinitiv (Connexor-Subkategorie „INF“) unmittelbar vor einem Wahrnehmungsverb, dessen Lemma entweder *hören*, *sehen*, *spüren*, *fühlen* oder *riechen* lautet, unmittelbar vor einem Satzende-Punkt. Mit beliebigem Abstand vor der Verbkombination ein Pluralnomen (Connexor-Subkategorie „PL“) sowie das Lemma *haben* ohne Trennwort.	Hausbewohner hatten auch am Montagnachmittag die Frau und einen Mann in der Wohnung streiten gehört.
Abfrage 5	W-Fragen ohne Verb: adverbiales Interrogativpronomen am Satzanfang (TreeTagger Wortklasse „PWAV“) ohne nachfolgendes Verb (= nicht TreeTagger-Wortklasse „VRB“) vor Fragezeichen	Warum kein geeintes Deutschland?

3) Suchmuster aus linearen Abfolgen sowie hierarchischen Annotationsmerkmalen unterschiedlicher Parser, Metadaten, Dependenzen oder Korpusstatistiken

	Suchmuster	Beispiel
Abfrage 6	Movierte Anreden in virtuellen Subkorpora: Token *Frau* unmittelbar vor einem Nomen unter Ausschluss von Eigennamen (Connexor-Wortklasse „N“, aber nicht „N Prop“) mit der Endung *in* aus Texten des Themenbereichs „Politik/Wirtschaft/Gesellschaft“ (Domäne) seit 2000 (Jahr)	„Frau Kapitänin, meine Herren“, grüßt Grobien jedes Mal sein Publikum, wenn er als Meister der Zeremonie etwas ansagen muss.
Abfrage 7	Genitivobjekte (Xerox-Dependenz „OBJ GEN“) zu Lemma *erfreuen*	Sie erfreuen sich ihrer Beliebtheit zu Recht.
Abfrage 8	Partizipialphrasen: Belege mit einem aus dem Verb *sehen* gebildeten Adjektiv (Partizip I) oder einer als Adjektiv gebrauchten Verbform (Partizip II) innerhalb einer Adjektivphrase (Xerox-Knoten „AP“), unmittelbar vor einem niederfrequenten Nomen (Frequenzklasse > 9)	Auch hier gibt es schöne, bisher nicht gesehene Töne.

4) Suchmuster mit regulären Ausdrücken

	Suchmuster	**Beispiel**
Abfrage 9	Straßennamen als Aneinanderreihung mit mindestens zwei Durchkopplungsbindestrichen und beginnend mit Wil: Token *Will*.+\-.+\-(Straße|Weg|Platz|Allee)$	Mehr als 100 Beamte riegelten deshalb am Vormittag das Gelände an der Willy-Brandt-Straße ab.
Abfrage 10	Internet-Domänen in Deutschland mit oder ohne Protokollangabe: Token (http:\/\/)?www\..+?\.de$	Informationen finden sich im Internet unter: www.uni-leipzig.de.

Die vom Benutzer intendierten Ergebnisse von Korpusabfragen beinhalten häufig nicht nur explizite Belege und eine Textsigle für die Zitation, sondern auch für die Interpretation bzw. Weiterverarbeitung notwendige Zusatzinformationen. Hierzu zählen in erster Linie textspezifische Metadaten, aber auch statistische Angaben zur Verteilung der Fundstellen. Beispielsweise gilt es bei der Auswertung von Recherchen zu berücksichtigen, dass ein einmal in einen Diskurs neu eingeführter Ausdruck oft nachfolgend vermehrt aufgegriffen wird und dadurch in einzelnen Texten einen besonders prominenten Status erhält. Gleiches gilt für autorenspezifische Redewendungen, Wortschöpfungen usw. Um diesen Effekt der „Clumpiness" oder „Burstiness"[82] aufzudecken, können statistische Maße zur Beurteilung der Gleichmäßigkeit der Verteilung herangezogen werden. Diese setzen voraus, dass Trefferlisten nach vordefinierten Einheiten (z.B. Texte oder noch kleinere Abschnitte) aufgeteilt und Zählungen von Vorkommen, Gesamtwortanzahlen etc. mitgeliefert werden. Ebenso ist für viele Fragestellungen die Ausgabe von Verteilungen über die verschiedenen Korpusstrata wünschenswert, um aussagekräftige Analysen zu ermöglichen.

Diese Vielfalt der potenziell notwendigen Zusatzinformationen macht eine wie auch immer geartete generelle Standardisierung von Rückgabeformaten schwierig, da für wechselnde Fragestellungen und empirische Phänomene jeweils angemessene Verfahren mit spezifischen Datenspezifika zum Einsatz kommen. Der Anforderungskatalog impliziert deshalb über die Ausgabe von Beispielbelegen bzw. Referenzen ihres Vorkommens in der Datenbasis (Korpus-, Text- oder Satznummern) hinaus keinerlei Vorgaben hinsichtlich des Formats oder des Umfangs von Antwortdatensätzen.

[82] Vgl. z.B. Bubenhofer et al. (2014, S. 135); Church/Gale (1995); Gries (2008a); Kilgarriff (2001).

3. Design und Implementierung eines Korpusabfragesystems

Für die Exploration und Verarbeitung linguistisch relevanter Primär- und Sekundärdaten stellen Korpusabfragesysteme multifunktionale Benutzeroberflächen (*front ends*) bereit. Wie im vorigen Kapitel aufgezeigt wurde, können diese Oberflächen interaktiv-grafischer Natur oder formularbasiert sein sowie optional die Verwendung einer syntaktisch spezialisierten Korpusabfragesprache erlauben. Unabhängig von solchen konkreten Designentscheidungen hängt ihre Leistungsfähigkeit in herausragendem Ausmaße davon ab, wie die abzufragenden strukturierten bzw. semi-strukturierten Inhalte konkret auf der dahinter liegenden Speicherebene (*back end*) organisiert sind. Komplexe Abfragen mit verschiedenartigen Suchparametern wie in unserem Anforderungskatalog oder statistische Analysen und Visualisierungen empirischer Phänomene lassen sich bei ungünstiger Modellierung und Implementierung der Datenstrukturen nur mit erheblichen Performanzeinschränkungen – oder im misslichsten Fall überhaupt nicht – durchführen.

Typologisch werden mehrfach annotierte Korpussysteme bisweilen im Umfeld sogenannter „Big Data"-Technologien (Geiselberger/Moorstedt 2013; Mohanty et al. 2015; Rahm et al. 2015) verortet (vgl. hierzu z.B. Bubenhofer/Scharloth 2015). Dieser Terminus subsumiert Strukturen und Methoden für die Verarbeitung umfangreicher Datenvolumina, welche klassische Ansätze der Datenhaltung und des Information-Retrieval um Features zur Optimierung kritischer Antwortzeiten erweitern. Zwar liegt der praktische Fokus häufig auf Profilinhalten aus sozialen Netzwerken oder Nutzungsdaten aus Telekommunikationsprotokollen, grundsätzlich können aber auch umfangreiche Textrepositorien, die als „Rohstoff" (Heyer et al. 2008) für gezielte sprachwissenschaftliche Auswertungen einer besonderen maschinellen Verarbeitung bedürfen, von einschlägigen Strategien profitieren.

Charakteristisch für Big Data ist das in Laney (2001) beschriebene und z.B. in Klein et al. (2013) näher beleuchtete 3-V-Modell. Darin werden die informatischen Herausforderungen sehr großer Informationsmengen unter Verwendung dreier Dimensionen herausgestellt. Diese Dimensionen lassen sich mit gewissen Einschränkungen auf Korpussysteme anwenden:

– Erste Dimension: Volumen (*volume*): Der Wunsch nach immer umfangreicheren authentischen Textbelegen für die Erforschung natürlicher Sprache führt, erleichtert durch mittlerweile durchgängig elektronische Publikations-

verfahren und die damit einher gehende Verfügbarkeit von Quellen sowie durch die Verschiebung früherer physikalischer Speicherbeschränkungen, zu einem enormen Wachstum aktueller Korpussammlungen.[83]

– Zweite Dimension: Vielzahl (*variety*): Korpussysteme verwalten nicht nur Rohtexte, sondern darüber hinaus vielfältige Meta- und Annotationsdaten. Deren hochkomplexe Inhalte, Strukturen und wechselseitige Beziehungen gilt es abzubilden und zuverlässig recherchierbar zu machen.
– Dritte Dimension: Verarbeitungsgeschwindigkeit (*velocity*): Geschwindigkeitsaspekte beziehen sich einerseits auf die Wachstumsrate aktueller Korpora, in weitaus stärkerem Maße jedoch auf das Bedürfnis nach optimierten Recherchezeiten. Auch wenn aufwändigen Korpusanfragen ein flexibleres Zeitfenster als ad hoc-Recherchen bei einer Internet-Suchmaschine zugestanden werden, bleiben stundenlange Abarbeitungszyklen in der Regel unakzeptabel.

Aus diesen Charakteristika und Aspekten lassen sich erste technische Folgerungen für das Design von Korpusverwaltungssystemen ableiten. Um große Mengen an Primär- und Sekundärdaten hinsichtlich der in wechselnden Projektkontexten anstehenden linguistischen Fragestellungen angemessen und reproduzierbar auswerten zu können, bedarf es eines gleichermaßen verlässlichen, performanten und funktional erweiterbaren Ansatzes. Zentrales Anforderungskriterium ist die feingranulare Abbildung verschachtelter Mehrebenen-Strukturen bei gleichzeitiger Unterstützung multidimensionaler Recherchen. Die verwendete Speicherarchitektur sollte außerdem aus Gründen der Nachhaltigkeit kompatibel mit Standards sein, die gegenwärtig im Kontext nationaler und supranationaler Infrastrukturverbundprojekte ausgearbeitet werden.[84] Im informationstechnologischen Forschungsumfeld finden sich hierzu mehrere einschlägige, teilweise überlappende oder kombinierbare Modelle, die nachfolgend aufgeführt und hinsichtlich ihrer grundsätzlichen Eignung bewertet werden.

---

[83] Dessen ungeachtet liegen die absoluten Datengrößen sprachwissenschaftlicher Korpora noch weit unter denen, die bei der umfassenden Analyse von Internetnutzungsprotokollen, Mobilfunksignalen oder Konsumstatistiken anfallen.

[84] Hierzu zählen etwa die bereits erwähnten Projekte TextGrid, CLARIN oder DARIAH, die unter anderem Vorschläge für die Strukturierung von Metadaten für Sprachressourcen anbieten. Einen prototypischen Ansatz, dessen Schwerpunkt allerdings nicht in der Evaluation potenzieller Speicherarchitekturen liegt, präsentiert z.B. Cunningham/Bontcheva (2006) mit SALE (Software Architecture for Language Engineering), das für Design, Implementation und Evaluation von GATE (General Architecture for Text Engineering) entwickelt wurde; vgl. https://gate.ac.uk.

3.1 Spektrum der Speicherungsmodelle

Die Auswahl eines konkreten Speicherungsmodells soll die gleichermaßen bedarfsgerechte wie effiziente Implementierung der Datenbasis auf Grundlage einer vorgelagerten semantischen Modellierung sicherstellen. Aspekte der Effizienz beziehen sich dabei auf die intendierten Operationen, also üblicherweise auf das Neuanlegen, Ändern, Abfragen oder Löschen von Datensätzen. Zwischen Speicherungs- und Zugriffsmodellen existieren deshalb naturbedingt enge Wechselbeziehungen. Korpusabfragesysteme als Sonderform analytischer Informationssysteme legen dabei, ähnlich wie Data Warehouse- oder OLAP (Online Analytical Processing)-Systeme, den Schwerpunkt auf die multidimensionale Abfrage heterogener Daten. Umfangreiche Modifikationen des Datenbestands fallen dagegen vergleichsweise selten an bzw. können zumeist durch in betriebsarme Zeitfenster ausgelagerte Transaktionen vollzogen werden, so dass die Optimierung der Ad hoc-Retrievaloperationen im Vordergrund steht.

Eine strikte Unterscheidung und Ordnung der für Textkorpora zur Verfügung stehenden Speicherungsoptionen ist nicht trivial, weil disparate Charakteristika in solche Bewertungen einfließen: Interne Repräsentationsformen (XML, Listen, Tabellen, Indizes) spielen ebenso eine Rolle wie systemspezifische Manipulationswerkzeuge (SQL oder die diversen Standards im XML-Umfeld). Auch hardwaretechnische Aspekte wie die Wahl des Speichermediums (magnetische Festplatte, Solid-State-Disk, Hauptspeicher) gilt es zu berücksichtigen. Weiterhin existieren vorkonfektionierte oder anwendungsspezifisch zusammengestellte Hybride, etwa datenbankbasierte In-Memory-Systeme. Nichtsdestotrotz erscheint eine generalisierende Bestandsaufnahme zweckmäßig, um vor dem Hintergrund existierender Lösungen zu einer begründeten Wahl der technischen Basis zu gelangen. Unsere Übersicht orientiert sich primär an den historisch für die Konstruktion von Korpusabfragesystemen zum Einsatz gekommenen Technologien und abstrahiert von einzelnen Produkten auf übergreifende Kategorien.

3.1.1 Dateisystembasierte Lösungen

Aufbereitete Rohtexte sowie der Output linguistischer Tagger liegen gemeinhin in Form strukturierter XML-Dateien vor. Ein naheliegender Ansatz ist deshalb, diese Daten unmittelbar mit XML-Werkzeugen und unter Ausnutzung etablierter Standards wie XPath (*XML Path Language* zur Adressierung von Teilen einer XML-Instanz), XSLT (*Extensible Stylesheet Language Transformations* zur Konvertierung zwischen unterschiedlichen XML-Modellen) und XQuery (*XML Query Language* zur Recherche in XML-Instanzen) zu analysie-

ren.[85] Sämtliche Inhalte befinden sich dabei innerhalb des Dateisystems und lassen sich durch das Hinzufügen von Massenspeichergeräten variabel erweitern. Bei der Wahl einer konkreten Softwarelösung rücken neuere Entwicklungen wie verteilte oder parallele Dateisysteme in den Fokus, die Aspekte wie Datenverteilung (Ablage von Dateien auf einem einzigen Speicherknoten vs. Aufteilung auf multiple Knoten), Änderungsprotokollierung, Fehlertoleranz beim Ausfall einzelner Hardwarekomponenten oder Unterstützung paralleler I/O-Zugriffe unterstützen. Anwendungslogik und Frontends lassen sich mit beliebigen Programmierumgebungen bzw. Frameworks erstellen.

Allerdings sind dieser auf den ersten Blick flexiblen Vorgehensweise vergleichsweise enge Grenzen gesetzt. Diese betreffen zum einen die fehlende native Unterstützung für die Versionierung oder kollaborative Manipulation von Korpusdateien, andererseits die Verarbeitung sehr großer Datenmengen: Stand-alone XML-Editoren, -Parser und -Prozessoren[86] erfordern erfahrungsgemäß für das Validieren oder Transformieren des umfangreichen Outputs linguistischer Tagger, der je nach Größe eines Korpusarchivs rasch ein physikalisches Volumen von mehreren Gigabyte erreicht, immer wieder zeitaufwändige Anpassungen. Kann auf stringente Konsistenzprüfungen verzichtet werden, kommen auch nicht-validierende Skriptsprachen-Ergänzungen wie das Perl-Modul Twig oder Unix-Werkzeuge mit eingebauter Unterstützung für reguläre Ausdrücke wie grep oder sed in Betracht.

Grundsätzlich jedoch generiert die unmittelbare Exploration XML-kodierter Annotationsphänomene ohne professionelle Indizierung außerordentlich lange Antwortzeiten, sowohl auf RAID-Festplatten wie auf schnelleren Solid-State-Disks. Multidimensionale Abfragen verschiedenartiger XML-Strukturen, wie sie bei Mehrebenen-Annotationen üblich sind, über Dateigrenzen hinweg lassen sich ohne erheblichen programmatischen Aufwand kaum realisieren. Dieses Verhalten entspricht den Erwartungen an die festplattengestützte Recherche auf Rohdaten, die Cunningham (2000, S. 109) wie folgt formuliert: „Modern corpora, and annotations upon them, frequently run to many mil-

[85] Zu den genannten Standards vgl. z.B. Becher (2009), Tidwell (2008) und Wamsley (2007) sowie www.w3.org. Unter www.w3.org/Tools/HTML-XML-utils/ findet sich dort eine nützliche Sammlung von Werkzeugen (hxcount, hxextract usw.) für den dateibasierten Umgang mit XML-Inhalten.

[86] Prototypisch für dieses Marktsegment lassen sich die De facto-Standards SAX (*Simple API for XML*, ereignisbasiert) bzw. DOM (*Document Object Model*, baumbasiert) nennen. Auf ihnen basieren diverse freie und kommerzielle, validierende und nichtvalidierende XML-Parser, einen funktionalen Mittelweg strebt z.B. die *Streaming API for XML* (StAX) an; vgl. auch www.xml.com/pub/rg/XML_Parsers.

lions of tokens. To enable efficient access to this data the tokens and annotation structures must be indexed."

3.1.2 Hauptspeicherbasierte Lösungen

Hier reicht die Palette von proprietären Softwareentwicklungen bis hin zu hauptspeicherresidenten Datenbanksystemen (*in-memory databases, IMDB*). Der gemeinsame Lösungsansatz beruht auf der Ablage sämtlicher abfragerelevanter Korpusinhalte im schnellen Hauptspeicher (RAM) eines Rechners sowie auf der damit verbundenen drastischen Reduzierung der Suchzeiten, unabhängig vom letztlich eingesetzten Query-Algorithmus. Ad-hoc-Abfragen kleiner bis mittlerer Datenmengen sind auf diese Weise performant durchführbar. Der entscheidende Vorteil im Vergleich mit dateisystembasierten Ansätzen besteht in der Umgehung von hardwarebedingten Engpässen, die durch sequenzielle Lesezugriffe auf Festplattenlaufwerke entstehen. Retrievalalgorithmen können unmittelbar auf Datenstrukturen (Einzelwörter, Wortfolgen, Sekundärdaten etc.) zugreifen bzw. diese gruppieren und anordnen. Insbesondere für nicht-indizierbare reguläre Ausdrücke eröffnet der wahlfreie Speicherzugriff eine attraktive Option zur Vermeidung linear oder gar exponentiell ansteigender Abfragekosten. Exemplarisch hierfür steht die erste Version der Korpusdatenbank ANNIS (ANNotation of Information Structure), die im Kontext des in Berlin und Potsdam angesiedelten Sonderforschungsbereichs 632 („Information Structure: The Linguistic Means of Structuring Utterances, Sentences and Texts") entwickelt wurde: „[…] the application reads its data from files at startup and keeps them completely in memory during a session. This was motivated by the criterion of speed; in particular, query execution profits a lot from ANNIS being memory-based." (Dipper et al. 2004, S. 253)

Indes leidet die ausschließliche Verwendung von physischem Arbeitsspeicher – das Auslagern (*swapping*) in virtuelle Hauptspeicherbereiche wäre naheliegenderweise kontraproduktiv – auch unter wesentlichen Einschränkungen: RAM ist ein flüchtiges, nicht-persistentes Speichermedium, in dem Suchdaten und -ergebnisse nach jedem Systemstart neu geladen bzw. berechnet werden müssen; Systemabstürze haben im ungünstigsten Fall einen umfassenden Datenverlust zur Folge. Weiterhin lässt sich Arbeitsspeicher nicht ähnlich unbegrenzt erweitern wie Massendatenspeicher, so dass die RAM-basierte Verwaltung sehr großer Sprachressourcen im Terabyte-Bereich selbst beim Einsatz von Kompressionstechniken derzeit kaum realisierbar erscheint.

Als Konsequenz empfiehlt sich die Nutzung von Hauptspeicherstrukturen in erster Linie als flankierende Maßnahme im Kontext anderer Architekturen. Entsprechend hybride Retrievallösungen ermöglichen eine projektgerechte

Balance zwischen Abfrageeffizienz, Datenpersistenz und Erweiterbarkeit. In diesem Sinne setzt beispielsweise ANNIS neben RAM-Operationen primär auf ein relationales Datenbankmanagementsystem als dauerhaften Datenspeicher. Auch die bereits eingeführte Leipzig Corpora Collection (LCC) nutzt hauptspeicherbasierte Strukturen für Berechnungen, die mit häufigen Festplattenzugriffen unzumutbare Laufzeiten generieren würden: „Ein großes Problem stellt die Analyse der großen Datenmengen an sich dar: Die große Zahl der Lexikoneinträge erlaubt zwar den Einsatz von Algorithmen mit linearer Komplexität, d.h. deren Laufzeit linear mit der Zahl der bearbeiteten Einträge wächst wie etwa Programme zur morphologische Zerlegung, die jedes Wort einzeln bearbeiten, schafft aber Probleme bei Algorithmen höherer (also z.B. quadratischer) Komplexität. Beispielsweise muss bei der Bestimmung der Kollokationen jedes in Frage kommende Wortpaar untersucht werden, also jedes Wort mit jedem anderen verglichen werden. Dies ist aus Zeitgründen nur noch mit speziellen Algorithmen möglich, die alle notwendigen Daten komprimiert im Arbeitsspeicher halten.“ (Quasthoff/Wolff 1999).

3.1.3 Volltextsuchmaschinen

Im Vordergrund der beiden bislang charakterisierten Modelle standen physische Aspekte des Datenspeichers, ohne Festlegung auf ein bestimmtes Zugriffsparadigma. Dies ändert sich zumindest partiell bei Korpusrecherchen unter Nutzung von Volltextsuchmaschinen. Unter dieser Bezeichnung firmiert spezialisierte Software für das Information Retrieval wie beispielsweise Apache Lucene (eine gleichermaßen populäre und leistungsstarke freie Software zur Volltextsuche, die aufgrund ihrer Skalierbarkeit in textzentrierten und datenintensiven Internetportalen wie z.B. Twitter zum Einsatz kommt, oft in Kombination mit der Serverplattform Apache Solr), Managing Gigabytes (hierauf basiert z.B. die Verwaltung und Abfrage von DeReKo vermittels COSMAS II) oder Oracle Text (bietet spezialisierte Index- und Suchanfragetypen für die Verwaltung semi-strukturierter Texte in Datenbanktabellen und externen Dateien).[87] Volltextsuchmaschinen unterstützen die wortbasierte Suche und Filterung in mehr oder weniger komplexen Dokumenten. Texte werden intern in eine systemspezifische Repräsentationsform umgewandelt, indiziert und unter Zuhilfenahme von Abfragemodellen wie dem Booleschen oder

[87] Zu den genannten Systemen vgl. Hardt (2004), Witten et al. (1999) bzw. Bryla/Loney (2013). Ebenfalls Lucene-basiert ist beispielsweise die am Institute of Dutch Lexicology (INL) in Leiden entwickelte Suchmaschine „BlackLab“, vgl. https://github.com/INL/BlackLab. Das Digitale Wörterbuch der deutschen Sprache (DWDS) nutzt die linguistische Volltextsuchmaschine „DDC-Concordance“ (Sokirko 2003).

dem Vektorraum-Modell recherchierbar gemacht. XML-Dokumente, also beispielsweise aufbereitete Korpustexte oder -annotationen, lassen sich auf diese Weise unter Einbeziehung einzelner Auszeichnungshierarchien durchsuchen.

Der originäre Anwendungsfokus liegt auf der horizontalen Verkettung von Sprachelementen, in der Regel von Einzelwörtern oder Wortgruppen. Dabei kommen invertierte Indizes zum Einsatz, die Textwörter als Zugriffsschlüssel und deren Position im Text als Werte speichern, um ein effizientes Auffinden von Belegstellen zu ermöglichen. Eine typische Verarbeitungs-Pipeline besteht aus folgenden Schritten: Zunächst wird ein Dokument eingelesen, eventuell vorgefundenes Markup geparst, und der Reintext in Einzelwörter segmentiert. Optional können weitere Verarbeitungsschritte zwischengeschaltet werden, etwa das Ausfiltern von Stoppwörtern, eine Vereinheitlichung von Schreibvarianten, Lemmatisierung etc. Für Korpusrecherchen auf unverfälschten, authentischen Sprachdaten sind diese Zwischenschritte allerdings von eher untergeordneter Bedeutung. Die Ergebnisse der Segmentierung/Tokenisierung konstituieren eine invertierte Indexliste. Die darin kodierten Angaben über die Wortpositionen erlauben eine Formulierung und Gewichtung von Retrievalanfragen mit mehreren Suchwerten. Beispielsweise helfen Entfernungsoperatoren bei der Eingrenzung von Suchergebnissen auf diejenigen Dokumente, in denen die spezifizierten Suchwörter in einer bestimmten Abfolge und mit einem bestimmten Abstand erscheinen. Weiterhin kommen beim Retrieval interne Indexstatistiken, etwa zum Type-Token-Verhältnis, zum Einsatz.

Volltextsuchmaschinen stellen eine erprobte und skalierbare Technologie für die Suche nach sequenziell angeordneten Sprachdaten dar. Für das (vertikale) Retrieval in mehrfach annotierten Korpora sind allerdings jedoch z.T. erhebliche Anpassungen notwendig, vgl. Ghodke/Bird (2008) zur Nutzung von IR-Technologie für syntaktische Annotationen. Ein häufiges Desiderat besteht etwa in der Ersetzung eines vordefinierten Tokenisierers durch externe Parser/Tagger. Sollen darüber hinaus konkurrierende Annotationssysteme einbezogen werden (vgl. z.B. Chiarcos et al. 2009), so erweist sich eine multiple Segmentierung als notwendig, die im parallelen Aufbau unterschiedlicher Indexstrukturen resultiert. Die Suche über verschiedene Annotationsebenen hinweg fällt dann nicht mehr in den Bereich der üblicherweise von Volltextsuchmaschinen abgedeckten Funktionalitäten und erfordert grundlegende Modifikationen des Retrievalalgorithmus. Kommt schließlich noch der Anspruch hinzu, Beziehungen zwischen mannigfaltigen inner- und außersprachlichen Metadaten erschließbar zu machen, stoßen reine Volltextsuchmaschinen vollends an ihre Grenzen, da komplex strukturierte Sekundärdaten nicht in den für sie üblichen Indexstrukturen ablegbar sind. Aus diesem Grund erweitern entsprechend am-

bitionierte Korpusretrievalprojekte ihre technologische Basis um hierauf spezialisierte Software, wie zum Beispiel Datenbankmanagementsysteme.[88]

3.1.4 Datenbankbasierte Korpusverwaltung

Datenbankmanagementsysteme (DBMS) arbeiten als zusätzliche Softwareschicht für die Datenhaltung zwischen Betriebssystem und Endanwendung. In der klassischen Drei-Schichten-Architektur (*three tier architecture*) der Software-Entwicklung entspricht dies der Positionierung als Backend (*data tier*). Das DBMS unterstützt dabei die logische Organisation, persistente Speicherung und den physischen Zugriff auf umfangreiche Datensammlungen unter Verwendung eines mehr oder weniger standardisierten Spektrums von Tabellen- und Indextypen, Abfragesprachen etc. Insbesondere stellt es grundlegende Funktionalitäten wie Datensicherheit (durch integriertes Rechte-, Backup- und Replikations-Management), Datenintegrität (durch Definition von Regeln zur Vermeidung unerlaubter bzw. inkonsistenter Datenmanipulation) und Mehrbenutzerfähigkeit (durch Verwaltung von Datensperren und konkurrierender Transaktionen) sicher.[89] Im Bedarfsfall organisiert das DBMS eine verteilte Datenhaltung sowie den Einsatz integrierter Anfrage-Optimierer und Caching-Mechanismen für das Retrieval.

Weitverbreitet ist das relationale Datenmodell, das Datensätze in Form von in Zeilen und Spalten gegliederten Tabellen abspeichert und relationale Algebra als abstrakte Grundlage des operationalisierten Zugriffs verwendet. Für die Manipulation und Abfrage von Datensätzen hat sich darauf aufbauend mit der Structured Query Language (SQL) ein präziser und systemübergreifender Sprachstandard etabliert. Relationale Datenbankmanagementsysteme (RDBMS) erlauben die konsistente Implementierung eines semantischen (konzeptuellen) Schemas, das den abzubildenden Weltausschnitt sowie funktionale Beziehungen definiert, etwa auf Basis einer Entity-Relationship-Modellierung.[90] Physische Datensätze – in unserem Falle also segmentierte Rohtexte eines Korpus sowie multidimensionale Annotations- und Metadaten – sind über Primärschlüssel eindeutig referenzierbar; unterschiedliche Bezie-

---

[88] Das am IDS Mannheim entwickelte KorAP (Korpusanalyseplattform der nächsten Generation)-Framework etwa nutzt zu diesem Zweck neben Lucene optional eine Graphdatenbank; vgl. Bański et al. (2013, 2014), Kupietz et al. (2017) und Schnober (2012).

[89] Für Anforderungen an datenbankbasierte Transaktionen hat sich diesbezüglich das Akronym ACID (Atomicity, Consistency, Isolation und Durability) etabliert. Zu Datenbanksystemen vgl. einführend z.B. Date (2004); Elmasri/Navathe (2009); Kemper/Eickler (2015); Rahm (1993); Rahm/Vossen (2003); Ritter et al. (Hg.) (2015).

[90] Vgl. z.B. Chen (2002) und Date/Darwen (2007).

hungstypen (1:n, n:m etc.) lassen sich unter Verwendung sogenannter Fremdschlüssel oder durch Erstellung zusätzlicher Tabellen/Relationen abbilden.

Neben relationalen Systemen haben sich im Verlauf der vergangenen Jahrzehnte diverse weitere Typen von Datenbanksoftware konstituiert. Seit den 1990er Jahren sind dies etwa objektorientierte Implementierungen, die komplexe Informationsobjekte nicht in aufwändig abzufragende Relationen abbilden, sondern ohne Zerlegung als Ausprägungen einer Klasse modellieren, und in der Folge nahtlose Anbindungen an objektorientierte Entwurfs- und Programmierumgebungen unterstützen. Mit der mittlerweile umfassenden Verbreitung der Auszeichnungssprache XML rücken weiterhin XML-Datenbanken in den Fokus strukturierter Informationsspeicherung, da sie baumartige XML-Instanzen (also beispielsweise Standoff-Annotationen von Sprachkorpora) unmittelbar in Form nativer Datentypen ablegen und deren Hierarchie mit Hilfe von einschlägiger XML-Technologie wie XPath oder XQuery traversieren können. Gemeinsam mit anderen dokumentenzentrierten Datenbanktypen gehören sie zur Familie der NoSQL („Not only SQL")-Systeme, die auf relationale Tabellenstrukturen verzichten und dadurch datenintensive Abfragen oder Änderungen vergleichsweise ressourcenschonend abarbeiten können. Als typische Eigenarten von NoSQL-Implementierungen gelten der Einsatz schemaloser Datenmodelle sowie die Verteilung umfangreicher Daten innerhalb von Rechnerverbünden. Weitere Spezialisierungen in diesem Sinne sind Graphdatenbanken, die annotierte Korpora als Netzstrukturen mit Knoten und Kanten abbilden. Einzelne Token-Knoten sind in diesen Fällen gemeinhin mit ihren unmittelbaren Vorgängern und Nachfolgern in der linearen Textstruktur verbunden sowie mit Type-Knoten, die z.B. lemmaspezifische Angaben enthalten.[91] Ebenfalls unter das NoSQL -Etikett fallen Key-Value- und spaltenorientierte Datenbanken sowie multi-dimensionale Speicherformen (MOLAP). Zur Gewährleistung rascher Antwortzeiten verzichten einzelne Systeme gemäß des BASE (Basically Available, Soft state, Eventual consistency)-Prinzips unter bestimmten Umständen auf höchstmögliche Konsistenz.

Mehrere groß angelegte sprachwissenschaftliche Korpusprojekte – in Deutschland beispielsweise die Leipzig Corpora Collection (LCC) oder die Such-

[91] Entsprechend konstruierte Datenbanken spielen ihre Stärken insbesondere in Situationen aus, in denen nicht ein Gesamtkorpus traversiert, sondern – etwa für Kollokationsanalysen – der lokale Kontext eines Knotens exploriert werden soll; vgl. z.B. Efer (2015) oder Pęzik (2014) bzw. die HASK Collocation Dictionaries unter http://pelcra.pl/hask_en/. Einen Vergleich von Graphdatenbanken und relationalen Ansätzen für ein technologisch vergleichbares Anwendungsszenarium im Umfeld künstlicher neuronaler Netze bietet Fries et al. (2014).

und Visualisierungsplattform ANNIS[92] – verwalten Korpusinhalte mit Hilfe relationaler Datenbanktechnologie. Aufgrund der Flexibilität der zugrundeliegenden Modellierungsmechanismen ist es damit möglich, Beziehungen zwischen segmentierten Primärdaten (Wörtern, Wortgruppen, Sätzen, Texten usw.) und beliebigen Annotationen abzubilden. Davon profitieren in besonderer Weise Vorhaben, die mehrere Annotationsvarianten parallel verfügbar machen wollen.

Idealerweise werden bereits während der konzeptuellen Modellierungsschritte inhomogene Layer aufgetrennt. Maschinell erstellte Standoff-Annotationen enthalten häufig verschiedenartige Typen von Informationen – neben der Wortartenklassifizierung etwa auch morphologische oder syntaktische Informationen. Erst eine typgerechte Relationierung ermöglicht dann feingranulare Recherchen nach syntaktischen Konstituenten, Flexionsangaben, Genus, Numerus, Kasus usw.; gleiches gilt für die Disambiguierung unterschiedlicher Lesarten. Datenbanksysteme profitieren im Übrigen davon, dass Korpusinhalte ein vergleichsweise stabiles Dateninventar bilden und nur selten verändert werden (müssen). Einmal angelegte DBMS-Indizes bleiben dadurch tendenziell langfristig unfragmentiert und effektiv.

Zusätzlich interessant wird der relationale Ansatz für spätere linguistisch motivierte Explorationen durch die explizite Kodierung ansonsten nur implizit vorhandener Informationen, etwa von Erst- oder Letztwörtern in Sätzen: „Relational databases can be used to create large corpora that provide both very good search performance and a wide range of queries“ (Davies 2005, S. 307). Weiterhin ermöglicht er die flexible Bereitstellung von Positionsangaben für jedwede digital erfassbaren Sprachphänomene bzw. -segmente. Und schließlich ist er für die unentbehrliche Anbindung außersprachlicher Metadaten essentiell: „Linguistic databases typically include important bodies of information whose structure has nothing to do with the […] particular recording, nor with the sequence of characters in any particular text. […] This side information is usually well expressed as a set of relational tables.“ (Bird/Liberman 1999, S. 41)

[92] Zu ANNIS vgl. Dipper et al. (2004), Krause/Zeldes (2014), Rosenfeld (2010) und Stede (2007); zur LCC vgl. Biemann et al. (2007), Goldhahn et al. (2012), Quasthoff et al. (2006) und Richter et al. (2006) sowie Kapitel 2. Weiterhin existieren experimentelle Implementierungen relationaler Korpusdatenbanken, vgl. z.B. Bindernagel (2007), Ekoniak (2006), Künneth (2001) und Zierl (1997) sowie Modellierungen und Evaluierungen relationaler Lösungen für syntaktische Annotationen, vgl. z.B. Chubak/Rafiei (2012), Yoshikawa/Amagasa (2001) oder Zhang et al. (2001). Auch die populäre Sketch Engine (Kilgarriff et al. 2014) verwendet mit Manatee (Rychlý 2007) ein Datenbanksystem für die Speicherung und Indizierung von umfangreicher Textkorpora (Pomikálek et al. 2012).

3.2 Ein Referenzsystem für die relationale Korpusspeicherung

In Anbetracht der für Sprachanalysen benötigten umfangreichen Datenvolumina, des Bedarfs an gleichermaßen horizontalen wie vertikalen Rechercheoptionen, der Heterogenität der zu indizierenden Sekundärdaten sowie der für nachhaltige Untersuchungen unverzichtbaren Datensicherheit und -integrität konzentrieren wir uns im Folgenden auf die Praktikabilität relationaler Datenbanktechnologien für die Verwaltung von Mehrebenen-Korpora. Die Zielsetzung umfasst Design und Implementierung einer prototypischen Referenzplattform, die für eine effiziente Durchführung der linguistisch motivierten Korpusabfragen unseres Anforderungskatalogs herangezogen werden kann.

Die Entscheidung pro RDBMS ist im aktuellen Forschungsumfeld nicht alternativlos. So führt Ghodke/Bird (2010, S. 267) analog zu unseren obigen Charakterisierungen aus:

> Many existing systems load the entire corpus into memory and check a user-supplied query against every tree. Others avoid the memory limitation, and use relational or XML database systems. Although these have built-in support for indexes, they do not scale up either.

Es besteht also Evaluationsbedarf, dem wir nachfolgend entsprechen. Dabei streben wir den Beleg an, dass der relationale Ansatz bei angemessener Umsetzung – insbesondere hinsichtlich der Indizierung und unter Einsatz eines für Sprachdaten optimierten Retrievalmodells – für den korpuslinguistischen Einsatz praxistauglich und mit zeitgemäßer Hardware effizient ist.

Dass (relationale) Korpusdatenbanken gerade bei komplexeren Abfragespezifikationen bisweilen an ihre Grenzen stoßen und entsprechende Evaluierungen wünschenswert erscheinen, konstatiert z.B. auch Dipper et al. (2004) angesichts eigener Erfahrungen mit der ANNIS-Datenbank: „The results […] show that the overall performance of the ANNIS prototype has still to be improved. Here, more complex queries […] require unacceptable processing times." An dieser Stelle knüpfen wir an und thematisieren nachfolgend den Aufbau eines prototypischen Referenzsystems für korpuslinguistische Datenbanksysteme, gefolgt von einer empirischen Überprüfung grundlegender Designentscheidungen.

3.2.1 Behandlung von Primär- und Sekundärdaten

Das Referenzsystem basiert inhaltlich auf dem Deutschen Referenzkorpus DeReKo. Sämtliche Sprachbelege entstammen der DeReKo-Freigabe vom

15.04.2014 (DeReKo-2014-I).[93] Die aufbereiteten Primärdaten liegen als wohlgeformte XML-Instanzen gemäß des IDS-Textmodells vor. Jeweils eine XML-Datei fasst die Inhalte eines Subkorpus – bei Periodika in der Regel für jeweils ein Kalenderjahr – zusammen, wobei Korpora bzw. Subkorpora stets aus einem oder mehreren Dokumenten und Dokumente aus einem oder mehreren Texten bestehen können. Jeder Text ist durch eine eindeutige Sigle in Form einer Buchstaben-Ziffern-Kombination charakterisiert, die an führender Stelle ein Kürzel des betreffenden Korpus enthält, z.B.:

```
<textSigle>FOC13/JAN.00009</textSigle>
```

Weitere Metadaten im Textheader kodieren Angaben zu Titel, Autor, Publikationsdatum und -ort sowie inhaltliche Beschreibungen zu Texttyp (Ausprägungen: *Bericht, Interview, Kommentar, Tageszeitung, Zeitschrift, Zitat* etc., ggf. mit Subkategorien) und -thema, die ebenfalls in die Datenbank importiert und für Analysefunktionalitäten eingesetzt werden sollen:

```
<t.title assemblage="regular">FOC13/JAN.00009 FOCUS, 07.01.2013,
 S. 115; BESTSELLER</t.title>
<h.author>RU: Stefan Ruzas</h.author>
<imprint>
  <publisher>Hubert Burda Media</publisher>
  <pubDate type="year">2013</pubDate>
  <pubDate type="month">01</pubDate>
  <pubDate type="day">07</pubDate>
  <pubPlace>München</pubPlace>
</imprint>
<textDesc>
  <textType>Zeitschrift: Wochenzeitschrift</textType>
  <textTypeRef/>
  <textTypeArt/>
  <textDomain>Kultur</textDomain>
  <column>KULTUR UND LEBEN, MEDIEN</column>
</textDesc>
```

Für die Pflege der DeReKo-Korpora bedient sich der Programmbereich Korpuslinguistik am Institut für Deutsche Sprache (IDS) in Mannheim zusätzlicher Metadaten, die in einer separaten Datenbank hinterlegt sind; Kupietz/Keibel (2009) bieten eine diesbezügliche Dokumentation. Aus dieser Quelle fließen folgende textspezifische Angaben in unser Referenzsystem ein:

[93] www.ids-mannheim.de/kl/projekte/korpora/releases.html.

– Textsorte/Genre; Ausprägungen: *Abhandlung, Agenturmeldung, Beratung, Bericht, Zeitung etc.*
– Ressort; Ausprägungen: *Astronomie, Auto, Regional, Sport, Unterhaltung* etc.
– Topic; Ausprägungen: *Freizeit_Unterhaltung:Reisen, Fiktion:Vermischtes* etc.[94]
– Land; Ausprägungen: *A, CH, D*

Weitere Metadaten werden innerhalb der Datenbankumgebung semi-automatisch generiert:

– Medium als Ableitung aus Texttyp, Genre und Ressort; Ausprägungen: *Publikumspresse, Bücher, sonstige Printmedien, Gesprochenes, Internet/Wikipedia*
– Register als Ableitung aus Texttyp und Ressort; Ausprägungen: *Pressetexte, Gebrauchstexte, Literarische Texte*
– Domäne als inhaltliche Topic-Zusammenfassung; Ausprägungen: *Fiktion, Kultur/Unterhaltung, Mensch/Natur, Politik/Wirtschaft/Gesellschaft, Technik/Wissenschaft, unklassifizierbar*
– Geburts- sowie ggf. Wohnort des Autors auf Basis eigener Recherchen
– Region als Zuweisung aller Orte zu je einer Großregion; Ausprägungen: *Nordwest, Nordost, Mittelwest, Mittelost, Mittelsüd, Südwest* (*einschließlich Schweiz*), *Südost* (*einschließlich Österreich*)

Der aufbereitete Primärtext erscheint im Body der XML-Korpustexte. Er enthält Auszeichnungen zu Textstruktur (Zwischenüberschriften, Absatz, Satz), die nicht unmittelbar in die Datenbank übernommen werden sollen. Die Segmentierung des Fließtexts erfolgt auf Basis der Standoff-Annotationen der externen Tagger/Parser. Bei Bedarf bietet sich darüber hinaus der Rückgriff auf die Kennzeichnung von Überschriften (Elementtyp „head“) an, um auf diese Weise „ungrammatische“, nicht-satzförmige Konstruktionen auszufiltern:

```
<text>
  <front/>
  <body>
    <div n="0" complete="y" type="Zeitung">
      <head type="main">BESTSELLER</head>
      <head type="sub">FOLK-POP Casting mit Folgen</head>
      <p><s>Sie lief gar nicht gut, die Casting-Show „X Factor"
       des Fernsehsenders Vox Ende 2012: Zoff in der Jury und
       derart miese Quoten, dass die Zahl der Live-Shows prompt
       von acht auf vier zusammengestrichen wurde.</s> <s>Und
```

[94] Weiß (2005) beschreibt die thematische DeReKo-Erschließung mit Hilfe eines semiautomatischen Verfahrens, das die Anwendung von Textmining (Dokumentclustering) und die Verortung von Clustern in einer Themenontologie beinhaltet.

```
        trotzdem stehen die Sieger plötzlich auf Platz eins der
        deutschen Albumcharts.</s> <s>Mrs. Greenbird nennt sich das
        Folk-Pop-Duo Sarah Nücken und Steffen Brückner, und ihre
        Herkunft überrascht: Die Rheinländer sind Christen, die
        sich in ihren evangelischen Freikirchen mit Namen wie
        Mosaik Düsseldorf auch als „Lobpreisleiter" und Musiker
        engagieren.</s> <s>Beide betonen aber: „Wir haben unsere
        Seele nicht verkauft." </s></p>
       <p>Lobgesang Mrs. Greenbird sind nicht nur Casting-Sieger,
        sondern auch bekennende Christen </p>
    </div>
  </body>
</text>
```

Die Originaltexte liegen dreifach annotiert vor, mit einem Gesamtdatenvolumen im einstelligen Terabyte-Bereich. Zum Einsatz kamen der *Connexor Machinese Phrase Tagger*, *TreeTagger* sowie der *Xerox Incremental Parser* (vgl. Abschnitte 2.3.1 und 2.4) Als Ergebnis existieren für jedes Teilkorpus drei parallele Standoff-Annotationen, jeweils in XML-Notation. In die Datenbasis aufgenommen werden sollen daraus jeweils die Oberflächenform (*token*), die Grundform (*lemma*) sowie die Wortartklassifikation (*part-of-speech*).

Ergänzend hinzu kommen Positionsangaben sowie – sofern vom Tagger bereitgestellt – Konfidenzwerte oder weitere morpho-syntaktische Angaben. Die Struktur der berücksichtigten Annotationsformate wurde bereits ausführlich vorgestellt, so dass an dieser Stelle ein verkürzter Ausschnitt genügen soll. Die initialen Wörter des ersten Satzes aus obigem Fließtext stellen sich gemäß *TreeTagger*-Notation folgendermaßen dar:

```
<lexeme id="5000" pos="81827" len="3">
  <surface-form>Sie</surface-form>
  <sense id="0">
    <base-form>Sie|sie|sie</base-form>
    <part-of-speech conf="1.000000">PPER</part-of-speech>
  </sense>
</lexeme>
<lexeme id="5001" pos="81831" len="4">
  <surface-form>lief</surface-form>
  <sense id="0">
    <base-form>laufen</base-form>
    <part-of-speech conf="1.000000">VVFIN</part-of-speech>
  </sense>
</lexeme>
```

Unser korpuslinguistisches Fundament besteht mithin aus einer Hierarchie korpus-, text-, wortgruppen- und wortbasierter Angaben. Gelegentlich schließen diese auch Informationen zu untergeordneten Segmentebenen ein, etwa durch die Kennzeichnung der Einzelglieder von Komposita oder der Affixe mit grammatikalischer Bedeutung in den Xerox-Grundformen.

Um eine konsistente Anpassung der Datengrundlage an wechselnde Fragestellungen garantieren zu können, führen wir eine virtuelle Dreiteilung der Korpusbasis ein:[95]

– Das umfangreiche Untersuchungskorpus (UK) dient als primäre Datenbasis. Es enthält eine breite Palette unterschiedlicher Sprachbelege (Belletristik ab 1955, Pressetexte ab 1990) und ist aufgrund seiner Größe – ca. 8 Milliarden laufende Wortformen – insbesondere für die Recherche nach seltenen Phänomenen oder für explorative Studien prädestiniert.
– Das ausgewogene Korpus (AK) stellt eine Teilmenge (< 1%) des Untersuchungskorpus dar. Speziell für die Analyse von Variationsparametern strebt es eine gleichmäßige Proportionierung verschiedener Strata (Medium, Register, Domäne, Region) an. Um einzelne Merkmalsausprägungen nicht signifikant vernachlässigen zu müssen, liegt sein Umfang immer noch im zweistelligen Millionen-Token-Bereich.
– Das Reservekorpus (RK) nimmt Texte auf, die zwar nicht für unmittelbar anstehende Untersuchungen benötigt werden, aber perspektivisch eine Rolle spielen könnten und deshalb verfügbar bleiben sollen.

3.2.2 Konzeptuelle Datenmodellierung

Anknüpfend an die Dokumentation der relevanten Primär-, Annotations- und Metadaten stellt sich die Frage nach der Art ihrer internen Repräsentation in einer Korpusdatenbank. Ein naheliegender Gedanke wäre die Verwendung nativer XML-Datentypen, schließlich liegen die Inhalte bereits mehrheitlich in Form wohlgeformter XML-Instanzen vor. Allerdings präferieren wir XML eher als systemübergreifendes Austauschformat denn als Datentyp für die effiziente Recherche in sehr großen Repositorien. Des Weiteren enthalten die von den Taggern gelieferten Standoff-Dateien lediglich eine Untermenge der insgesamt abfragerelevanten Informationen und z.B. keine außersprachlichen Metaangaben zu Korpus- oder Texttyp, Publikationsort, Datum usw. Es empfiehlt sich also, die gesammelten Quelldatenstrukturen insgesamt so zu analysieren und abzubilden, dass in einem Folgeschritt eine stabile Implementierung und schlussendlich ein präzises Retrieval realisierbar sind. Eventuell

[95] Zur Begündung des Korpusdesigns vgl. auch Bubenhofer et al. (2014, S. 54ff.).

aufgefundenen Redundanzen sollen dabei ebenso wie nicht abfragerelevante Angaben ausgeschlossen werden.

Die konzeptuelle Datenmodellierung soll den zu erfassenden Weltausschnitt mit Hilfe eines theoretischen Instrumentariums formal umfassend beschreiben, so dass sich daraus die für unsere Aufgabenstellung benötigten Tabellenstrukturen ableiten lassen. Wir folgen dabei der Notation von Peter Chen, dem Begründer der Entity-Relationship-Diagramme (ER-Diagramme; vgl. Chen 1976). Abbildung 10 veranschaulicht Objektinventar, Eigenschaften und funktionale Abhängigkeiten unseres Modellentwurfs: Die erfassten Entitätstypen werden durch Rechtecke, deren Attribute in Ellipsenform dargestellt. Beziehungstypen, die Zusammenhänge zwischen den Entitätstypen ausdrücken, werden als Rauten abgebildet. Die an den Verbindungslinien zwischen den Entitätstypen ergänzten Zahlen und Buchstaben geben Kardinalitäten an. Sie spezifizieren, wie oft eine Entität in einer Beziehung auftreten kann, wobei „n“ bzw. „m“ für „unbegrenzt“ steht.

Unser Modell bildet sämtliche Sprachdaten top-down von Korpus- bis auf Wortebene herab als eigenständige Objekte mit 1:n-Beziehungen und nicht etwa als n-Gramme ab;[96] die im IDS-Textmodell zwischen Korpus und Text liegende Dokumentebene lassen wir mangels akuter Abfragerelevanz außen vor. Ein Korpus besteht demnach aus einem oder mehreren Texten, ein Text aus einem oder mehreren Sätzen und ein Satz aus einem oder mehreren Wörtern. Die Zuordnung zu virtuellen Korpora (Untersuchungskorpus UK, Reservekorpus RK, ausgewogenes Korpus AK) erfolgt durch den Attributtyp „Korpustyp“ auf Textebene, weil sich Texte aus identischen Subkorpora basierend auf variierenden textspezifischen Metadaten hier unterschiedlich verteilen.

[96] Die Evaluierung dieser Designentscheidung in Form eines Vergleichs der Abfrageperformanz von n-Grammen bzw. relationierten Tabellen folgt in Abschnitt 3.3.1.

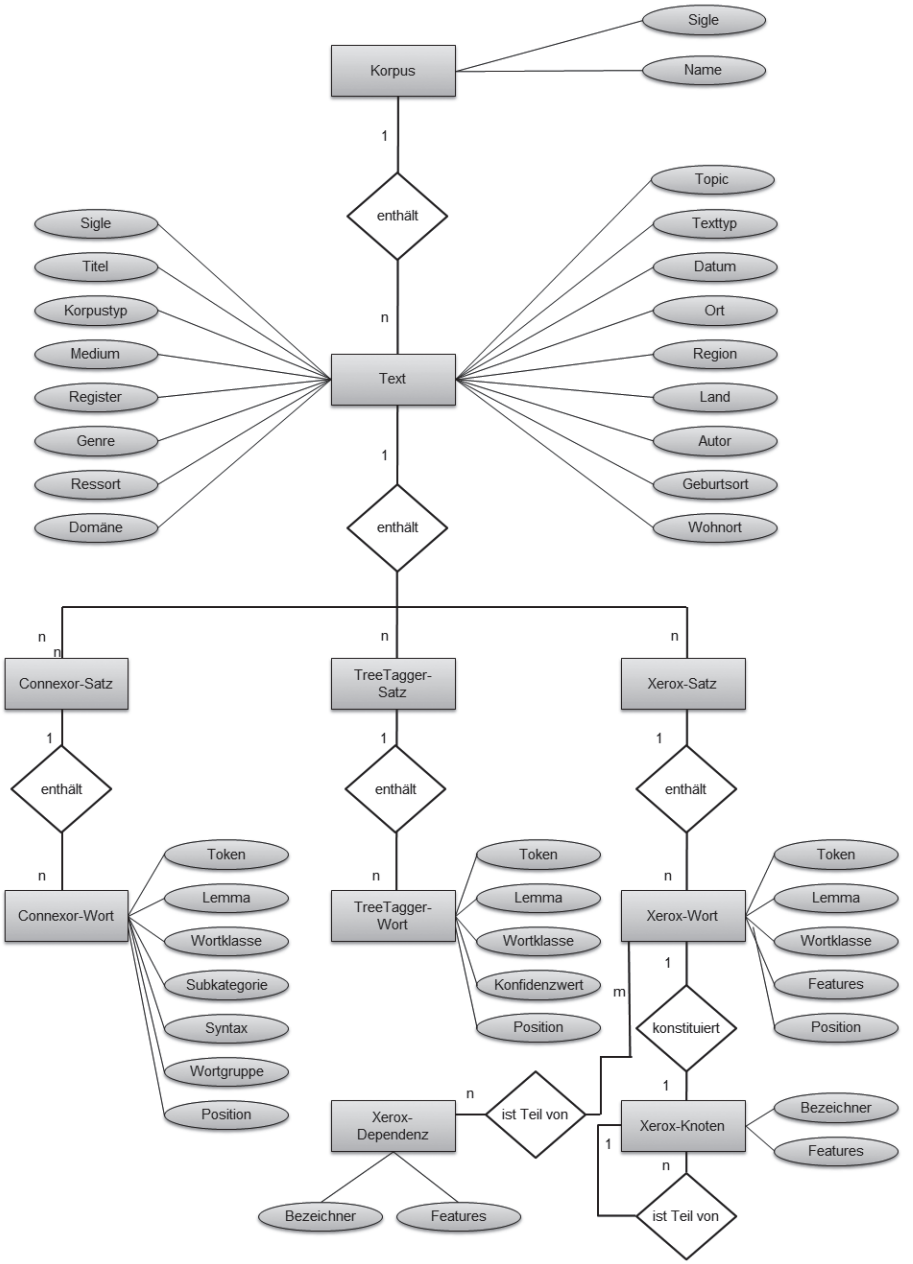


Abb. 10: Semantisches Datenmodell des Referenzsystems

Aufgrund der nicht durchgehend kongruenten Segmentierung unterschiedlicher Taggingwerkzeuge unterscheidet das Modell ab der Satzebene explizit zwischen Connexor-, TreeTagger- und Xerox-Objekten. Dabei liegt ihm eine bewusste Vereinfachung zugrunde, um mit multiplen, hinsichtlich Lemma bzw. Wortklasse ambigen Lesarten umzugehen: Falls der XML-Output eines Taggers für eine Wortform mehrere Lesartkandidaten liefert, soll lediglich einer davon im Modell berücksichtigt werden. Idealerweise ist das die mit größter Wahrscheinlichkeit in den Satzkontext passende Variante, etwa gemäß eines vom Tagger mitgelieferten Konfidenzwerts. Als Folgeentscheidung wird jeder laufenden Wortform auch exakt eine Grundform sowie exakt eine Wortklasse zugeordnet. Lemma- und Wortklassenvarianten, die an nicht berücksichtigte Lesarten gekoppelt sind, bleiben außen vor. Diese pragmatische Designentscheidung kann selbstverständlich bei Bedarf anders getroffen werden.

Zu jeder laufenden Wortform gehört zwingend die Angabe ihrer Startposition im Originaltext. Diese kann bei Bedarf für Zugehörigkeitsprüfungen zu Wortgruppen oder anderen übergeordneten Annotationsebenen genutzt werden. Redundante Angaben wie Wortlänge oder Wortendeposition berücksichtigt das Modell hingegen nicht, da sie implizit aus Startposition und Wortformbezeichner folgen.

Weiterhin weisen Wortformen in unserer Modellierung annotationsspezifische Informationen auf. Im TreeTagger-Wortklassenelement finden sich die bereits erwähnten und zwischen null und eins angesiedelten Konfidenzwerte, im Connexor-Output optional morphologische Subkategorien (die XML-Elementtypen sub1 und sub2 werden hier zusammengefasst) sowie syntaktische bzw. Wortgruppen-Angaben. Die vom Xerox-Parser gelieferten Featurelisten werden unverändert, d.h. als Fließtext und nicht weiter relationiert, in das Modell übernommen (z.B. „`CAP XIP_CAP COMMON SG3 NDATS NDATW NACCS NACCW NNOMS NNOMW P3 WEAK STRONG SG NEUT DAT ACC NOM END2 NOUN NOAMBIGUITY LAST FIRST`"). Aus Sicht der standardisierten relationalen Normalisierung widerspricht das zwar der ersten Normalform (1NF), die ausschließlich atomare Wertebereiche für Attributtypen akzeptiert. Für den vorliegenden Zweck erscheint dieses Vorgehen allerdings vertretbar, weil die Einzelwerte nicht für das Retrieval benötigt und ggf. lediglich als unstrukturierter Text präsentiert werden sollen. Auch hier sind spätere Ausdifferenzierungen von Modellierung und Implementierung denkbar.

Eine weitere taggerspezifische Besonderheit des Modells betrifft den konzeptuellen Umgang mit XML-Baumstrukturen.[97] Die geschachtelten Xerox-Annotationen dienen als Grundlage für die Abbildung syntaktischer Retrievalstrukturen: Ein Xerox-Wortknoten kann konstituierendes Bestandteil eines syntaktisch übergeordneten Knotens (d.h. einer Wortgruppe) sein; entsprechend weist unser Entwurf hier eine n:1-Beziehung auf. Redundante Positionsangaben von übergeordneten Knoten im Xerox-Output können für die konzeptuelle Modellierung vernachlässigt werden. Ähnliches gilt bei Dependenzen (Objektrelationen, thematische Relationen etc.). Hier besteht eine n:m-Beziehung: Dependenzstrukturen umfassen potenziell mehrere Wörter und ein Wort darf zu unterschiedlichen Dependenzstrukturen beitragen. Die implizit über die Wort-Entitäten verfügbaren Start- und Endpositionen müssen dabei im Gegensatz zum typisierenden Bezeichner und der Featureliste nicht explizit berücksichtigt werden.

3.2.3 Physisches Datenbankschema

Die Ableitung eines logischen bzw. physischen Datenbankschemas aus der konzeptionellen Modellierung verbindet unsere bisherige systematische Gegenstandsbeschreibung mit der konkreten technischen Umsetzung. Die Zielsetzung besteht in einer systemnahen Beschreibung relationaler Speicherstrukturen, die sich in Form von Datenbanktabellen implementieren lassen. In den nachfolgenden Diagrammen werden die zur Realisierung von Abhängigkeiten erforderlichen Fremdschlüssel (*foreign keys*) als Verbindungslinien zwischen den Tabellen kodiert; Krähenfüße symbolisieren die mehrwertige Seite einer Beziehung. Ein Primärindex (eindeutiges Schlüsselfeld als Realisierung eines identifizierenden Attributs) wird durch eine Raute (#) gekennzeichnet, obligatorisch gefüllte Spalten durch ein Sternchen (*) und optional leere Spalten durch eine Null (0).

Der Ableitungsprozess selbst ist weitestgehend normiert, erlaubt aber Anpassungen an systemspezifische Parameter.[98] Entitäten des konzeptuellen Modells werden gemeinhin in Tabellen überführt, deren Spalten die spezifizierten Attribute widerspiegeln. 1:n-Beziehungen zwischen Entitäten implizieren Master- und Detailtabellen, von denen die letztgenannten die Master-Primärschlüssel in einer zusätzlichen Spalte aufnehmen. Echte n:m-Beziehungen

[97] Zum Mapping von XML-Strukturen in Datenbanken siehe Bourret (2005) oder exemplarisch die Evaluierung in Suri/Sharma (2012).

[98] Dies soll im vorliegenden Fall auch punktuell ausgenutzt werden, beispielsweise aufgrund spezieller Recherchebedürfnisse (etwa nach Satzgrenzen) oder Implikationen des realen Datenbestands.

schließlich werden zumeist in je zwei 1:n-Beziehungen aufgelöst und resultieren folglich in zusätzlichen Tabellen, welche die Primärschlüssel der Beteiligten als Fremdschlüssel verwenden.

Abbildung 11 dokumentiert die Behandlung der Korpus- und Textebene sowie der Connexor-Segmentierungen. Der Primärschlüssel der Korpustabelle (CO_CORPUS in TB_CORPUS) wandert als Fremdschlüssel in die Texttabelle; der Primärschlüssel für Texte (CO_TEXTID in TB_TEXT) übernimmt die gleiche Funktion in der Connexor-Satztabelle (TB_CONNEXOR_SENTENCE). Dort erscheinen zusätzlich die eindeutigen Nummern der jeweils ersten und letzten Satzwörter (CO_FIRSTWORDID bzw. CO_LASTWORDID).

Erklärungsbedürftig erscheint der Umgang mit dem Text-Attribut „Korpustyp". Während es in die Texttabelle – wie in der relationalen Praxis üblich – als Tabellenspalte (CO_CORPUSTYPE) übernommen wird, dient es auf Wortebene als Selektionskriterium für die Aufteilung in drei separate Tabellen: Die Ausprägung „RK" führt zur Speicherung wortbezogener Annotationen der betroffenen Texte in einer Reservekorpus-Tabelle (TB_CONNEXOR_ RK); die Ausprägung „AK" leitet selbige in eine spezielle Tabelle für das ausgewogene Korpus (TB_CONNEXOR_AK); die Ausprägungen „UK" und „AK" – das ausgewogene Korpus stellt eine echte Teilmenge des Untersuchungskorpus dar – dirigieren Wortinformationen in eine Untersuchungskorpus-Tabelle (TB_CONNEXOR_UK). Eine derartige physische Modellierung führt einerseits zu geringfügigen Datenredundanzen, weil die ca. 20 Millionen Wörter des ausgewogenen Korpus doppelt vorgehalten werden. Andererseits erlaubt es für dieses Teilkorpus optional eine vereinfachte Suchstrategie, die von einem für sehr große Datenmengen angelegten Suchalgorithmus unabhängig ist.

Erwähnenswert ist weiterhin, dass etliche textspezifische Metadaten in TB_ TEXT nicht ausgeschrieben, sondern als numerische Werte erscheinen. Die Domänen-Ausprägung „Fiktion" in der Spalte CO_DOMAIN erhält beispielsweise den Wert „1"; anstatt „Kultur/Unterhaltung" erscheint „2" usw. Hintergrund dieser Maßnahme ist der Umstand, dass hier eine Indizierung von Ziffern anstatt längerer alphanumerischer Werte zu tendenziell geringeren Speicheranforderungen führt.[99]

[99] Der Frage, ob solche rein numerischen Werte auch zu einer messbaren Beschleunigung späterer Abfragen führen, widmet sich Abschnitt 3.3.3.

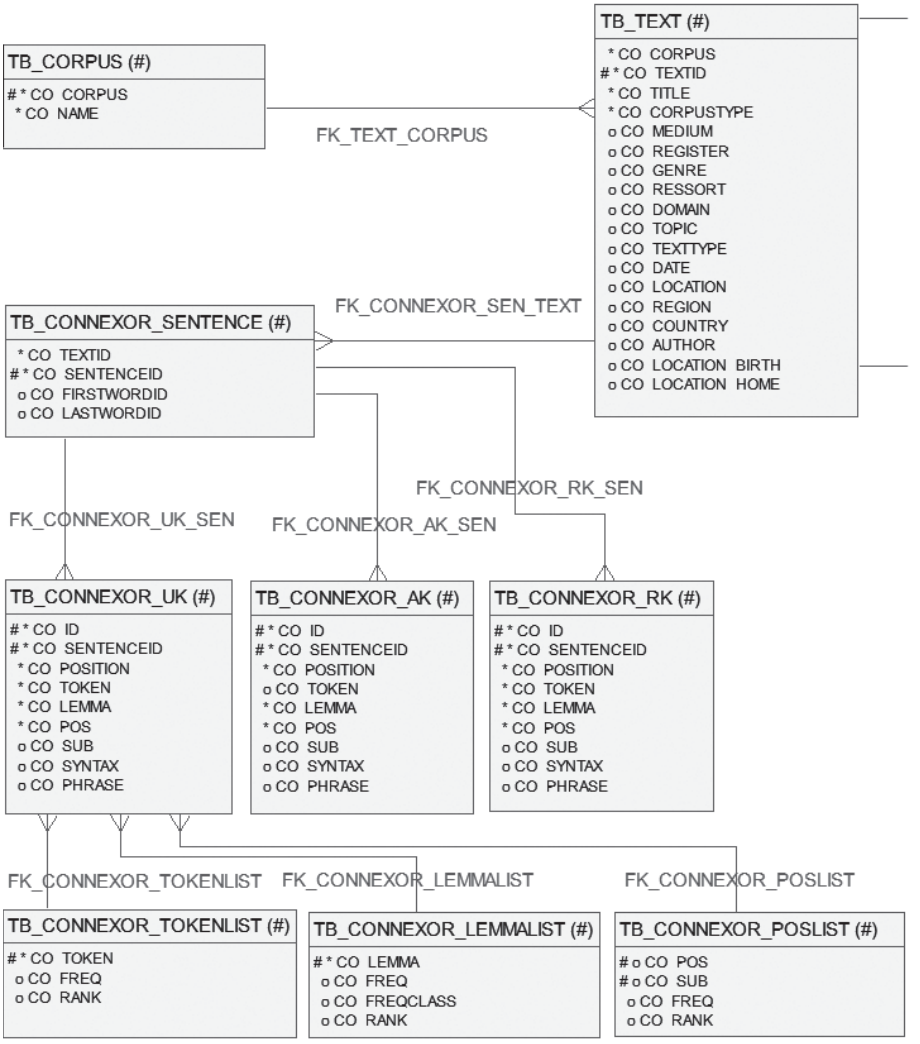


Abb. 11: Physisches Datenschema, Teil 1 (Korpus- und Textebene sowie Connexor-Segmentierung)

Die drei Connexor-Wortformentabellen enthalten jeweils folgende Spalten:

- CO_ID (eindeutige Nummerierung der Tokens)
- CO_SENTENCEID (eindeutige Nummerierungen der Sätze)
- CO_POSITION (numerische Position des Worts im Primärtext)
- CO_TOKEN (Wortform/Textwort)
- CO_LEMMA (Grundform)
- CO_POS (Wortklasse)
- CO_SUB (zusammengeführte optionale Angaben wie beispielsweise Modus, Tempus oder Aspekt von Verben)
- CO_SYNTAX (syntaktische Spezifikation)
- CO_PHRASE (Indikator der Zugehörigkeit zu einer Nominalphrase)

Ergänzend zu den explizit aus dem konzeptuellen Modell ableitbaren Tabellen sind TB_CONNEXOR_TOKENLIST, TB_CONNEXOR_LEMMALIST und TB_CONNEXOR_POSLIST klassische Lookup-Listen mit Frequenzangaben zu den vom Connexor-Parser ermittelten Wortformen/Token, Lemmata/Types bzw. Wortklassen. Sie bilden keine unmittelbar in den Quelldaten vorhandenen Objekte ab, sondern sollen nach erfolgtem Datenimport bzw. -update für das Gesamtkorpus berechnet und dauerhaft vorgehalten werden. Diese Vorgehensweise trägt dem Umstand Rechnung, dass entsprechende Verteilungswerte erfahrungsgemäß eine häufig genutzte Grundlage für spätere statistische Auswertungen sind.[100] In einem Online-Rechercheportal sollten sie z.B. als geordnete Listen sowie mit Hilfe von Suchformularen gezielt abrufbar sein. Neben absoluten Frequenzangaben (CO_FREQ) – oder, im Fall der Lemmata, zusätzlichen Frequenzklassen (CO_FREQCLASS)[101] – werden auch Rangzahlen (CO_RANK) kodiert.

Bezüglich der vom TreeTagger bzw. Xerox-Parser gelieferten Segmentierungen und Annotationen entspricht die physische Umsetzung – dokumentiert in den Abbildungen 12 und 13 – unserem Vorgehen für Connexor-Daten. Auf Wortebene orientieren sich die Tabellenspalten an den jeweiligen Objekt eigenschaften, so dass die Connexor-Angaben Subkategorie (CO_SUB), Syntax

[100] In unserem Anforderungskatalog aus Abschnitt 2.5 betrifft dies z.B. Abfrage 8 mit dem Suchkriterium „hochfrequent (Frequenzklasse < 9)“.

[101] Absolute Frequenzwerte sind für linguistische Untersuchungen häufig wenig sinnvoll, da sie – je nach Zusammensetzung der Datengrundlage – eine nicht zu rechtfertigende Exaktheit suggerieren. Aussagekräftiger erscheinen generalisierende Frequenzklassen, die wir in Anlehnung an die Mannheimer DeReWo-Listen (www.ids-mannheim.de/kl/projekte/methoden/derewo.html) berechnen. Ein Lemma wird dabei in die Frequenzklasse N eingeordnet, wenn es ca. 2^N-mal seltener als das häufigste im Gesamtkorpus enthaltene Lemma vorkommt.

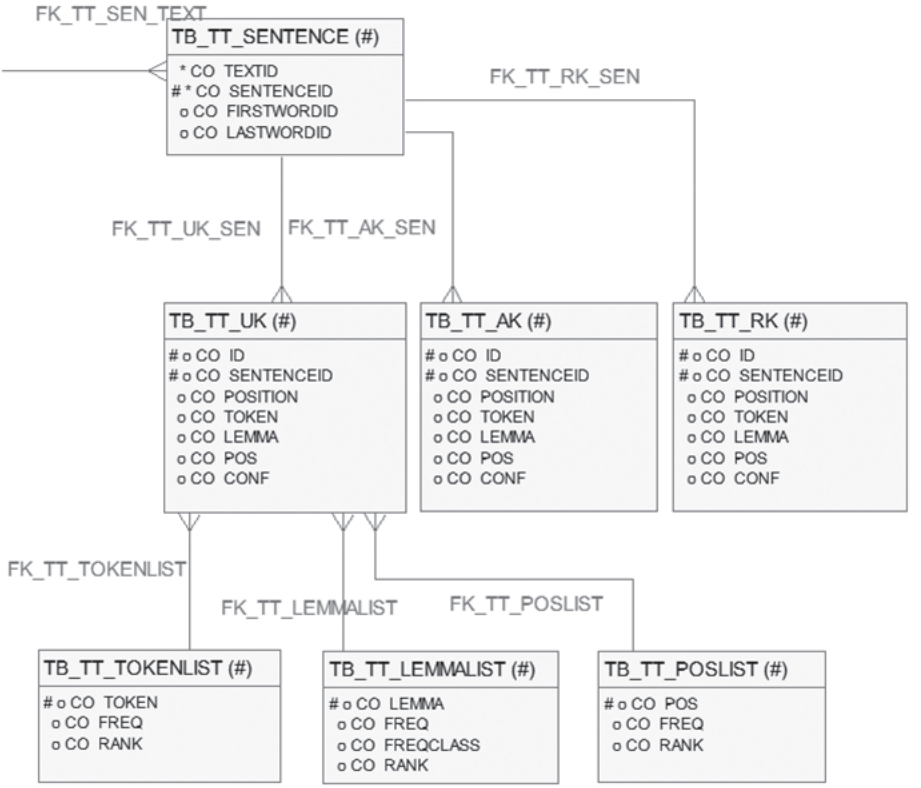


Abb. 12: Physisches Datenschema, Teil 2 (TreeTagger-Segmentierung)

(CO_SYNTAX) und Wortgruppe (CO_PHRASE) entfallen dürfen. Im Gegenzug ergänzen der TreeTagger-Konfidenzwert (CO_CONF) sowie die Xerox-Featureliste (CO_FTS) das Schema.

Ausschließlich für Xerox-Daten stehen die Tabellen TB_XEROX_NODE und TB_XEROX_DEP bereit und nehmen Knotenobjekte (z.B. Mitglieder einer Wortgruppe/Phrase) bzw. Dependenzen zwischen Wörtern eines Satzes (z.B. benannte Verb-Objekt-Beziehungen), referenziert durch Satznummern in CO_SENTENCEID, auf. Beide enthalten Spalten für Phänomenbezeichner (CO_TAG bzw. CO_NAME) sowie die Xerox-Featurelist (CO_FTS). Die Einbettung in übergeordnete Knoten wird in Tabelle TB_XEROX_NODE über die Spalte CO_PARENT realisiert, deren Inhalte auf ID-Schlüssel des Mutterknotens referenzieren. Die Spalte CO_ORDER organisiert mit numerischen Werten die Reihenfolgen nebengeordneter Geschwisterknoten. Dependenzstrukturen mit beidseitig potenziell höheren Kardinalitäten implementieren wir einschränkend und in Übereinstimmung mit einer Vorab-Analyse des

Datenbestands durch 1:n-Beziehungen: Eine Dependenzstruktur besteht jeweils aus maximal drei Wortknoten (Spalten CO_PARAM0, CO_PARAM1, CO_PARAM2 in TB_XEROX_DEP), ein Wortknoten darf Bestandteil beliebig vieler ausgezeichneter Dependenzstrukturen sein.

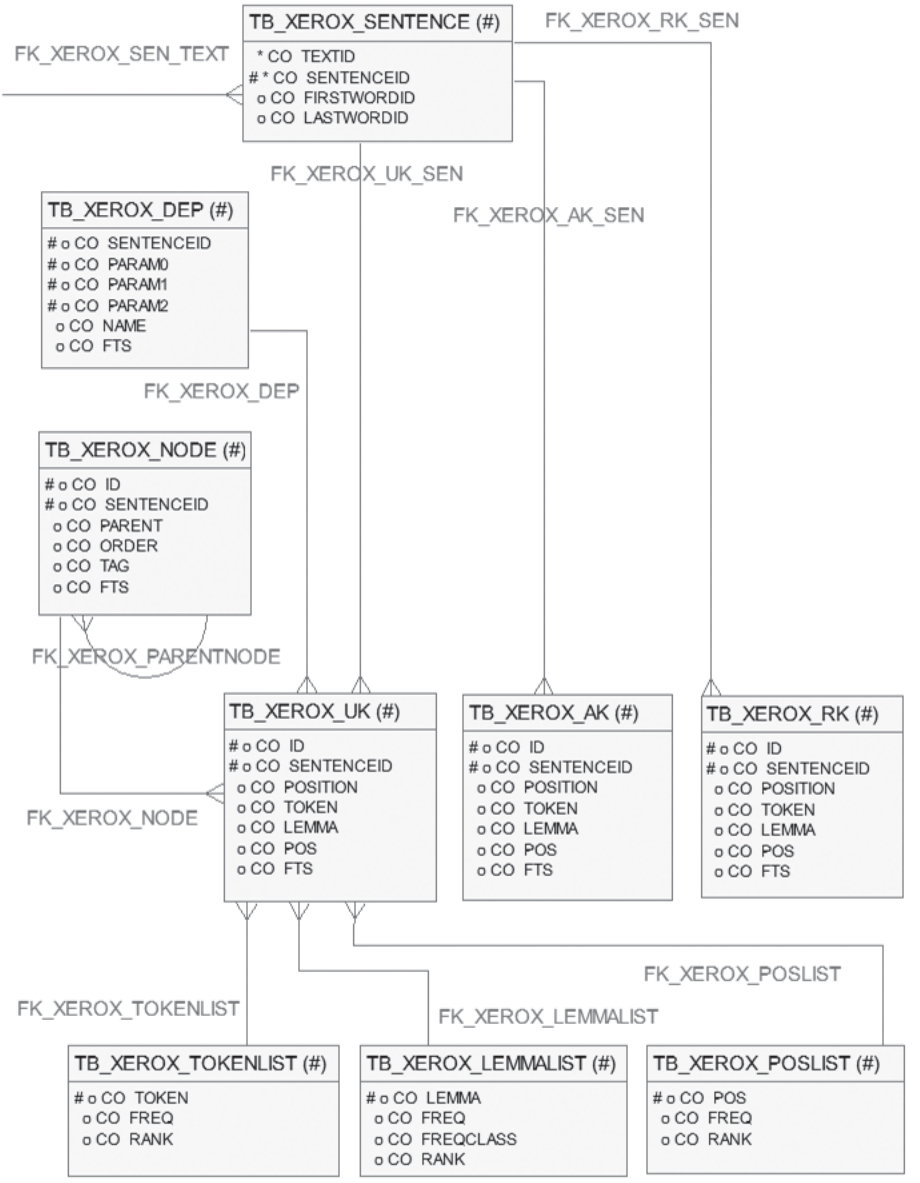


Abb. 13: Physisches Datenschema, Teil 3 (Xerox-Segmentierung)

3.2.4 Hard- und Software

Als Referenzsystem für die Evaluierung der Designentscheidungen dient eine kompakte Midrange-Workstation mit Linux-Betriebssystem (CentOS), deren Arbeits- und Festplattenspeicher an die zu verarbeitenden Datenvolumen angepasst wurden; siehe Tabelle 6. Datenbanktabellen, -indizes und -abfragen wurden für die nachfolgenden Designtests unter Nutzung eines objekt-relationalen Datenbank-Managementsystems (DBMS) implementiert.

Parameter	Referenzsystem
CPU	1 Quadcore-Prozessor Intel i5 mit 2,67GHz Taktung (ca. 5.300 bogomips)
RAM	16 GB
Festplatten	4 SATA-HDD mit insgesamt 6 TB Volumen (RAID Level 0) und ext3-Dateisystem

Tab. 6: Hardware-Parameter des Referenzsystems

Parameter	Wert
Blockgröße	8.192 Bytes
Cachegröße	256 MB
Maximale Speichernutzung	6 GB
Cursor-Maximalzahl	300

Tab. 7: Datenbank-Parameter (Auszug)

3.2.5 Datenimport

Der Datenimport überführt die in Abschnitt 3.2.1 dokumentierten Primär- und Sekundärdaten in unsere relationale Datenstruktur, die Dekomposition von XML-Strukturen wird in diesem Zusammenhang auch „shredding“ genannt. Während des Imports sollen sämtliche Ausgangsinformationen kumuliert werden, die für linguistisch motivierte Abfragen wertvoll sein könnten. Gleichzeitig müssen die Importstrukturen derart angelegt sein, dass sie etwaige Nachführungen und Ergänzungen ermöglichen. Der Import in den Evaluations-Prototyp erfolgt skriptbasiert durch im DBMS hinterlegte Prozedu-

ren.[102] Dabei werden die XML-kodierten Quelldateien linear eingelesen, der Inhalt in temporären Speicherstrukturen gesammelt und durch BULK INSERT-Anweisungen in die Zieltabellen geschrieben. Für Einzelwörter und Sätze werden eindeutige Schlüsselwerte automatisiert vergeben (CO_ID bzw. CO_SENTENCEID).

Der Importalgorithmus operiert auf verteiltem XML-Datenmaterial (Korpustexte sowie Tagger-Output) und relationiert neben korpus-, text- und wortspezifischen Ausgangsdaten auch syntaktische Strukturen wie die Xerox-Knotenhierarchien[103] und -Dependenzen. Zur exemplarischen Illustration der Vorgehensweise kann der Umgang mit der vergleichsweise flachen Tree-Tagger-Annotation herangezogen werden:

```
<sentence>
  <lexeme id="955" pos="4592" len="2">
    <surface-form>Im</surface-form>
    <sense id="0">
       <base-form>im</base-form>
       <part-of-speech conf="1.000000">APPRART</part-of-speech>
    </sense>
  </lexeme>
  <lexeme id="956" pos="4595" len="9">
    <surface-form>Gegenteil</surface-form>
    <sense id="0">
      <base-form>Gegenteil</base-form>
      <part-of-speech conf="1.000000">NN</part-of-speech>
    </sense>
  </lexeme>
…
</sentence>
```

Findet die Einleseprozedur in der vorstehenden Standoff-Annotation das XML-Element „sentence“, wird ein eindeutiger numerischer Satzschlüssel generiert. Ein „lexeme“-Element impliziert einen neuen Wortschlüssel, die vom Tagger

[102] Solche Prozeduren und Funktionen im DBMS eignen sich für die persistente serverseitige Ausführung wiederkehrender Aufgaben. Sie können bei Bedarf nativ mit XML-Daten umgehen – wichtig für das Import-Shredding – und bilden die Programmierbasis unseres Referenzsystems sowie des in Kapitel 6 präsentierten Online-Prototyps; zur Syntax vgl. Bryla/Loney (2013, S. 545 ff.).

[103] Hier gilt es beispielsweise zu beachten, dass der Schlüsselwert eines Xerox-Tokens stets genau einer Knoten-ID entspricht. Reine Wortknoten wandern nicht in die Knotentabelle, d.h. TB_XEROX_NODE enthält als Primärschlüssel nur IDs, die nicht für Token genutzt werden.

vergebenen Attribute „id“ und „len“ werden ignoriert. Der Inhalt des Elements „surface-form“ wandert als Tokenwert in die Spalte CO_TOKEN der entsprechenden Tabelle, der Inhalt des „lexeme“-Attributs „pos“ in die Positionsspalte CO_POSITION. Lemmata für die Spalte CO_LEMMA werden dem XML-Element „base-form“ entnommen, Wortklassenangaben für CO_POS dem XML-Element „part-of-speech“. Liegen für ein Lexem mehrere Lesarten („sense“) vor, wählt das Skript die Variante mit dem besten Konfidenzwert (Attribut „conf“) aus.

Der generierte Satzschlüssel dient in TB_TT_SENTENCE zur Verknüpfung mit den in der zentralen Tabelle TB_TEXT abgelegten textspezifischen Metadaten. Um diese initial zu bestimmen, ermittelt das Importskript in den Quelldateien, ausgehend von der Position des ersten Satzworts, zunächst das zugehörige „textSigle“-Element und speichert dessen Inhalt in der Spalte CO_TEXTID ab. Anschließend ergänzt es weitere im Primärtext enthaltene außersprachliche Metadaten (z.B. den Autorennamen für CO_AUTHOR, das Publikationsdatum für CO_DATE). Bedingt durch die im IDS-Textmodell definierte Trennung zwischen Text-, Dokument- und Korpusebene können deskriptive Metadaten an variierenden Stellen in der XML-Hierarchie erscheinen, was Konsequenzen für die Algorithmisierung des Imports hat: Findet das Skript beispielsweise im der Textsigle folgenden Autorenelement keinen Inhalt, so überprüft es sukzessive zunächst den zum übergeordneten Dokument gehörigen Metadatenblock. Sofern auch dort nichts Passendes steht, analysiert es den Korpus-Metadatenblock. Nicht in den Quelldateien vorhandene Angaben, beispielsweise zu Medium, Texttyp oder thematischer Domäne, werden aus der DeReKo-Metadatenbank automatisiert nachgeführt.

Semi-automatische bzw. manuelle Nachführungen finden an mehreren Stellen statt. Beispielsweise zeigt eine Inspektion der Quelldaten, dass bei Folgeauflagen literarischer Werke anstelle des Ersterscheinungsdatums zumeist lediglich das Publikationsdatum der spezifischen Auflage angegeben ist. Das für sprachwissenschaftliche Untersuchungen relevante Jahr der Textproduktion muss in diesen Fällen manuell recherchiert und eingetragen werden. Ähnliches gilt für andere Metadatentypen, etwa für die Geburts- und Wohnorte einzelner Textautoren (Spalten CO_LOCATION_BIRTH und CO_LOCATION_HOME in TB_TEXT).

	Untersuchungskorpus	Ausgewogenes Korpus	Reservekorpus
Texte	25.428.705	20.148	181.419
Connexor-Sätze	486.962.776	1.165.198	10.590.754
Connexor-Token	7.905.678.207	20.026.757	180.042.825
TreeTagger-Sätze	367.218.820	815.029	7.443.621
TreeTagger-Token	7.486.587.155	18.517.447	156.316.331
Xerox-Sätze	254.214.706	1.125.925	0
Xerox-Token	4.218.253.174	19.281.024	0

Tab. 8: Text-, Satz- und Wortvolumen des Referenzsystems

Tabelle 8 dokumentiert das im Referenzsystem nach Abschluss des Imports abgebildete Korpusvolumen. Dabei gilt es zu beachten, dass nur etwas mehr als die Hälfte aller Quelltexte auch mit dem Xerox-Tagger vorverarbeitet wurden. Folgerichtig fallen die entsprechenden Satz- und Tokenzahlen deutlich niedriger aus.

	Name	Sätze Connexor	Sätze TreeTagger	Sätze Xerox
Domäne	unklassifizierbar	22.082.564	15.613.695	17.114.568
Domäne	Fiktion	648.280	485.210	578.810
Domäne	Kultur	201.331.311	145.584.166	106.142.123
Domäne	Mensch	9.270.648	7.024.839	6.224.250
Domäne	Politik	216.095.592	170.704.513	106.777.114
Domäne	Technik	37.534.381	27.806.397	17.377.841
gesamt		**486.962.776**	**367.218.820**	**254.214.706**
Jahr	2000-09	184.167.499	146.561.082	143.771.203
Jahr	2010-	203.709.132	138.147.366	14.838.647
Jahr	-1969	507.727	364.581	176.955
Jahr	1970-79	108.199	82.845	107.872
Jahr	1980-89	159.911	117.882	159.061
Jahr	1990-99	98.310.308	81.945.064	95.160.968
gesamt		**486.962.776**	**367.218.820**	**254.214.706**

	Name	Sätze Connexor	Sätze TreeTagger	Sätze Xerox
Land	D	396.519.736	293.087.801	181.611.884
Land	D-Ost	161.276	124.490	164.923
Land	D-West	1.819.963	1.509.405	1.824.078
Land	A	52.413.904	41.953.417	46.539.535
Land	CH	32.935.801	27.871.982	22.609.214
Land	LU	1.658.989	1.412.927	0
gesamt		**485.509.669**	**365.960.022**	**252.749.634**
Medium	Publikumspresse	380.105.074	307.209.938	249.446.482
Medium	Bücher	1.362.887	1.040.472	540.917
Medium	Internet	76.306.651	39.046.190	3.242.183
Medium	Gesprochenes	25.306.603	16.888.829	271.085
Medium	Sonstiges	712.021	614.543	714.039
gesamt		**483.793.236**	**364.799.972**	**254.214.706**
Region	überregional	59.123.219	51.640.704	61.274.256
Region	Herkunft unbek.	436.207	326.381	359.675
Region	nicht zuordenbar	76.318.061	39.055.324	3.247.965
Region	Südwest	35.079.658	28.969.509	22.527.056
Region	Mittelost	3.715.278	2.775.915	102.988
Region	Mittelsüd	22.510.752	18.285.794	15.544.403
Region	Mittelwest	97.494.028	72.365.881	64.812.088
Region	Nordost	55.161.454	45.610.755	15.948.838
Region	Nordwest	37.298.978	28.538.281	12.124.821
Region	Südost	82.962.260	65.582.273	41.709.421
gesamt		**470.099.895**	**353.150.817**	**237.651.511**
Register	Presse	374.197.655	302.544.045	250.043.178
Register	Gebrauch	111.643.000	63.799.150	3.720.418
Register	Literarisch	1.122.034	875.569	451.110
gesamt		**486.962.689**	**367.218.764**	**254.214.706**

Tab. 9: Stratifikation des Untersuchungskorpus (UK) nach Metadaten

Die Tabellen 9 und 10 differenzieren die Verteilung ausgewählter, für linguistisch motivierte Untersuchungen als Variabilitätsfaktoren relevanter Metadaten über das Untersuchungskorpus (UK) und das ausgewogene Korpus (AK). Als Vergleichsgröße ist in der Darstellung die Satzanzahl – jeweils auf Grundlage der unterschiedlichen Segmentierungsergebnisse von Connexor, TreeTagger und Xerox – gewählt. Bei der Interpretation gilt es zu beachten, dass für die Komposition des ausgewogenen Korpus die tatsächliche Anzahl der Wortformen (Token) als Bezugsrahmen zum Einsatz kam.

	Name	**Sätze Connexor**	**Sätze TreeTagger**	**Sätze Xerox**
Domäne	unklassifizierbar	303.604	212.516	297.733
Domäne	Fiktion	177.772	132.098	184.455
Domäne	Kultur	254.487	152.591	224.867
Domäne	Mensch	15.140	11.313	14.488
Domäne	Politik	304.535	233.015	303.249
Domäne	Technik	109.660	73.496	101.133
gesamt		**1.165.198**	**815.029**	**1.125.925**
Jahr	2000-09	585.408	372.337	551.398
Jahr	2010-	0	0	0
Jahr	-1969	135.545	98.598	115.650
Jahr	1970-79	74.316	59.295	74.391
Jahr	1980-89	115.745	84.952	117.476
Jahr	1990-99	254.184	199.847	267.010
gesamt		**1.165.198**	**815.029**	**1.125.925**
Land	D	686.082	442.755	655.117
Land	D-Ost	127.697	96.324	131.462
Land	D-West	278.854	215.709	262.247
Land	A	39.499	32.470	42.779
Land	CH	32.657	27.439	33.908
Land	LU	0	0	0
gesamt		**1.164.789**	**814.697**	**1.125.513**
Medium	Publikumspresse	223.440	191.663	234.744

	Name	Sätze Connexor	Sätze TreeTagger	Sätze Xerox
Medium	Bücher	224.725	174.062	230.932
Medium	Internet	266.743	122.743	212.213
Medium	Gesprochenes	231.950	171.605	245.842
Medium	Sonstiges	218.340	154.956	202.194
gesamt		**1.165.198**	**815.029**	**1.125.925**
Region	überregional	112.580	94.135	115.406
Region	Herkunft unbek.	247.980	188.814	239.261
Region	nicht zuordenbar	269.139	124.340	212.458
Region	Südwest	69.620	56.424	72.625
Region	Mittelost	73.507	51.690	80.157
Region	Mittelsüd	48.605	38.461	50.651
Region	Mittelwest	82.245	61.783	86.856
Region	Nordost	117.038	86.531	118.561
Region	Nordwest	104.788	81.778	108.113
Region	Südost	39.696	31.073	41.837
gesamt		**1.165.198**	**815.029**	**1.125.925**
Register	Presse	291.657	245.871	294.503
Register	Gebrauch	691.444	435.481	647.414
Register	Literarisch	182.097	133.677	184.008
gesamt		**1.165.198**	**815.029**	**1.125.925**

Tab. 10: Stratifikation des ausgewogenen Korpus (AK) nach Metadaten

Für eine Beurteilung des Retrievalaufwands bei der Recherche nach speziellen Wortklassenausprägungen spielt die konkrete Verteilung der Wortklassen eine zentrale Rolle. Die nachfolgenden Tabellen dokumentieren diese Zahlen für die drei erfassten Tagging-Werkzeuge.

Wortklasse	Anzahl
A	501.956.255
ADV	534.336.883
CC	196.380.018
CS	82.538.036
DET	826.871.425
INTERJ	796.762
N	2.203.142.955
NUM	268.378.679
PREP	711.083.172
PRON	525.552.895
V	886.321.944
Satzzeichen	1.168.319.183

Tab. 11: Verteilung der Wortklassen (Connexor) im Referenzsystem

Wortklasse	Zusatz	Anzahl
A	CMP	17.616.842
A	SUP	13.736.266
N	Abr	7.234.440
N	Abbr PL	66.635
N	PL	367.641.151
N	Prop	283.908.872
N	Prop PL	1.389.700
NUM	ORD	38.061.926
V	IMP	4.861.462
V	IND	551.554.601
V	IND PAST	184.535.879
V	IND PRES	367.018.722
V	INF	119.485.854
V	PCP	150.618.260
V	PCP PERF	145.672.842

Wortklasse	Zusatz	Anzahl
V	PCP PROG	4.945.418
V	SUB PAST	20.178.145
V	SUB PRES	34.757.702

Tab. 12: Verteilung der dokumentierten Wortklassen-Zusatzangaben (Connexor) im Referenzsystem

In der Übersicht der Connexor-Zusatzangaben fehlen Kombinationen, die in der Tagger-Dokumentation nicht erwähnt sind, in den Annotationsdateien aber trotzdem auftreten. Ein Beispiel hierfür ist die – inhaltlich nicht plausible, aber mitunter tatsächlich gelieferte – Kombination „Verb im Infinitiv und Plural" (V INF PL). Die Überführung der XML-Annotationen in Datenbanktabellen ermöglicht damit als Nebeneffekt nachgelagerte Untersuchungen zur Plausibilität automatisch generierter Annotationen.

Wortklasse	Anzahl
$(	289.772.116
$,	373.049.803
$.	453.766.796
ADJA	388.189.541
ADJD	168.947.132
ADV	343.991.861
APPO	863.025
APPR	564.019.762
APPRART	135.439.540
APZR	617.197
ART	724.618.442
CARD	187.679.592
FM	12.659.197
ITJ	701.374
KOKOM	37.800.181
KON	190.075.933
KOUI	7.159.834

Wortklasse	Anzahl
KOUS	64.992.899
NE	534.686.327
NN	1.525.917.467
PAV	188.318
PDAT	27.906.835
PDS	21.339.062
PIAT	54.752.407
PIS	34.023.260
PPER	149.235.148
PPOSAT	57.013.085
PPOSS	12.347
PRELAT	1.218.200
PRELS	46.072.433
PRF	45.838.895
PROAV	35.732.704
PTKA	1.690.057
PTKANT	1.550.256
PTKNEG	40.747.194
PTKVZ	41.463.706
PTKZU	30.449.305
PWAT	660.045
PWAV	6.653.763
PWS	6.068.281
TRUNC	11.227.309
VAFIN	216.511.206
VAIMP	239.493
VAINF	21.648.906
VAPP	7.762.726
VMFIN	64.347.008
VMINF	4.582.559

Wortklasse	Anzahl
VMPP	103.209
VVFIN	288.399.851
VVIMP	1.550.299
VVINF	112.834.003
VVIZU	6.902.869
VVPP	138.265.520
XY	4.648.877

Tab. 13: Verteilung der Wortklassen (TreeTagger) im Referenzsystem

Wortklasse	Anzahl
ADJ	326.691.132
ADV	222.501.943
CONJ	161.704.824
DET	482.814.818
ITJ	301.357
NEGAT	22.702.800
NOUN	1.114.450.409
NUM	117.282.065
POSTP	3.974.712
PREP	409.634.741
PRON	178.457.093
PTCL	46.462.193
PUNCT	619.141.647
SYMBOL	15.227.463
TRUNC	7.696.207
VERB	489.209.770

Tab. 14: Verteilung der Wortklassen (Xerox) im Referenzsystem

3.3 Evaluierung einzelner Designentscheidungen

Gegenstand der nachfolgenden Evaluierungen sind Lösungsvarianten für grundlegende Aspekte des DBMS-basierten Korpusretrievals. Die Zielsetzung besteht dabei nicht im systemübergreifenden Benchmarking oder im Vergleich mit existierenden Produktivsystemen. Vielmehr sollen, ausgehend von der bereits diskutierten grundsätzlichen Eignung relationaler Technologien, ausgewählte Speicher- und Abfragevarianten gegenübergestellt und empirisch bewertet werden. Eingeführte Modelle und Strategien dienen dabei als Ausgangspunkte, die unter Berücksichtigung spezieller Eigenheiten von Sprachressourcen – z.B. Wortlängen oder Wortverteilungen – erprobt werden sollen. Generell gilt es, von den ermittelten absoluten Werten auf relative Performanzvor- und -nachteile der jeweiligen Variante zu abstrahieren. Das Vorgehen orientiert sich in gewisser Weise an der Erstellung geografischer Karten: Zeichnet man diese mit allen erdenklichen Details im Maßstab 1:1, ist die Abstraktionsleistung gleich null, d.h. die Karten helfen nicht bei der Durchdringung komplexer Gegebenheiten. Erst durch eine Reduzierung der zu berücksichtigenden Parameter – für Korpusdatenbanken also der als relevant erachteten Speicher- bzw. Abfrageparameter – ist eine Abstraktion möglich.

Performanz bedeutet im vorliegenden Kontext eine Minimierung der Antwortzeiten für Sprachkorpora variablen Umfangs. Absolute Zahlen – im vorliegenden Fall die gemessenen Sekunden, Minuten oder gar Stunden – helfen, sofern unter vergleichbaren Bedingungen ermittelt, bei der realistischen Einordnung der zugehörigen Implementierung. Bislang existieren für linguistisch motivierte Analysen häufig nur ungefähre Vorstellungen und Schätzungen zur Adäquatheit einzelner Implementierungsvarianten. Diesbezügliche Auswertungen konzentrieren sich oft auf spezialisierte Korpustypen und Fragestellungen, oder aber die verwendeten Volumina entsprechen nicht den quantitativen Anforderungen moderner Korpusgenerationen.[104] Unsere Evaluierung soll dagegen datenbankgestützte Strategien für umfangreiche Korpusgrößen empirisch analysieren und die vielversprechendsten Modelle in Kapitel 4 anhand des systematischen Referenzkatalogs auf prototypische linguistisch motivierte Abfrageszenarios anwenden.

Um dies leisten zu können, ist authentisches Sprachmaterial in angemessener Größenordnung notwendig; vgl. hierzu Kapitel 2. Unsere Untersuchungen fin-

[104] Beispielsweise legt Evert (2010) den Fokus auf Konkordanz-Recherchen innerhalb der webbasierten, nicht linguistisch annotierten Google n-Gramme (Web1T5) mit einer Billion Token. Davies (2005) recherchiert im Corpus del Español, das ca. 100 Millionen Token umfasst. Wesentlich kleiner sind die Korpusvolumen von Bindernagel (2007) (100.000 Token) oder Zhang et al. (2001) (drei Datensets mit ca. 470.000, 3,6 Millionen bzw. 20 Millionen Token).

den durchgehend auf realen Sprachdaten statt, nicht auf abstrakt konstruierten Datensets. Für die nachfolgenden Evaluierungsläufe wurde aus dem oben beschriebenen Gesamtkorpus durch Zufallsauswahl (*random sample*) ein Subkorpus mit 1 Milliarde (10^9) Textwörtern erstellt; dies entspricht der Größenordnung von Korpora der sogenannten dritten Generation; vgl. Kapitel 1.

Generell sind die durchgeführten Testläufe so aufgebaut, dass die Korrelation zweier Variablen – also etwa Indexspalten und Antwortzeit – präzise dokumentiert werden kann. Es wurde versucht, den Einfluss weiterer potenziell relevanter Variablen in separaten Schritten zu untersuchen, um Fehlinterpretationen auszuschließen. Naheliegende Beispiele hierfür sind Korpusgröße oder Wortlänge bzw. -frequenz: Indexabfragen weisen bei Wörtern variabler Länge (die wiederum mit der Frequenz korreliert) unterschiedliche Performanzwerte auf. Ignoriert man diesen Umstand, zeigen Laufzeitanalysen ggf. ein ganz anderes Bild, das dann für konkrete Schlussfolgerungen hinsichtlich der Praxistauglichkeit einer Variante nicht mehr aussagekräftig sein kann. Ähnliches gilt für Caching-Effekte: Berücksichtigt eine Auswertung nicht die Unterschiede zwischen Abfragen mit oder ohne Cache-Nutzung, bleiben der Nutzen selbst gemittelter Antwortzeiten fragwürdig.

3.3.1 Datenmodell

Antwortzeiten DBMS-basierter Recherchen werden maßgeblich vom zugrunde liegenden physischen Datenmodell beeinflusst. Bekanntermaßen ist die Modellierung eines Weltausschnitts, also die formale Abbildung der für eine Fragestellung relevanten Datenstrukturen, häufig auf unterschiedliche Weise möglich. Gerade bei der Beschreibung multidimensionaler und damit multifunktionaler Ausgangsdaten bewegt sich der Designer stets in einem Kontinuum konkurrierender Präferenzen. Dies gilt nicht zuletzt für Korpusinhalte, die ja ihrerseits auf einem bestimmten logischen Textmodell beruhen. Prototypisch hierfür steht der Umgang mit linear fortlaufenden Textwörtern: Sollen diese, weil erfahrungsgemäß häufig nach unmittelbar aufeinander folgenden Sequenzen (Kollokationen) gefahndet wird, als Wortfolgen (n-Gramme) modelliert werden? Oder ist eine strikte Einzelwort-Segmentierung vorzuziehen, um z.B. auch diskontinuierliche Strukturen adäquat abfragbar zu machen?

Vor diesem Hintergrund lassen sich zwei konkurrierende Modelle ausmachen: N-Gramm-Tabellen einerseits sowie Type-/Token-Relationierung kombiniert mit Tabellenverknüpfungen (*self inner joins*) andererseits. Wir suchen für beide Ansätze Antworten auf folgende Fragen:

– Existieren grundlegende modellbedingte Abfragebeschränkungen?
– Welche Konsequenzen ergeben sich für das Gesamtdatenvolumen?
– Lassen sich signifikante Unterschiede hinsichtlich der Abfrageperformanz beobachten?

3.3.1.1 N-Gramm-Tabellen

N-Gramme erfreuen sich in computerlinguistischen Anwendungsszenarien hoher Popularität, etwa zur Beurteilung der „Natürlichkeit" automatisch generierter Übersetzungen, zur Vorhersage potenzieller Folgewörter oder zur kontextbasierten Disambiguierung von Mehrdeutigkeiten. Davies (2005) stellt am Beispiel des „Corpus del Español" ein n-Gramm-Modell für die datenbankbasierte Verwaltung eines linguistischen Korpus mit ca. 100 Millionen Textwörtern sowie deren Lemmata und Wortklassenbestimmungen vor. Der Leitgedanke besteht im Aufbrechen der linearen Textabfolge durch die Segmentierung in n-Tupel (geordnete Paare, Tripel, Quadrupel etc.), die anschließend in sogenannten n-Gramm-Tabellen abgelegt werden. Die SQL-Recherche erfolgt ausschließlich auf diesen Fragmenten bzw. den hierauf angelegten Indizes und vermeidet dadurch eine Traversierung komplexer relationaler Strukturen. Es wird also darauf verzichtet, Daten aus unterschiedlichen Tabellen oder Zeilen während einer Abfrage zu verknüpfen, was tendenziell rechenintensiver ausfiele.

Spaltenname	Datentyp
SID	NUMBER(12)
WORD1	CHAR(1000)
WORD2	CHAR(1000)
WORD3	CHAR(1000)

Tab. 15: Physikalischer Aufbau der n-Gramm-Tabelle (NGRAM3)

Auf Basis unseres Subkorpus mit einer Milliarde fortlaufenden Textwörtern erstellen wir eine n-Gramm-Tabelle für Recherchen nach Mono-, Bi- und Trigrammen. Tabelle 15 illustriert die dazu gehörige Tabellenarchitektur auf dem Referenzsystem: NGRAM3 speichert jeweils drei aufeinander folgende Textwörter (WORD1, WORD2, WORD3) in einer gemeinsamen Tabellenzeile, die durch eine satzspezifische Referenz (SID) komplettiert wird. Der Speicherbedarf für die enthaltenen 1 Milliarde Trigramme liegt bei 29,9 GB, verteilt auf 955.938 Datenblöcke (zur Blockgröße vgl. Tab. 7). Denkbar wäre eine Anrei-

cherung um weitere linguistisch relevante Attribute wie Wortklasse oder Frequenz, was allerdings eine massive Erhöhung der Spaltenzahl auf sieben bzw. zehn zur Folge hätte. Jedes tokenspezifische Zusatzattribut erhöht die Spaltenzahl gemäß der Formel: Spaltenzahl n-Gramm-Tabelle = n*Attributzahl + 1. Eine Tetragramm-Tabelle mit Wortklassen- und Frequenzangaben umfasst demnach bereits 13, eine Pentagramm-Tabelle 16 Spalten usw. Für unsere Messungen beschränken wir uns auf die Abfrage von maximal drei aufeinander folgenden Textwörtern ohne Metadaten, um die Speicherplatzanforderungen in Grenzen zu halten. Damit einher geht eine Einschränkung der maximal abfragbaren Wortabstände: Recherchen nach linguistischen Phänomenen, die mehr als drei unmittelbar aufeinander folgende Wörter umfassen, lassen sich ebenso wenig durchführen wie solche nach zwei Wörtern mit einem Wortabstand größer als eins.

Zentrale Evaluationsfragen beziehen sich auf die optimale Indizierung einer n-Gramm-Tabelle: Verringern maßgeschneiderte Indizes Abfragezeiten für Wortfolgen signifikant im Vergleich mit Suchoperationen, die ausschließlich auf Tabellenzeilen operieren? Und wenn ja: Welche Spalten/Spaltenkombinationen (*composite indexes*) liefern dabei die besten Ergebnisse? Entscheiden allein Selektivität, also die Effektivität bei der Einschränkung der relevanten Datenmenge, und Dichte, d.h. Anzahl der Duplikate in den Indexstrukturen, – oder sollten ergänzende Spalten in einen Index aufgenommen werden, um Rückgriffe auf die eigentliche Tabelle zu minimieren? Für die Abschätzung des in der Datenbank erforderlichen Speichervolumens ist darüber hinaus auch eine Messung der physischen Indexgrößen aussagekräftig. Zu diesem Zweck legen wir die in Tabelle 16 aufgeführten Indizes an. Der Index „W1" indiziert ausschließlich die erste Wortspalte, Index „W1S" nimmt zusätzlich die zugehörige Satznummer auf etc.

Index	Indizierte Spalten	Größe in MB	Clustering Factor	Index-tiefe	Blatt-knoten
W1	WORD1	18.318	553.702.473	2	592.634
W1S	WORD1,SID	24.701	855.154.146	2	794.143
W1W2	WORD1,WORD2	24.445	877.042.965	2	785.007
W1W2S	WORD1,WORD2,SID	30.873	976.563.824	2	992.849
W1W2W3	WORD1,WORD2,WORD3	30.246	923.649.038	2	959.290
W1W2W3S	WORD1,WORD2,WORD3,SID	36.630	941.968.458	2	1.129.684

Tab. 16: Physischer Aufbau der Indizes der n-Gramm-Tabelle

Ein Blick auf die Indexgrößen zeigt das erwartete Bild. Je mehr Spalten in einen Index einfließen, desto mehr Speichervolumen wird benötigt (minimal 18,3 GB bis maximal 36,6 GB bei gemeinsamer Indizierung aller drei Wortspalten sowie der Schlüsselspalte). Das ist, bei aller Trivialität, insofern erwähnenswert, als für Abfragen nach ein, zwei oder drei aufeinander folgenden Wörtern vermutlich – im Vorgriff auf die nachfolgenden Ergebnisse – mehrere Indizes parallel vorgehalten werden müssen. Selbst in unserem vergleichsweise einfachen, weil rein tokenbasierten Retrievalsetting führt dieses Vorgehen bereits zu einer grundsätzlich unwillkommenen Vervielfachung des Speicherbedarfs.

Tabellenindizes bauen für die effektive Inhaltsabfrage eine balancierte logische Baumstruktur (*B-Tree*) auf, die nach Schlüsselwerten sortiert und unabhängig von der physischen Abfolge in der Tabelle ist. Reihenfolgen zwischen einzelnen Einträgen werden durch doppelt verkettete Listen kodiert, wobei jeder Listeneintrag sowohl eine Referenz auf den vorangehenden wie auf den nachfolgenden Eintrag enthält. Blattknoten (*Leaf Blocks*) genannte Einheiten nehmen die Indexeinträge physisch auf und entsprechen damit den Datenblöcken einer Tabelle (*Data Blocks*). Neben der Anzahl der Blattknoten und der Indextiefe[105] bietet ein Vergleich der „Clustering Factor"-Werte grundsätzliche Hinweise auf die Effektivität der angelegten Indizes. Grob gesagt repräsentiert der Clustering Factor die Anzahl der logischen I/O-Operationen, die für den Tabellenzugriff unter Nutzung eines Index erforderlich sind.[106] Damit ist er ein Maß für die „Ordnungsstärke" eines Index in Relation zur zugrunde liegenden Tabelle. Liegt der Wert nahe an der Anzahl der Tabellenblöcke (hier: 955.938), dann verweisen die Indexeinträge eines Blattknotens tendenziell eher zu Zeilen, die in einem gemeinsamen Datenblock liegen. Daraus resultieren ebenso tendenziell kürzere Zugriffszeiten. Liegt der Clustering Factor nahe an der Anzahl der Tabellenzeilen, dann ist die Werteverteilung im Index disparater; bei eindeutigen Schlüsselindizes (*Primary Keys*) entspricht der Clustering Factor in etwa der Zeilenanzahl. Für Indexsuchen mit potenziell vielen Treffern (*Index Range Scans*) liegt deshalb die Vermutung nahe, dass Indizes mit hohem Clustering Factor längere Abfragezeiten befördern – jedenfalls sofern Rückgriffe auf die Tabelle erforderlich und nicht sämtliche abzufragenden Attribute im Index enthalten sind.

[105] Die Indextiefe bezeichnet die Anzahl der Stufen in einer Indexstruktur. Grundsätzlich ermöglichen dabei kleine Werte kürzere Zugriffszeiten.

[106] Vgl. Kyte (2010, S. 449): „We could also view the clustering factor as a number that represents the number of logical i/o's against the table, that would be performed to read the entire table via the index".

Um zu belastbaren Ergebnissen zu kommen, formulieren wir separate SQL-Abfragen nach Frequenzen für Monogramme (d.h. Recherchen auf der Spalte WORD1), Bigramme (Recherchen auf WORD1 und WORD2) und Trigramme (Recherchen auf WORD1, WORD2 und WORD3); siehe Tabelle 17. Dabei beziehen wir mit der Aggregatfunktion `count(sid)` bewusst den Satzschlüssel ein, obwohl ein `count(*)` oder `count(word1)` zumeist kürzere Abfragezeiten liefern dürfte. Der Grund hierfür liegt in dem Umstand, dass typische Korpusrecherchen eben nicht nur kontextfreie Frequenzen oder Wortbelege liefern sollen, sondern auch Metadaten wie Korpus- oder Textname, Publikationsjahr usw. Um diese Metadaten – oder auch längere Satzkontexte – in einem Folgeschritt ermitteln zu können, ist die Ermittlung eines entsprechenden Satzschlüssels unabdingbar; unsere Abfragen orientieren sich in diesem Sinne an der realistischen Recherchepraxis.

Abfragetyp	Indexnutzung	SQL
Monogramm <WORT1>	Ohne Index	`select /*+ NO_INDEX(NGRAM3) */ count (sid) from ngram3 where word1=<WORT1>;`
Monogramm <WORT1>	Index <INDEXNAME>	`select /*+ INDEX (NGRAM3 <INDEXNAME>)*/ count (sid) from ngram3 where word1=<WORT1>;`
Bigramm <WORT1> <WORT2>	Ohne Index	`select /*+ NO_INDEX(NGRAM3) */ count (sid) from ngram3 where word1=<WORT1> and word2=<WORT2>;`
Bigramm <WORT1> <WORT2>	Index <INDEXNAME>	`select /*+ INDEX (NGRAM3 <INDEXNAME>)*/ count (sid) from ngram3 where word1=<WORT1> and word2=<WORT2>;`
Trigramm <WORT1> <WORT2> <WORT3>	Ohne Index	`select /*+ NO_INDEX(NGRAM3) */ count (sid) from ngram3 where word1=<WORT1> and word2=<WORT2> and word3=<WORT3>;`
Trigramm <WORT1> <WORT2> <WORT3>	Index <INDEXNAME>	`select /*+ INDEX (NGRAM3 <INDEXNAME>)*/ count (sid) from ngram3 where word1=<WORT1> and word2=<WORT2> and word3=<WORT3>;`

Tab. 17: SQL-Abfragen für n-Gramme

Erfahrungsgemäß verhalten sich SQL-Abfragen hochfrequenter Phänomene schon wegen der zu verarbeitenden Datenmenge deutlich laufzeitintensiver als Recherchen nach niederfrequenten Inhalten. Zur Befüllung der Platzhalter für WORT1, WORT2 und WORT3 ermitteln wir deshalb vorab für jeden Abfragetyp repräsentative Wörter bzw. Wortfolgen[107] mit abgestuften Auftretenshäufigkeiten in der n-Gramm-Tabelle.

Dazu bestimmen wir die Frequenzen und Ränge sämtlicher Mono-, Bi- und Trigramme und teilen diese in sieben Häufigkeitsklassen ein (HK-1 bis HK-7, vgl. Tab. 18 bis 20). Diese Token-Häufigkeitsklassen entsprechen formal den Frequenzangaben in unserem Referenzsystem, also den Spaltenwerten von CO_FREQCLASS in den Lemmalisten aus Abschnitt 3.2.3.

Jede SQL-Abfrage wurde fünfmal durchgeführt, um Mittelwerte und Konfidenzintervalle zu berechnen. Parallelisierungen kamen nicht zum Einsatz, die Abarbeitung eines Statements wurde von je einem CPU-Kern übernommen. Die Gesamtzahl der Abfragen ergibt sich aus der Anzahl der Testläufe pro SQL-Abfrage, der Abfragetypen (drei), der Häufigkeitsklassen (jeweils sieben) und der einzubeziehenden Indexvarianten (sechs Indizes plus jeweils ein Durchlauf ohne Indexnutzung): 5 × 3 × 7 × 7 = 735.

Um Caching-Effekte bei der Protokollierung mehrerer unmittelbar aufeinander folgender Testläufe zu vermeiden, wurde vor jedem SELECT-Statement der Buffercache der Datenbankinstanz explizit geleert.

Frequenz	Wort 1	Rang	Häufigkeitsklasse
24.561.941	*der*	1	HK-1
6.022.683	*des*	10	HK-2
693.072	*keine*	101	HK-3
78.603	*Wochen*	1.003	HK-4
5.916	*urkundliche*	10.000	HK-5
295	*Restlaufzeit*	100.000	HK-6
12	*durchschleusen*	1.000.000	HK-7

Tab. 18: Ausgewählte Beispielwörter für Monogramme

[107] Dabei beschränken wir uns auf Textwörter und numerische Zahlenwerte. Die Recherche nach – generell hochfrequenten – Satzzeichen wird in Abschnitt 3.3.4 thematisiert.

Frequenz	Wort 1	Wort 2	Rang	Häufigkeitsklasse
2.559.658	*in*	*der*	1	HK-1
711.638	*mit*	*dem*	11	HK-2
185.168	*auch*	*nicht*	100	HK-3
34.416	*war*	*und*	1.001	HK-4
5.227	*wir*	*kommen*	10.000	HK-5
585	*praktische*	*Umsetzung*	100.000	HK-6
48	*Schwierigkeiten*	*ergeben*	1.000.000	HK-7

Tab. 19: Ausgewählte Beispielwörter für Bigramme

Frequenz	Wort 1	Wort 2	Wort 3	Rang	Häufigkeitsklasse
705.852	*Damen*	*und*	*Herren*	2	HK-1
144.026	*bei*	*Abgeordneten*	*der*	10	HK-2
33.957	*in*	*der*	*Vergangenheit*	100	HK-3
8.173	*wenn*	*wir*	*uns*	1.000	HK-4
1.717	*zu*	*100*	*Prozent*	10.000	HK-5
291	*habe*	*sie*	*nicht*	100.000	HK-6
38	*war*	*jedoch*	*bis*	1.000.000	HK-7

Tab. 20: Ausgewählte Beispielwörter für Trigramme

Ein interessanter Nebeneffekt dieser Frequenz- und Rangbestimmung besteht in der Überprüfung des Zipf'schen Gesetzes[108] für Wortkombinationen. Dieses vermutlich bekannteste allgemeine Verteilungsgesetz von George Kingsley Zipf besagt, dass in einer nach Frequenzen geordneten Rangliste aller Wörter einer Sprache das Produkt aus Worthäufigkeit und Rang approximativ konstant ist. Vereinfacht ausgedrückt wird dadurch erklärt, dass wenige Wörter sehr häufig und viele Wörter sehr selten vorkommen. In einer doppelt logarithmischen Darstellung resultiert dieser Umstand in einer annähernden Geraden für log(Frequenz) und log(Rang). Abbildung 14 visualisiert die Verteilungen für Einzelwörter (Monogramme) sowie für Bi- und Trigramme im

[108] Der beschriebene Zusammenhang wurde inzwischen mehrfach für natürlichsprachliche Texte evaluiert. Neben einer grundsätzlichen Bestätigung zeigten Messungen aber auch notwendige Anpassungen auf, etwa für sehr seltene und sehr häufige Wörter; vgl. u.a. Zipf (1949); Prün (2005). Wir werden in Abschnitt 3.3.4 Auswirkungen der Zipf'schen Verteilung auf die Recherche nach hochfrequenten Wörtern aufzeigen.

n-Gramm-Korpus. Während erstere zumindest für höhere Rangzahlen der Zipf'schen Formel folgen, weisen Bi- und Trigramme erkennbare Abweichungen auf: Die Verläufe erscheinen eher kurvenförmig und die Steigung (*slope*) fällt zunehmend geringer aus. Damit entsprechen unsere Ergebnisse den Untersuchungen von Bardoel (2012) für n-Gramme in Zeitungskorpora.

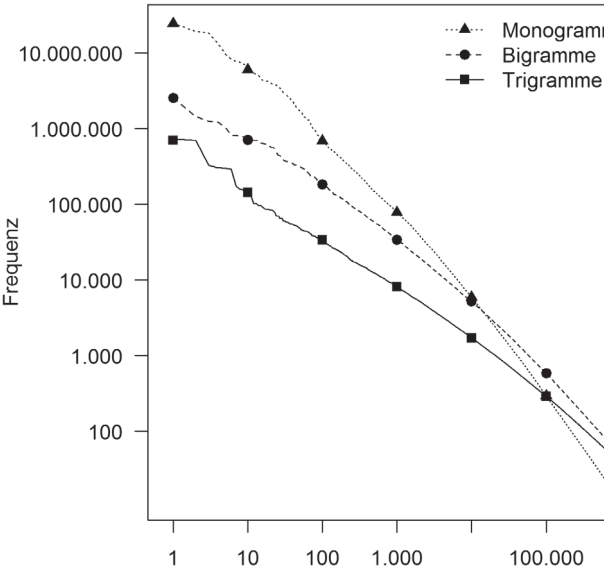


Abb. 14: Verteilung der n-Gramme (doppelt logarithmische Darstellung)

Betrachtet man die Abfragezeiten für Monogramme in Tabelle 21, so fällt zunächst die weitestgehend konstante Laufzeit für Abfragen ohne Indexnutzung ins Auge. Unabhängig von der Häufigkeitsklasse der einzelnen Beispielwörter liegt sie bei ca. 131-132 Sekunden. Dieses Verhalten entspricht unseren Erwartungen, weil für indexlose Abfragen stets ein kompletter Tabellenscan durchgeführt werden muss. Ebenso erwartungsgemäß divergieren die unterschiedlichen Indizes HK-übergreifend massiv, was auch in der logarithmierten Darstellung (Abb. 15 links oben) deutlich wird. Das Spektrum reicht von Sekundenbruchteilen für seltene Wörter bis hin zu mehreren Stunden für hochfrequente Vorkommen. Die mehrfachen Überschneidungen der Konfidenzintervalle[109] legen nahe, dass die Betrachtung weiter ausdifferenziert werden sollte, um signifikante Unterschiede zwischen den Ergebnissen aufzudecken.

[109] Konfidenzintervalle bezeichnen in der Statistik die Grenzen, innerhalb derer sich in der Grundgesamtheit, auf die durch die Stichprobe verallgemeinert werden soll, der tatsächliche Wert mit einer bestimmbaren Wahrscheinlichkeit befindet. Wir spezifizieren ein Signifikanzniveau von 0,05, d.h. 95% der tatsächlichen Werte liegen innerhalb der Intervallgrenzen.

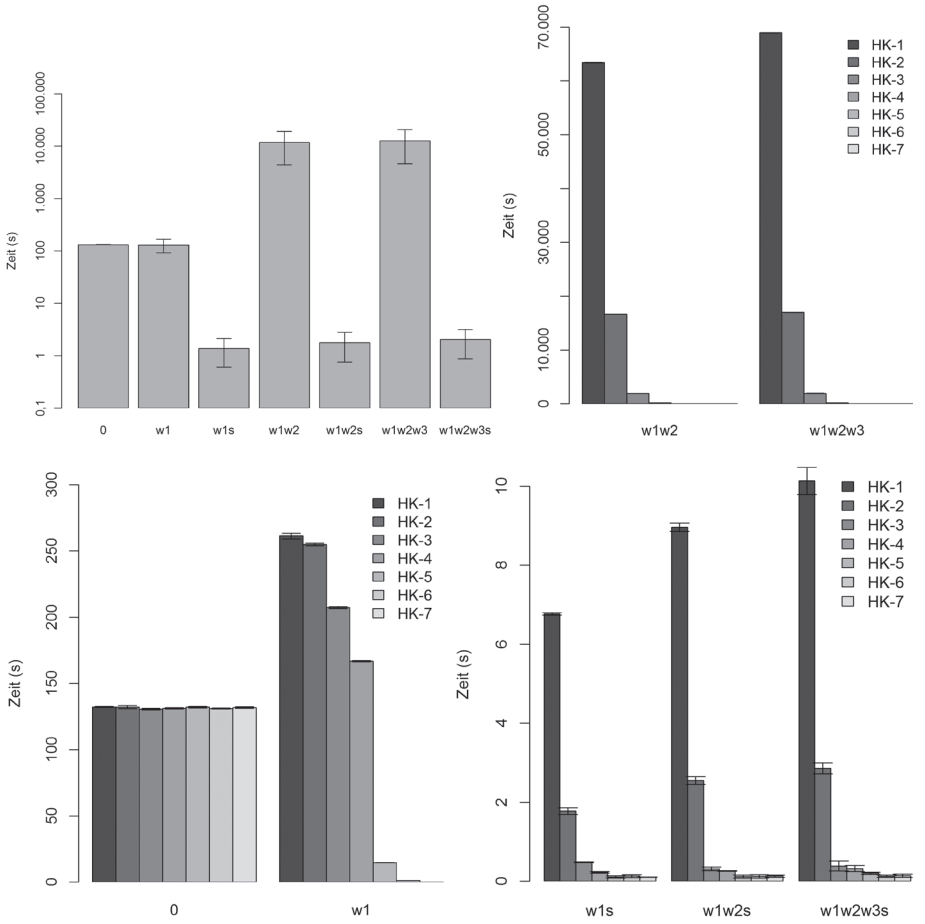


Abb. 15: Abfragezeiten für Monogramme (HK-übergreifend und HK-spezifisch)

Index	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
0	132,17	132,17	130,58	131,17	132,02	131,05	131,76
w1	261,32	254,98	207,17	166,78	14,72	1,24	0,12
w1s	6,76	1,78	0,48	0,23	0,11	0,13	0,1
w1w2	63.415,92	16.679,68	1.974,91	217,56	14,64	0,69	0,17
w1w2s	8,96	2,55	0,32	0,26	0,12	0,12	0,13
w1w2w3	68.933,52	17.025,44	1.982,13	229,55	14,51	0,23	0,16
w1w2w3s	10,13	2,86	0,39	0,33	0,2	0,13	0,14

Tab. 21: Mittelwerte der Monogramm-Abfragezeiten in Sekunden

Eine solche Verfeinerung erfolgt durch die explizite Gruppierung der Abfragezeiten nach Häufigkeitsklassen (Abb. 15 rechts oben sowie links und rechts unten, jeweils nicht logarithmiert). Dabei treten signifikante Unterschiede zutage: Index w1 verlängert die Laufzeiten der SQL-Statements gegenüber der indexlosen Suche für hohe und mittlere Wortfrequenzen, während dieselben für weniger frequente Phänomene sinkt. Dieses Verhalten lässt sich dadurch erklären, dass der Index zwar zunächst beim Auffinden der passenden Wörter von Nutzen ist, danach jedoch für die Ermittlung der Satzreferenz (SID) nochmals auf die Tabelle zugegriffen werden muss; dieser zusätzliche Aufwand steigt mit der Anzahl der Treffer. Zudem weist ein reiner Tokenindex eine hohe Dichte auf, d.h. die gegenüber eindeutigen Indexspalten hohe Anzahl an Duplikaten wirkt sich negativ auf die Performanz aus. Noch deutlicher wird der nachteilige Einfluss unpassender Indizes bei w1w2 und w1w2w3: Hier schlägt nicht nur der Umstand zu Buche, dass die Satzreferenz nicht in den Indizes enthalten ist, sondern auch die ineffektive primäre Selektionsleistung aufgrund der Anreicherung der Indexstrukturen um für die Abfragen nicht zielführende Tabellenspalten.

Index	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
0	165,55	164,37	165,45	165,15	164,72	164,3	165,16
w1	289,62	277,6	256,73	195,32	157,98	87,17	77,6
w1s	26.306,87	14.176,45	12.753,81	6.789,48	4.232,12	35,87	101,84
w1w2	232,9	193,2	167,52	89,77	15,26	1,79	0,25
w1w2s	0,85	0,31	0,11	0,08	0,06	0,06	0,03
w1w2w3	7.618,33	2.436,79	576,25	109,76	16,92	2,09	0,3
w1w2w3s	1,18	0,44	0,26	0,25	0,09	0,07	0,05

Tab. 22: Mittelwerte der Bigramm-Abfragezeiten in Sekunden

Eine analoge Betrachtung der Abfragezeiten für Bigramme (Tab. 22 und Abb. 16) bestätigt die ermittelten Zusammenhänge. Am schlechtesten bedienen die Indizes w1s (fehlende zweite Wortspalte) und w1w2w3 (geringe Selektionsleistung und fehlende Satzreferenz) die Anforderungen. Die Indizes w1 und w1w2 bieten ohne Satz-ID im Vergleich zur erneut konstanten indexlosen Suche zwar Vorteile für seltene Wortkombinationen, bei hochfrequenten Paaren liegen die Ergebnisse aber bereits im mehrminütigen Bereich. Weiterhin fällt ein positiver Zusammenhang zwischen Clustering Factor und Abfragezeit auf: w1 arbeitet beinahe durchgehend performanter als w1s, sobald Rückgriffe auf Tabellenspalten notwendig werden. Die besten Resultate erzielt

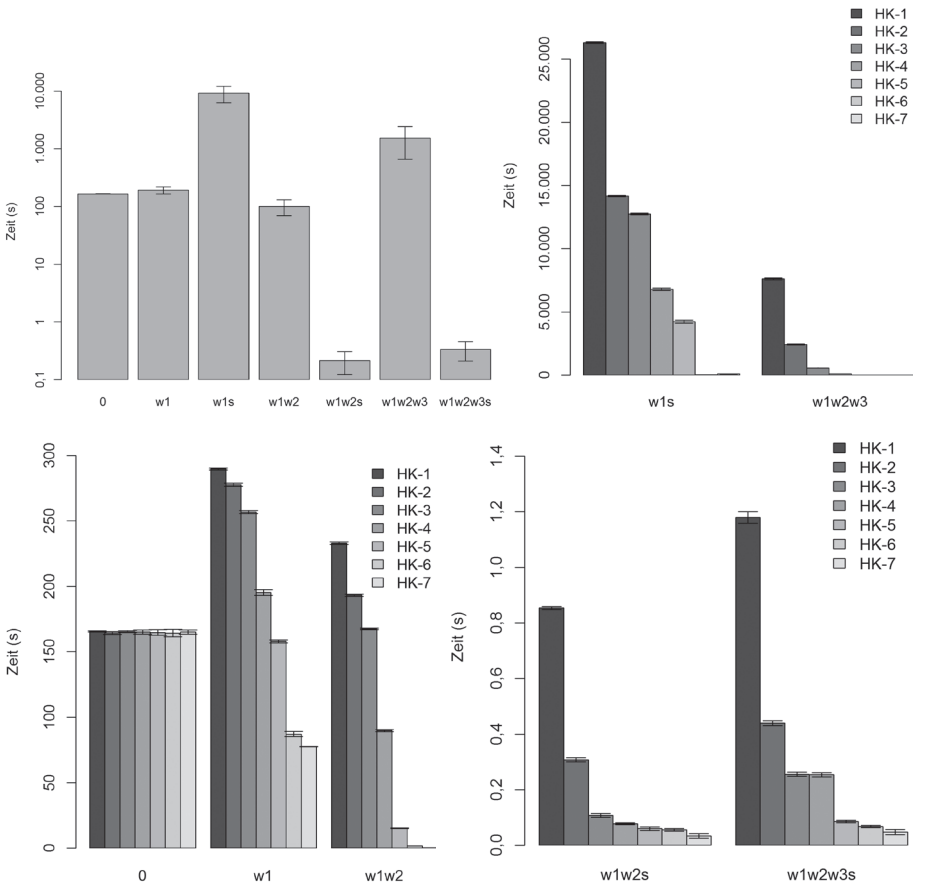


Abb. 16: Abfragezeiten für Bigramme (HK-übergreifend und HK-spezifisch)

w1w2s, gefolgt von w1w2w3s, allerdings mit erkennbaren Unterschieden bei häufigen Wortkombinationen.

Ein Blick auf die absoluten Zahlen der jeweils besten Indexvariante für Mono- und Bigramme zeigt eine erhebliche Verbesserung der Performanz bei Bigramm-Abfragen. Dieser Umstand erklärt sich aus dem deutlich niedrigeren Umfang insbesondere der hochfrequenten Klassen: HK-1 und HK-2 für Bigramme etwa enthalten Phänomene, die jeweils um ca. ein Zehntel seltener sind als Phänomene der entsprechenden Monogramm-Klassen.

Bei der Trigramm-Abfrage belegen die für Mono- und Bigramme leistungsstärksten Indizes w1s und w1w2s ungeachtet des vergleichsweise niedrigeren Clustering Factors mit Abstand die schlechtesten Plätze, gefolgt von w1 und

w1w2. Am besten schneiden Indizes ab, die sämtliche abgefragten Spalten abdecken. Interessant erscheint hier darüber hinaus ein anderer Aspekt: Die Abfragedauer von Wortkombinationen hängt bei unpassenden Indizes offenkundig massiv von den Verteilungen der initialen Einzelwörter ab, allerdings nicht nach einem intuitiv vermutbaren Muster. Beispielsweise könnte man annehmen, dass Indizes, die ausschließlich das erste Wort erfassen (w1 oder w1s), bei Trigrammen mit hochfrequentem WORD1 besser abschneiden als bei Trigrammen mit seltenerem Erstwort. Das Gegenteil ist der Fall: Bei „zu 100 Prozent“ (HK-5) oder „in der Vergangenheit“ (HK-3) beobachten wir extrem lange Abfragezeiten, während die Ergebnisse bei „Damen und Herren“ (HK-1) vergleichsweise rasch terminieren. Hier macht sich wieder der Umstand bemerkbar, dass für eine zunächst zwar rasch ermittelte, aber umfangreiche erste Selektion nachfolgend viele Tabellenspalten hinsichtlich der nicht im Index enthaltenen Attribute überprüft werden müssen.

Absolut betrachtet lassen sich bei Trigrammen erneut HK-übergreifend geringere Frequenzen konstatieren. Für die meisten Häufigkeitsklassen werden bei geeignetem Index die jeweiligen Wortkombinationen rascher gefunden als entsprechende Bigramme, weil die Trefferzahlen deutlich niedriger sind. Der optimale Index w1w2w3s benötigt selbst für HK-1 weniger als eine halbe Sekunde; für die weiteren Häufigkeitsklassen sind in den Testläufen kaum noch spürbare Unterschiede feststellbar.

Index	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
0	165,78	165,70	165,68	165,39	165,95	165,61	165,91
w1	157,88	252,26	271,24	188,97	290,87	188,76	194,56
w1s	2.388,51	10.986,25	26.271,10	3.318,78	21.187,30	2.336,51	6.809,39
w1w2	151,42	114,77	195,64	130,93	19,27	15,53	17,70
w1w2s	2.427,33	563,60	7.757,89	336,40	23,77	18,58	21,86
w1w2w3	150,04	115,85	85,35	24,03	5,52	1,50	0,22
w1w2w3s	0,36	0,12	0,12	0,06	0,08	0,10	0,06

Tab. 23: Mittelwerte der Trigramm-Abfragezeiten in Sekunden

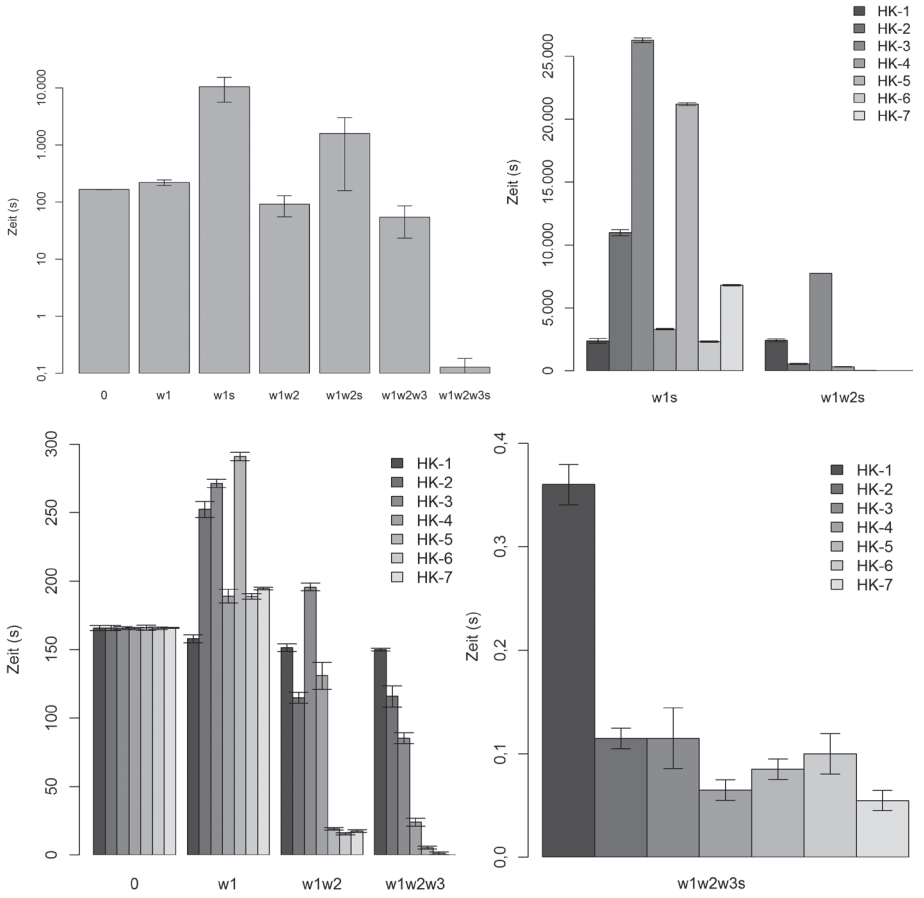


Abb. 17: Abfragezeiten für Trigramme (HK-übergreifend und HK-spezifisch)

Insgesamt bestätigen unsere Untersuchungen zu n-Gramm-Tabellen und -Indizes den bekannten Grundsatz, dass ein Index passend zur intendierten Abfrage angelegt sein sollte, um von Nutzen zu sein. In Konkurrenz stehen dabei maximale Selektivität einerseits sowie Indizierung sämtlicher zurückzuliefernder Attribute zur Vermeidung von Rückgriffen auf die Tabelle andererseits; unsere Testläufe zur Recherche nach Wortkombinationen legen eine Präferenz für die zweite Variante nahe. Übertragen auf die korpuslinguistische Praxis bedeutet das weiterhin: Wo gleichermaßen nach Einzelwörtern und potenziell beliebig umfangreichen Wortkombinationen gesucht wird, benötigen n-Gramm-Tabellen entsprechend viele Spezialindizes. Selbst bei einer Beschränkung auf Mono-, Bi- und Trigramme gibt es nicht die eine ideale Indexvariante. Vor allem bei hochfrequenten Phänomenen zeigen sich signifikante

Unterschiede, was in der Konsequenz zu einer drastischen Erhöhung von Indizierungsaufwand und Datenvolumen führt. Intensiviert würde dieser Effekt ggf. durch zusätzliche Spalten (Kombinationen aus vier oder mehr Textwörtern, Lemma- und Wortklassenangaben etc.), mit denen die Anzahl der performant abfragbaren Suchattribute wie auch der Maximalabstand zwischen diesen erhöht werden könnten.

3.3.1.2 Token-Tabellen

Ein alternatives Modell basiert auf der vollständigen Relationierung sämtlicher abfragerelevanten Primär- und Sekundärdaten sowie auf der Übersetzung komplexer Recherchen in geschachtelte SQL-Verknüpfungen (*self joins*), d.h. in durch logische Operatoren wie „AND" verbundene Unterabfragen. Chiarcos et al. (2008) verfolgen diesen Ansatz für die linguistische Datenbank ANNIS; Bird et al. (2005) messen die Abfragezeiten eines entsprechenden Datenbank-Frameworks für vergleichsweise kleine Testkorpora. Dabei werden annotierte Sprachbelege segmentiert und jeder Wortknoten mit Angaben zu den jeweiligen Vater- und Geschwisterknoten in einer separaten Tabellenzeile abgelegt. Künneth (2001) diskutiert ähnliche Strategien für den Umgang mit satzübergreifenden Korpusabfragen und favorisiert eine Type-/Token-Relationierung mit einer „n-Tabellen-Struktur", bei der Wortschlüssel für jeden Einzeltext in einer separaten Tabelle vorgehalten werden. Für Korpussammlungen mit Millionen von Texten überschreitet das letztgenannte Vorgehen allerdings rasch die Grenzen der Praktikabilität.

Spaltenname	Datentyp
SID	NUMBER(12)
ID	NUMBER(12)
WORD	CHAR(1000)

Tab. 24: Physikalischer Aufbau der relationierten Token-Tabelle

Der streng relationale Ansatz modelliert sämtliche Textwörter in einer gemeinsamen Token-Relation; Tabelle 24 veranschaulicht deren Zusammensetzung. Die Tabelle TOKENTAB speichert pro Zeile ein Textwort (WORD) gemeinsam mit der satzspezifischen Referenz (SID) und einer fortlaufenden Token-ID (ID), welche die lineare Abfolge im Satz dokumentiert. Da jedes Token nur einmal vorgehalten wird, erhöht sich das Speichervolumen nicht durch redundante Daten wie noch beim n-Gramm-Modell. Der Speicherbedarf für 1 Milliarde Tabellenzeilen liegt bei 25,5 GB, verteilt auf 817.695 Datenblöcke (zur Block-

größe vgl. Tab. 7). Gemeinsam mit einem zusammengesetzten B-Baum-Index[110] „wsid“ (siehe Tab. 25) benötigt diese Lösung deutlich unter 60 GB und damit weniger als die Hälfte der kumulierten Datenmenge aus n-Gramm-Tabelle und deren drei Top- Indizes w1s, w2w2s und w1w2w3s.

Index	Indizierte Spalten	Größe in MB	Clustering Factor	Indextiefe	Blattknoten
WSID	WORD,SID,ID	32.107	437.160.197	2	1.001.645

Tab. 25: Physikalischer Aufbau des zusammengesetzten TOKENTAB-Index

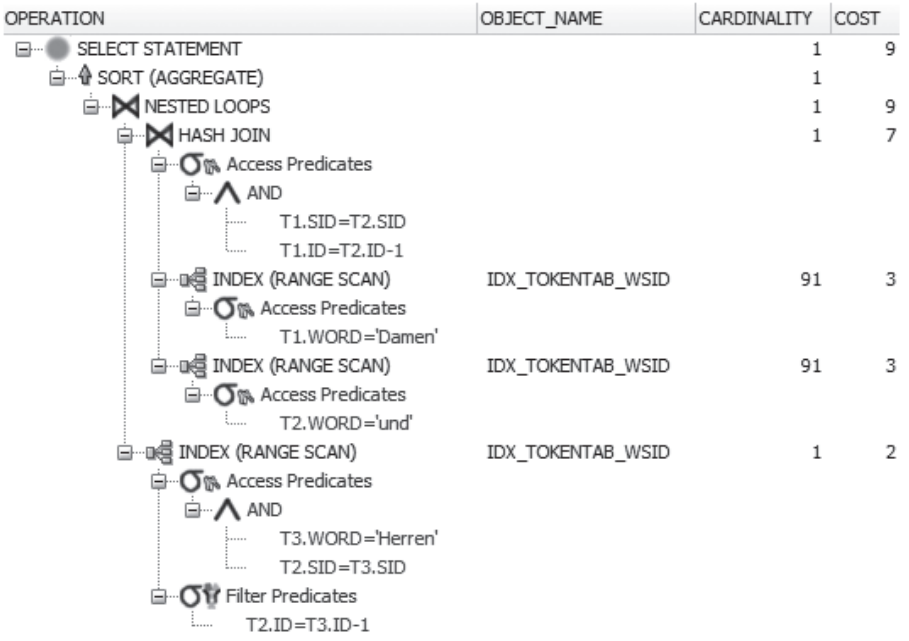


Abb. 18: Indexnutzung im Ausführungsplan einer TOKENTAB-Abfrage

[110] Alternativ unterstützen viele Datenbanksysteme weitere Indextypen, beispielsweise kombinierbare Bitmap-Indizes für Einzelspalten; zu den Vor- und Nachteilen vgl. z.B. Sharma (2005). Für Fälle, in denen ausschließlich Indexwerte abgefragt werden, kämen darüber hinaus sogenannte Index-organisierte Tabellen (*index organized tables, IOT*; vgl. Burleson 2014, S. 689 ff.) in Betracht. Tabellenwerte werden in diesen Fällen direkt in einer B-Baum-Struktur ohne Heap-Tabelle abgespeichert.

Durch die Verwendung fortlaufender Satz- und Token-IDs entfällt jedwede Limitierung der abfragbaren Wortabstände.[111] Für die Formulierung frequenzklassenspezifischer SQL-Abfragen kommen wiederum unsere Monogramm-, Bigramm- und Trigramm-Beispiele zum Einsatz:

– Recherche nach Einzelwörtern (ohne SQL-Verknüpfung), z.B.: `select count (sid) from tokentab where word=<WORT1>;`
– Recherche nach zwei unmittelbar aufeinander folgenden Wörtern (mit einer SQL- Verknüpfung), z.B.: `select count (t1.sid) from tokentab t1, tokentab t2 where t1.word=<WORT1> and t2.word=<WORT2> and t1.sid=t2.sid and t1.id=t2.id-1;`
– Recherche nach drei unmittelbar aufeinander folgenden Wörtern (mit zwei SQL- Verknüpfungen), z.B.: `select count (t1.sid) from tokentab t1, tokentab t2, tokentab t3 where t1.word=<WORT1> and t2.word=<WORT2> and t3.word=<WORT3> and t1.sid=t2.sid and t2.sid=t3.sid and t1.id=t2.id-1 and t2.id=t3.id-1;`
– Eine explizite Angabe des zu verwendenden Index ist nicht mehr notwendig. Der datenbankinterne Optimierer bestimmt unter Heranziehung von Datenverteilungs-Statistiken, Speichereigenschaften etc. den optimalen Ausführungsplan und entscheidet sich bei der vorliegenden Architektur stets für den zusammengesetzten Index; vgl. Abbildung 18.

Die in Abbildung 19 logarithmiert visualisierten Abfragezeiten des TOKENTAB-Modells folgen einer eindeutigen Tendenz. Zwar sinken auch sie beinahe ausnahmslos mit abnehmender Phänomenhäufigkeit, liegen aber bei zunehmender Anzahl der Verknüpfungen immer offensichtlicher über den Ergebnissen der n-Gramm-Tabelle.

Einen direkten Vergleich beider Modelle bietet Abbildung 20. Zusammengefasst lässt sich feststellen:

– Die Einzelwortabfragen liefern für beide Modelle nahezu identische Werte. Die im TOKENTAB-Index hinzugekommene Wort-ID-Spalte beeinflusst also nicht dessen Selektionsleistung.
– Die Laufzeiten der Bigrammabfragen weichen für seltene Phänomene (HK-6 und HK-7) ebenfalls kaum voneinander ab, steigen in der Verknüpfungsvariante für höhere Frequenzen aber überproportional.

[111] Für satzübergreifende Abfragen müsste darüber hinaus pro Datensatz eine zusätzliche Text-ID vorgehalten und indiziert werden, wobei dann evtl. andererseits die Satz-ID wegfallen dürfte.

Recherchen nach drei unmittelbar aufeinander folgenden Wörtern weisen in der TOKENTAB-Variante mit zwei SQL-Verknüpfungen den ungünstigsten Verlauf und geringsten Slope auf. Die Antwortzeiten liegen dort durchgehend im Bereich mehrerer Sekunden, während die n-Gramm-Variante in keinem Fall mehr als eine halbe Sekunde benötigt.

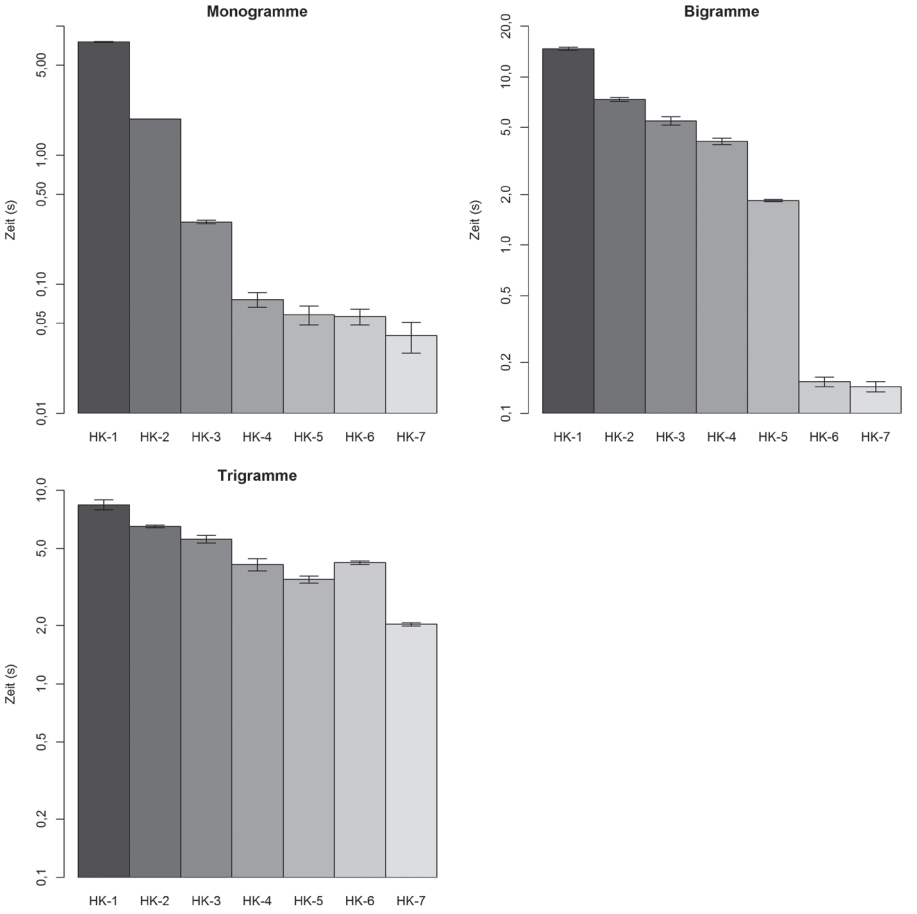


Abb. 19: Abfragezeiten der relationierten Token-Tabelle

Index	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
0	138,33	138,10	138,05	138,15	138,01	138,17	138,02
wsid	7,54	1,91	0,3	0,08	0,06	0,06	0,04

Tab. 26: Mittelwerte der TOKENTAB-Einzelwortabfragen in Sekunden

Index	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
0	281,75	278,00	276,38	277,17	275,54	275,7	275,56
wsid	14,67	7,33	5,47	4,13	1,84	0,15	0,14

Tab. 27: Mittelwerte der TOKENTAB-Bigrammabfragen in Sekunden

Index	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
0	413,65	413,96	419,50	415,18	413,47	416,80	414,03
wsid	8,42	6,51	5,59	4,13	3,46	4,22	2,03

Tab. 28: Mittelwerte der TOKENTAB-Trigrammabfragen in Sekunden

Als Fazit kann daher festgestellt werden, dass die konsequente Relationierung fortlaufender Token zwar zu einer gleichermaßen übersichtlichen wie speicherplatzfreundlichen Datenhaltung sowie zu ohne Einschränkungen spezifizierbaren Wortabständen in Suchabfragen führt. Für Korpusabfragen mit mehreren Suchparametern erscheint gleichwohl ein besser skalierbares Retrievalkonzept wünschenswert. Die TOKENTAB-Antwortzeiten liegen auf dem Referenzsystem in einem für den praktischen Einsatz tolerierbaren Bereich, sollten jedoch für komplexe Recherchen idealerweise noch optimiert werden.

Ein naheliegender Anhaltspunkt hierfür hat sich bereits herauskristallisiert, nämlich die Beschränkung der SQL-Verknüpfungen pro Abfrage. Alternative Vorgehensweisen bei Schlüsselwerten und hochfrequenten Phänomenen gilt es noch zu evaluieren; siehe hierzu die nachfolgenden Abschnitte 3.3.3 und 3.3.4. In Kapitel 4 wird unser relationales Modell explizit für variable Korpusgrößen auf den Prüfstand gestellt, wobei dann an Stelle einfacher Textwörter weitere linguistisch motivierte Suchattribute aus dem Anforderungskatalog zum Einsatz kommen.

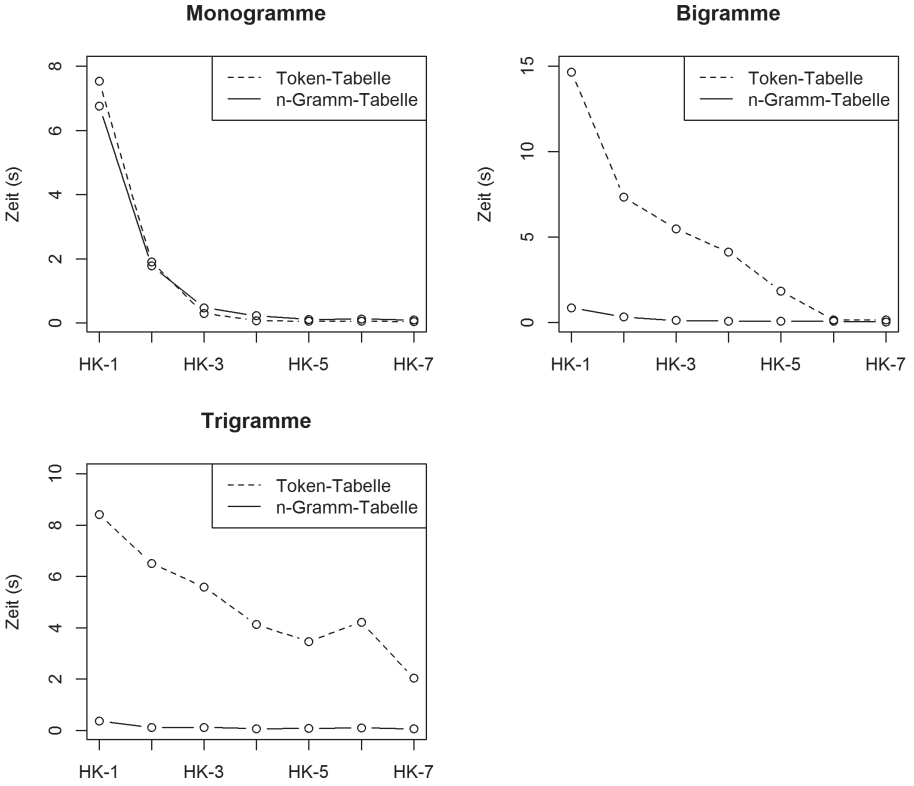


Abb. 20: Unterschiede zwischen n-Gramm- und Token-Tabelle

3.3.2 Platzhalteroperatoren und reguläre Ausdrücke

Platzhalteroperatoren (*wildcards*) und reguläre Ausdrücke (*regular expressions*, kurz *RegExp*) stellen mächtige Werkzeuge für die regelhafte Erweiterung von Suchmustern für Korpusrecherchen dar. Mit ihrer Hilfe ist es möglich, in einer Abfrage mehrteilige Kriterien zu spezifizieren und komplexe linguistische Phänomene zu erschließen. Gebräuchlich ist etwa die subsumierende Recherche nach Wörtern mit unterschiedlichen Flexionsendungen oder Pluralformen durch den Einsatz von Rechts-Trunkierungen. Dabei steht der Platzhalter (z.B. *?* für ein einzelnes Zeichen bzw. * für mehrere Zeichen) am Wortende: *Freund?* findet sowohl *Freund* als auch *Freunds* und *Freunde*; *wohn** findet *wohne*, *wohnst*, *wohnt* und *wohnen* (aber auch *wohnhaft* usw.!). Links-Trunkierungen hingegen erschließen Wortformen mit identischer Endung: **ismus* findet *Aktionismus*, *Hinduismus*, *Sozialismus*, *Tourismus* usw. Durch eine gleichzeitige Links- und Rechts-Trunkierung kann nach eingebetteten Sub-

strings gesucht werden: **gegen** findet *allgegenwärtig*, *dagegen*, *demgegenüber* usw. Und schließlich lassen sich Trunkierungsoperatoren auch im Wortinnern einsetzen*: geg*en* findet *gegangen*, *gegeben*, *gegessen* usw.

Weil solch vergleichsweise unspezifische Platzhalter in manchen Fällen ein gesuchtes Sprachphänomen nicht ausreichend exakt eingrenzen und infolgedessen auch ungewollte Ergebnisse einschließen, bieten sich reguläre Ausdrücke für feinkörnigere Formulierungen an.[112] Verbreitet ist hier der POSIX-Standard, der ebenso wie andere RegExp-„Dialekte" ausdrucksmächtige Klassen und Operatoren zur Verfügung stellt: Der Punkt (.) etwa steht für ein beliebiges Zeichen, *[:alnum:]* für alphanumerische Zeichen, *[:digit:]* für Zahlen oder *[:punct:]* für Satzzeichen. Es lassen sich auch selbstdefinierte Listen bzw. Gruppen verwenden, z.B. *[aeiou]* für alle Vokale oder (*Straße*|*Weg*|*Platz*|*Allee*) für alternative Strings. Mit Quantifizierern kann exakt angegeben werden, wie oft ein Teilmuster im Korpusbeleg vorkommen soll: Das Fragezeichen (*?*) steht für kein oder maximal ein Vorkommen, der Asterisk (***) für null bis unendlich viele Wiederholungen, das Pluszeichen (*+*) für ein bis unendlich viele Wiederholungen. Auch Wortanfänge oder -endungen lassen sich kodieren, ebenso Negationen und vieles mehr.

Die Mächtigkeit regulärer Ausdrücke erleichtert mithin die Formulierung komplexer Suchmuster, hat allerdings auf das Laufzeitverhalten eine grundsätzlich negative Auswirkung. Traditionelle Indizes lassen sich hierfür nämlich nicht ad hoc verwenden, weil beim Indexaufbau eben nicht alle potenziellen Mustervarianten abgebildet werden können. Als Konsequenz führen Datenbanksysteme beim Einsatz regulärer Ausdrücke in der WHERE-Klausel einen aufwändigen Full-Table-Scan (auch *sequential scan*) durch, der die Abfragezeiten in der Regel massiv verlängert.

Es stellt sich also die Frage, auf welche Weise Indexkonstruktionen für Korpusabfragen mit regulären Ausdrücken gewinnbringend eingesetzt werden können. Giles (2005) beschreibt einen Ansatz, der für zu durchsuchende Tabellenspalten multiple Bitmap-Indizes auf je zwei aufeinander folgenden Zeichen einrichtet. Ein SELECT-Suchausdruck wird fortan in mehrere Substring-Unterabfragen aufgeteilt, diese parallel in einer Pipe abgearbeitet und unter Nutzung der *regexp_like*-Funktion gefiltert. Das Vorgehen offenbart jedoch, abgesehen von einem damit einher gehenden erhöhten Indexaufwand und -volumen, bei längeren Wörtern signifikante Leistungseinbrüche. Schneider (2012) evaluiert den Substring-Ansatz deshalb gegen ein Volltextindex-basiertes Verfahren und weist nach, dass das Vorschalten dieses Volltextindex vor die eigentliche Abfrage des regulären Ausdrucks signifikante Performanzvorteile mit sich bringt.

[112] Vgl. z.B. Friedl (2006).

Daran knüpfen wir im Folgenden an, verwenden allerdings anstatt eines Volltext- einen manuell erstellten funktionalen Reverse-Index (WSID_REVERSE), der sämtliche Korpustoken mit umgekehrter Buchstabenfolge sowie die zugehörigen Satz- und Wort-IDs aufnimmt. Aufbau und Volumen entsprechen dem in Tabelle 25 dokumentierten WSID-Index mit dem wesentlichen Unterschied, dass die Indizierungsanweisung nun *REVERSE*(*WORD*), *SID*, *ID* lautet. Im Zusammenspiel bedienen beide Indizes damit explizite Rechts- oder Linkstrunkierungen, was für die in unseren Referenzkatalog aufgenommenen prototypischen Abfragen mit Platzhalteroperatoren (Abfrage 2) und regulären Ausdrücken (Abfragen 9 und 10) gleichsam adäquat erscheint.

Die Evaluation erfolgt auf der bereits eingeführten Token-Tabelle. Für die Abfragen mit Platzhaltern werden Suchmuster mit einer ähnlichen Frequenz ausgewählt, um vergleichbare Resultate zu erhalten. Reguläre Ausdrücke werden sowohl mit vorgeschalteter Rechts-Trunkierung als auch mit vorgeschalteter Links-Trunkierung durchgeführt; die Suchmuster entsprechen dabei unseren Referenzabfragen 9 und 10. Diese weisen zwar unverkennbar ungleiche Trefferzahlen (147 vs. 33.659) auf, sind allerdings durch die vorgeschaltete indexbasierte Filterung dennoch vergleichbar, da die regulären Ausdrücke erst auf dem eingeschränkten Suchraum (175.570 bzw. 152.504 Treffer, also in einer ähnlichen Größenordnung) zum Einsatz kommen.

Abfragetyp	**Suchmuster**	**Treffer**
Rechts-Trunkierung	gegen*	778.014
Links-Trunkierung	*gegen	727.133
Links-Rechts-Trunkierung	*gegen*	1.043.454
Innere Trunkierung	geg*en	780.699
RegExp/Rechts-Trunkierung	Will.+\-.+\-(Straße|Weg|Platz|Allee)$	147
RegExp/Links-Trunkierung	(http:\/\/)?www\..+?\.de$	33.659

Tab. 29: Abfragen mit Platzhalteroperatoren und z.T. regulären Ausdrücken

Abfragetyp	**Vorgeschaltetes Suchmuster**	**Treffer**
RegExp/Rechts-Trunkierung	Will*	175.570
RegExp/Links-Trunkierung	*.de	152.504

Tab. 30: Vorgeschaltete Suchmuster für reguläre Ausdrücke

Durchgeführt wurden jeweils fünf Testläufe pro Abfrage- und Indextyp.[113] Insgesamt waren dies also 5 × 6 × 3 = 90 Abfragen, stets ohne Cache-Nutzung und unter Nutzung eines einzigen CPU-Kerns.

Exemplarisch hier die verwendeten SQL-Statements für eine Rechts-Trunkierung; die Abfragen der drei weiteren Typen (Links-Trunkierung, Links-Rechts-Trunkierung, Innere Trunkierung) gestalten sich entsprechend:

0:
```
select /*+ NO_INDEX (tokentab)*/ count(sid) from tokentab
    where word like 'gegen%';
```

WSID:
```
select /*+ INDEX (tokentab idx_wsid)*/ count(sid) from
    tokentab where word like 'gegen%';
```

WSID_REVERSE:
```
select /*+ INDEX (tokentab idx_wsid_reverse)*/
    count(sid) from tokentab where reverse(word) like
    reverse('gegen%');
```

Die Ergebnisse für die Trunkierungsabfragen mit Platzhalteroperatoren, aber ohne reguläre Ausdrücke (Abb. 21 mit logarithmisch skalierter y-Achse sowie Tab. 31) bieten keine nennenswerten Überraschungen: Bei der Rechts-Trunkierung liegt der nicht-reverse WSID-Index eindeutig vorne, während der umgekehrte Index sogar deutlich ungünstigere Resultate als ein Full-Table-Scan liefert. Für die Links-Trunkierung kehrt sich das Bild um. Unser Beispiel für innere Trunkierung (*geg*en*) nutzt den WSID-Index effektiver als den Reverse-Index, vermutlich aufgrund der höheren Selektivität des initialen Teilstrings *geg* gegenüber dem finalen *en*, das für die Indexabfrage in ein initiales *ne* gewandelt wird. Im Vergleich zur Wildcard-losen TOKENTAB-Einzelwortrecherche in Abschnitt 3.3.1 fällt interessanterweise kein signifikanter Leistungsabfall durch den Einsatz der Platzhalteroperatoren auf – der dort gemessene Mittelwert für die entsprechende Häufigkeitsklasse 3 von 0,3 Sekunden wird von den jeweils optimalen Trunkierungs-Testläufen (0,33 bzw. 0,36 Sekunden) bestätigt. Recherchen mit Platzhalteroperatoren müssen bei optimaler Indexnutzung also nicht zwangsläufig zeitaufwändiger ausfallen als Suchen nach exakten Übereinstimmungen. Eine Ausnahme bildet, auch dies keine Überraschung, die Rechts-Links-Trunkierung, die von keiner der beiden Indexvarianten positiv profitieren kann.

[113] Folgende Indexvarianten kamen zum Einsatz: 0=kein Index, WSID=nicht-reverser Index aus Wort/Satz-ID/Wort-ID, WSID_REVERSE= reverser Index aus Wort/Satz-ID/Wort-ID.

Index	Links-Trunkierung	Rechts-Trunkierung	Links-Rechts-Trunkierung	Innere Trunkierung
0	165,61	164,39	168,73	164,57
WSID	411,90	0,33	429,30	0,58
WSID_REVERSE	0,36	422,06	423,20	45,42

Tab. 31: Mittelwerte der Abfragen mit Platzhaltern in Sekunden

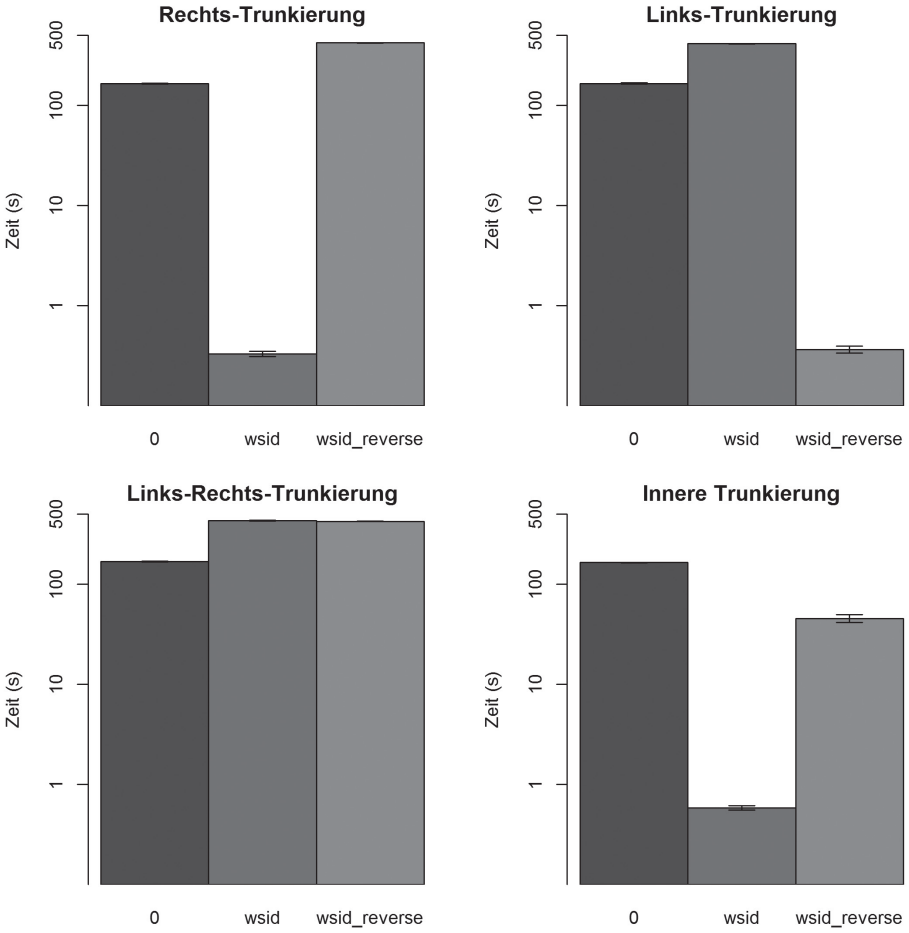


Abb. 21: Abfragen mit Platzhaltern

Ein exemplarischer Testlauf für Abfragen mit regulären Ausdrücken (hier: RegExp/Links-Trunkierung) umfasst folgende SQL-Statements:

0: `select /*+ NO_INDEX (tokentab)*/ count(sid) from tokentab where REGEXP_LIKE (word, '(http:\/\/)?www\..+?\.de$');`

WSID: `select /*+ INDEX (tokentab idx_wsid)*/ count(sid) from tokentab where word like '%.de' and REGEXP_LIKE (word, '(http:\/\/)?www\..+?\.de$');`

WSID_REVERSE: `select /*+ INDEX (tokentab idx_wsid_reverse)*/ count(sid) from tokentab where reverse(word) like reverse('%.de') and REGEXP_LIKE (word, '(http:\/\/)?www\..+?\.de$');`

Index	RegExp/Links-Trunkierung	RegExp/Rechts-Trunkierung
0	521,86	581,07
WSID	481,16	1,77
WSID_REVERSE	5,46	469,18

Tab. 32: Mittelwerte der Abfragen mit regulären Ausdrücken in Sekunden

In den Ergebnissen (Tab. 32 sowie Abb. 22 mit logarithmisch skalierter y-Achse) wird deutlich, dass sich unsere Strategie der Vorschaltung reverser bzw. nichtreverser Indizes vor die *REGEXP_LIKE*-Funktion enorm positiv auszahlt. Ohne Indexnutzung erfordert die Auflösung regulärer Ausdrücke noch das zeitlich Mehrfache eines Full-Table-Scans mit Platzhalteroperatoren. Die Einbeziehung passender Indizes für die mit *LIKE* in das Statement integrierten Platzhalter führt dagegen zu signifikant beschleunigten Testläufen. Dabei steht außer Frage, dass absolute Abfragezeiten gerade für reguläre Ausdrücke je nach Länge und Selektionsgrad des über die *LIKE*-Suche vorgeschalteten Suchmusters stark variieren können. Der Aussagewert unserer Testreihe liegt deshalb gerade hier weniger im absoluten, systemübergreifenden Benchmarking, sondern im Nachweis einer generell positiven Tendenz.

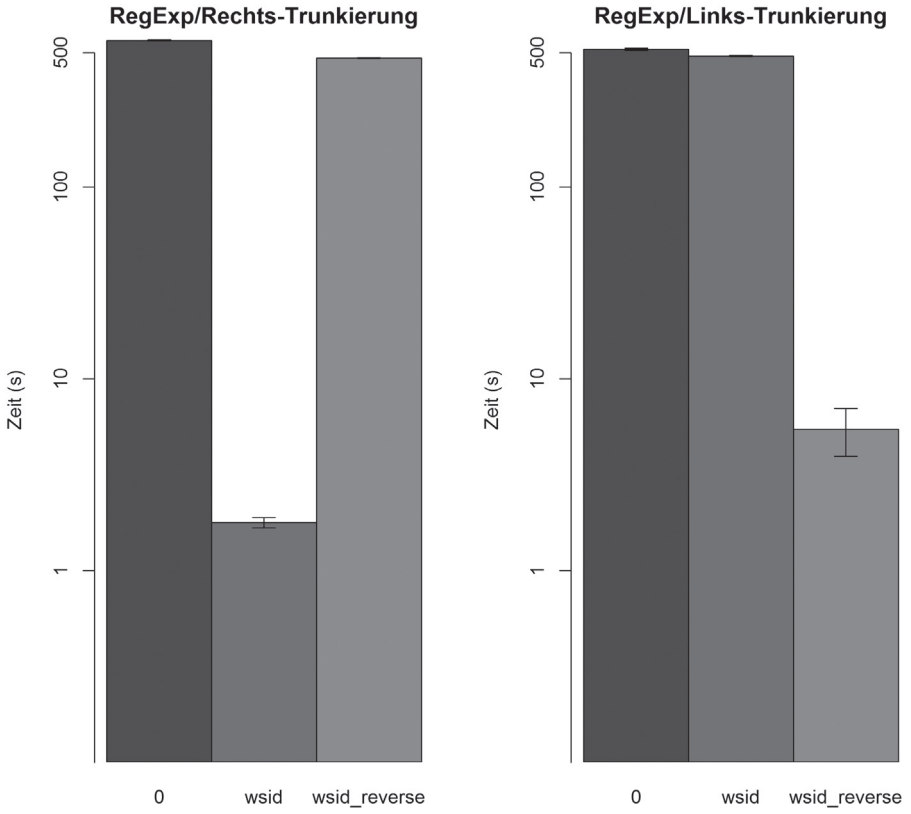


Abb. 22: Abfragen mit regulären Ausdrücken

3.3.3 Numerische und textuelle Schlüsselwerte

Sämtliche Indizes, die wir bislang für Recherchen nach Token und deren linearen Verkettungen genutzt haben, enthielten neben numerischen ID-Werten wenigstens eine textuelle Wortspalte als jeweils initiales Indexattribut. Daran anknüpfend erscheint die kontrastive Untersuchung einer potenziell effektiveren Indizierungsstrategie unter ausschließlicher Verwendung numerischer Werte, also durch die Kodierung von Textwörtern mit Hilfe eindeutiger Zahlen, naheliegend. Die damit verbundene Hoffnung basiert auf einer möglicherweise effizienteren Abfrage durchgehend numerischer Datenbank-Indizes, die im Kontext der datenbankgestützten Abfrage von Textkorpora bereits thematisiert, aber nicht umfassend evaluiert wurde:

> Verminderung von Redundanz bei der Speicherung der Korpusdaten: Für die Speicherung von Zeichenketten benötigen Datenbanksysteme mehr Kapazität auf dem Massenspeicher als für Zahlen. Steigerung der Suchgeschwindigkeit: Der Vergleich von Zahlen im Rahmen einer Suchoperation ist weniger aufwendig als die Suche nach Übereinstimmung in Zeichenketten. (Künneth 2001, S. 35).

Beide im Zitat genannten Aspekte, also Speicherbedarf und Abfragegeschwindigkeit, sollen nachfolgend überprüft werden. Dabei liefert ein Blick auf die Längen der erfassten Textwörter einen ersten Anhaltspunkt zur erreichbaren Volumenreduzierung. Als mittlere Wortlänge sämtlicher in TOKENTAB enthaltenen Wörter messen wir gerundet 5 Zeichen, bei einer maximalen Wortlänge von 883 Zeichen.[114] Auch wenn mehrfach positiv validierte statistische Untersuchungen einen grundsätzlich diametralen Zusammenhang zwischen Frequenz und Länge von Wörtern belegen,[115] so dass wir für hochfrequente – d.h. den Abarbeitungsaufwand einer Abfrage signifikant erhöhende – Wörter tendenziell kurze Längenwerte erwarten dürfen, erscheint hier doch ein gewisses Optimierungspotenzial vorstellbar.

Zu beachten ist eine den Abfragecode betreffende Konsequenz der Substituierung von Textwörtern durch Zahlenwerte: Um weiterhin intuitive formulierbare SELECT-Statements zu ermöglichen, ist der Rückgriff auf eine gesonderte Type-Tabelle erforderlich, die den Zusammenhang zwischen Suchwörtern und deren numerischen Äquivalenten kodiert. Nur auf diese Weise lässt es sich einrichten, dass der menschliche Nutzer Suchwörter – und nicht

[114] Dabei finden sich unter den aus mehreren hundert Zeichen bestehenden Werten beinahe ausnahmslos Internetadressen mit entsprechend langen Übergabeparametern, die in der Praxis kaum als Suchwerte genutzt werden dürften.

[115] Vgl. z.B. Best (2006, S. 26 ff.).

deren numerische Repräsentation – direkt eingibt. Konsequenterweise zieht diese Modellierungsvariante zusätzliche Verknüpfungen nach sich.

Tabelle 33 dokumentiert eine entsprechend modifizierte Relationierung. Die Token-Tabelle TOKENTAB_NUM enthält in der dritten Spalte nicht mehr einzelne Textwörter, sondern numerische Referenzen (WORD-REF); diese werden in der neu hinzu gekommenen Type-Tabelle TYPETAB_NUM aufgeschlüsselt. Als WORD-REF-Werte verwenden wir die Rangzahlen der jeweiligen Types („der" = 1, „des" = 10, „urkundliche" = 10.000 etc.), damit hochfrequente Wörter die niedrigsten und damit kürzesten Referenzen erhalten. Auf diese Weise erreichen wir im Vergleich zu einer rein zufälligen Vergabe von WORD-REF-Werten eine leichte, aber reproduzierbare Beschleunigung der Abfragezeiten für häufigere Wörter.[116]

Spaltenname TOKENTAB_NUM	**Datentyp**
SID	NUMBER(12)
ID	NUMBER(12)
WORD-REF	NUMBER(12)

Spaltenname TYPETAB_NUM	**Datentyp**
WORD-REF	NUMBER(12)
WORD	CHAR(1000)

Tab. 33: Physikalischer Aufbau der modifizierten Token-Tabelle (oben) und der Type-Tabelle (unten)

Der Speicherbedarf für 1 Milliarde Tabellenzeilen von TOKENTAB_NUM liegt bei 25,1 GB (ursprüngliche TOKENTAB 25,5 GB), verteilt auf 809.854 Datenblöcke (ursprüngliche TOKENTAB 817.695 Datenblöcke). Die Type-Tabelle mit 10.964.725 Tabellenzeilen benötigt ein Speichervolumen von 533,4 MB, verteilt auf 17.038 Datenblöcke. Die beiden kumulierten Tabellengrößen der Neumodellierung liegen also minimal über dem Wert der ursprünglichen TOKENTAB. Gleiches gilt für die in Tabelle 34 dokumentierten Indizes (WSID auf TOKENTAB_NUM, NUMREF auf TYPETAB_NUM). Eine nennenswerte Reduzierung des vergleichbaren Datenvolumens wird erreicht, sobald zusätzlich auch wieder ein Reverse-Index (NUMREF_REV) aufgebaut wird, weil dieser auf TYPETAB_NUM wesentlich kleiner als auf der Token-Tabelle ausfällt.

[116] Diese lag in der Häufigkeitsklasse HK-1 bei ca. 0,5s, für Wörter der Häufigkeitsklasse HK-2 noch bei ca. 0,1s.

Index	Indizierte Spalten	Größe in MB	Clustering Factor	Index-tiefe	Blatt-knoten
WSID	WORD-REF,SID,ID	31.782	480.215.227	2	1.032.302
NUMREF	WORD, WORD-REF	418	10.672.120	2	12.795
NUMREF_REV	REVERSE(WORD), WORD-REF	418	10.418.031	2	12.595

Tab. 34: Physikalischer Aufbau der modifizierten TOKENTAB-Indizes

Evaluiert wurden zunächst – in erneut jeweils fünf Testläufen auf einem CPU-Kern ohne Buffercache-Nutzung – unsere eingeführten Abfragen nach Mono-, Bi- und Trigrammen. Um neben der praxisnahen Verwendung einer zusätzlichen Verknüpfung zwischen Type- und Token-Tabelle (Variante VERKNÜPFUNG) auch Aussagen zur grundsätzlichen Eignung numerischer Schlüsselwerte treffen zu können, wurde jede Abfrage zusätzlich direkt, d.h. unter Eingabe des Referenzwerts (Variante DIREKT), getestet. Für Monogramme der HK-1 kamen damit nachstehende SQL-Statements zum Einsatz:

VERKNÜPFUNG:
```
select count (sid) from tokentab_num t1, typetab_num t2 where t2.word='der' and t1.word-ref=t2.word-ref;
```

DIREKT:
```
select count (sid) from tokentab_num where word=1;
```

Bigramme der HK-1 wurden wie folgt abgefragt:

VERKNÜPFUNG:
```
select count (t1.sid) from tokentab_num t1, tokentab_num t2, tokentab_numref t3, tokentab_numref t4 where t1.word=t3.id and t2.word=t4.id and t3.word='in' and t4.word='der' and t1.sid=t2.sid and t1.id=t2.id-1;
```

DIREKT:
```
select count (t1.sid) from tokentab_num t1, tokentab_num t2 where t1.word=4 and t2.word=1 and t1.sid=t2.sid and t1.id=t2.id-1;
```

Für Trigramme der HK-1 lauteten die SQL-Statements:

VERKNÜPFUNG:
```
select count (t1.sid) from tokentab_num t1, tokentab_num t2, tokentab_num t3, tokentab_numref t4, tokentab_numref t5, tokentab_numref t6 where t1.word=t4.id and t2.word=t5.id and t3.word=t6.id and t4.word='Damen' and t5.word='und' and t6.word='Herren' and t1.sid=t3.sid and t2.sid=t3.sid and t1.id=t2.id-1 and t2.id=t3.id-1;
```

DIREKT:
```
select count (t1.sid) from tokentab_num t1, tokentab_
num t2, tokentab_num t3 where t1.word=114 and t2.word=2
and t3.word=109 and t1.sid=t2.sid and t2.sid=t3.sid and
t1.id=t2.id-1 and t2.id=t3.id-1;
```

Variante	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
DIREKT	8,29	2,04	0,23	0,09	0,05	0,03	0,03
VERKNÜPFUNG	8,32	2,05	0,29	0,10	0,06	0,04	0,04

Tab. 35: Mittelwerte der Einzelwortabfragen mit numerischen Schlüsselwerten in Sekunden

Variante	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
DIREKT	15,79	7,66	5,51	4,70	1,6	0,17	0,15
VERKNÜPFUNG	16,53	7,88	5,53	4,58	1,9	0,18	0,20

Tab. 36: Mittelwerte der Bigrammabfragen mit numerischen Schlüsselwerten in Sekunden

Variante	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
DIREKT	10,69	6,64	5,84	3,12	3,46	3,48	2,13
VERKNÜPFUNG	40,69	31,24	24,88	18,88	18,46	19,93	13,93

Tab. 37: Mittelwerte der Trigrammabfragen mit numerischen Schlüsselwerten in Sekunden

Insgesamt lassen sich zwischen den Ergebnissen der DIREKT-Variante und den ursprünglichen Werten der Token-Tabelle aus Abschnitt 3.3.1 keine nennenswerten Unterschiede belegen; vgl. Abbildung 23. Insbesondere bestätigt sich nicht die Hoffnung auf eine Beschleunigung der Abfragezeiten durch ausschließlich numerische Indizes – ganz im Gegenteil: die numerische Direktabfrage für höherfrequente Suchmuster führt sogar zu leicht längeren Suchzeiten als die Abfrage mit textuellen Schlüsseln. Dies gilt für Monogramme der Häufigkeitsklassen HK 1 bis 2, für Bigramme in HK 1 bis 4 sowie für Trigramme in HK 1 bis 3. Die zusätzlichen Tabellenverknüpfungen wirken sich durchgehend laufzeitverlängernd aus; vgl. Tabellen 35 bis 37 und Abbildungen 24 bis 26. Deutlich spürbar wird dieser Effekt allerdings erst bei Trigrammen, d.h. bei Verknüpfungen zwischen insgesamt sechs Datensets.

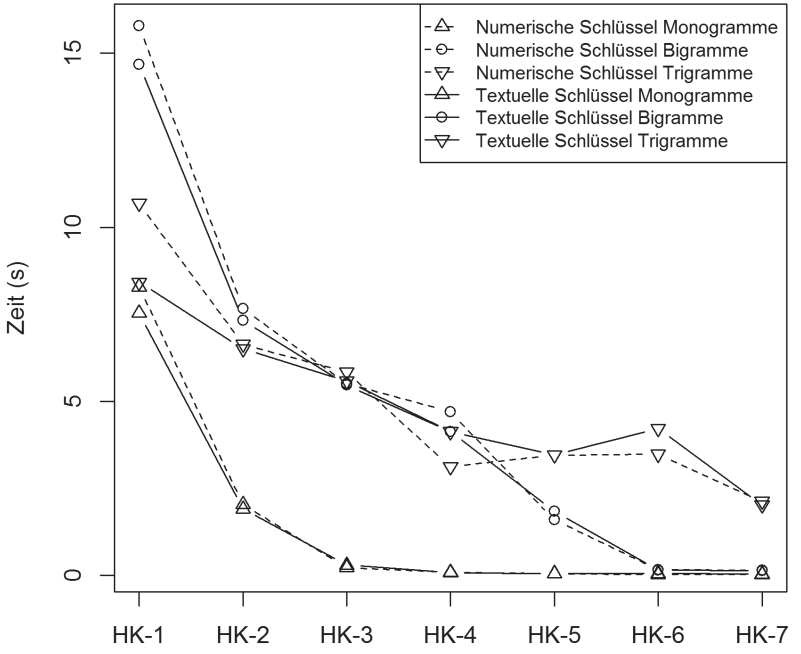


Abb. 23: Vergleich der Abfragezeiten für numerische bzw. textuelle Schlüsselwerte

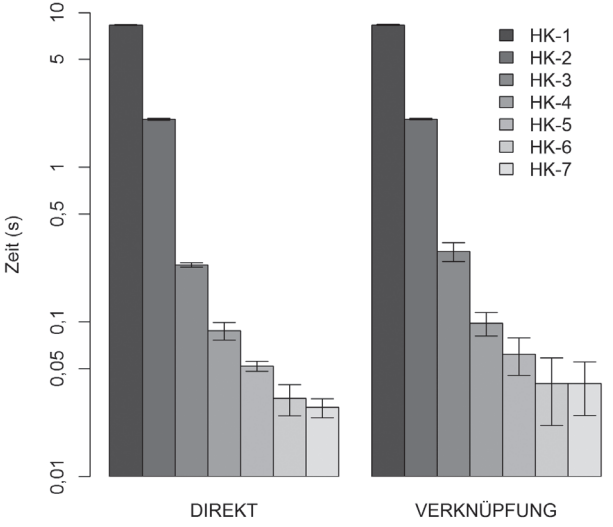


Abb. 24: Abfragezeiten der Einzelwortabfragen mit numerischen Schlüsselwerten

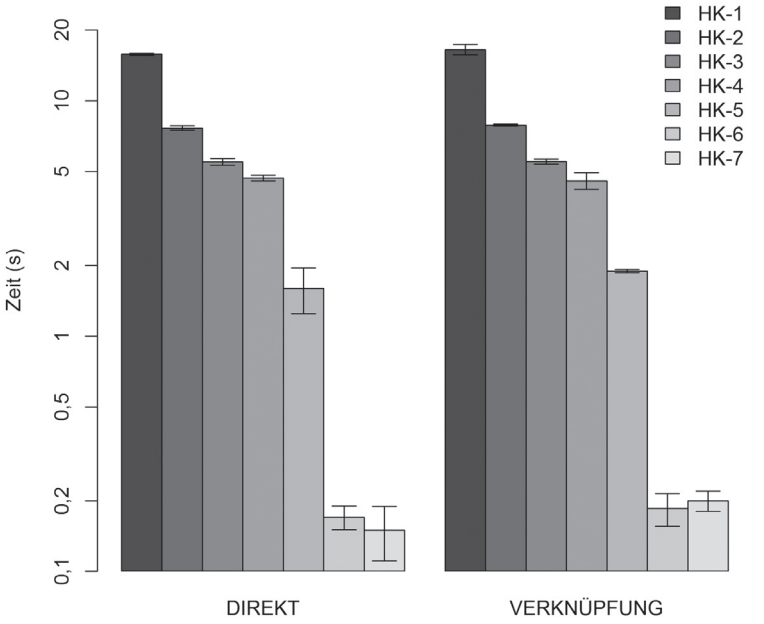


Abb. 25: Abfragezeiten der Bigrammabfragen mit numerischen Schlüsselwerten

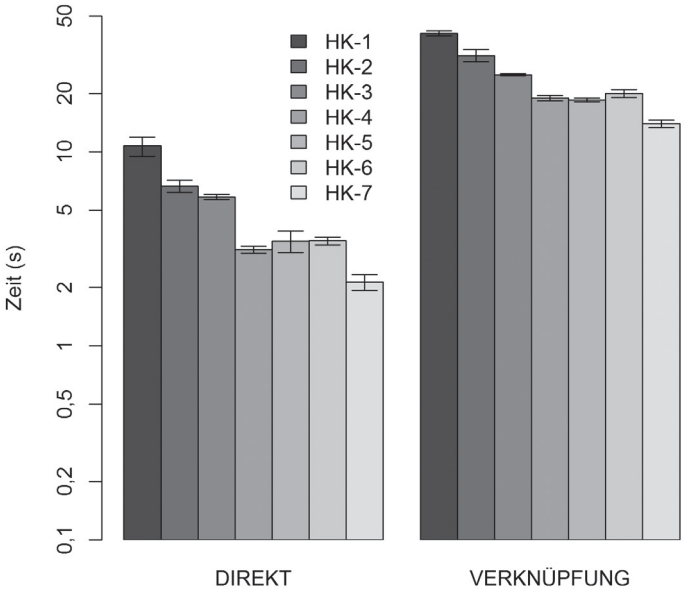


Abb. 26: Abfragezeiten der Trigrammabfragen mit numerischen Schlüsselwerten

Nach diesen eher ernüchternden Resultaten für eindeutige Suchmuster mit durchgehend numerischer Indizierung und Type-Token-Relationierung gilt es noch der Frage nachzugehend, wie Abfragen mit Platzhalteroperatoren und regulären Ausdrücken auf eine entsprechende Umstellung reagieren. Eine Unterscheidung zwischen DIREKT- und VERKNÜPFUNGS-Variante entfällt dabei aufgrund der Suchmuster-Trunkierung, d.h. alle aus Abschnitt 3.3.2 bekannten Abfragen werden mit Verknüpfung zwischen Type- und Token-Tabelle unter Nutzung der jeweils optimalen Indexvariante[117] ausgeführt. Für das Suchmuster mit RegExp/Links-Trunkierung liest sich das SQL-Statement nun wie folgt:

```
select count(sid) from tokentab_num t1, typetab_num t2 where
reverse(t2.word) like reverse('%.de') and REGEXP_LIKE (t2.word,
'(http:\/\/)?www\..+?\.de$') and t2.word-ref = t1.word-ref;
```

Die Ergebnisse der Trunkierungs-/RegExp-Abfragen (Tab. 38 und Abb. 27) unterstreichen die bereits beobachtete Tendenz. Die Verwendung numerischer Indizes vervielfacht die gemessenen Abfragezeiten drastisch, und das obwohl die mutmaßlich aufwändigen LIKE- bzw. REGEXP_LIKE-Operationen nun auf die vergleichsweise kleine Type-Tabelle angewendet werden. Selbst dieser prinzipielle Modellierungsvorteil wiegt also nicht die Nachteile zusätzlicher Verknüpfungen auf.

Insgesamt lässt sich deshalb feststellen, dass numerische Indizes das Laufzeitverhalten von Korpusabfragen in unserem Datenmodell tendenziell negativ beeinflussen. Positive Effekte lassen sich bestenfalls hinsichtlich des zu verwaltenden Datenvolumens beobachten, sobald mehrere Indizes – etwa ein zusätzlicher Reverse-Index – benötigt werden.

Abfragetyp	Abfragedauer
Rechts-Trunkierung	4,53
Links-Trunkierung	5,79
Innere Trunkierung	7,50
Links-Rechts-Trunkierung	179,39
RegExp/Rechts-Trunkierung	11,06
RegExp/Links-Trunkierung	34,86

Tab. 38: Mittelwerte der Platzhalter-Abfragen mit numerischen Schlüsselwerten in Sekunden

[117] Die Abfrage mit Links-Rechts-Trunkierung führt wiederum einen Full Table Scan durch.

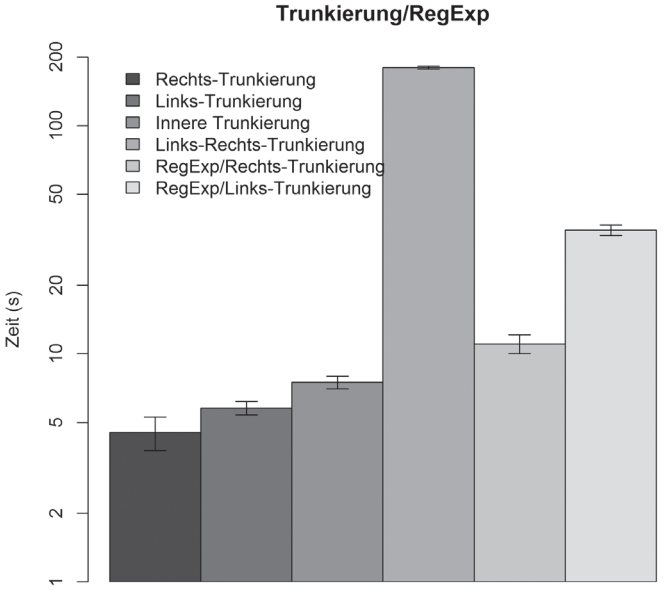


Abb. 27: Abfragezeiten der Platzhalter-Abfragen mit numerischen Schlüsselwerten

3.3.4 Hochfrequente Phänomene

Unsere bislang durchgeführten Testläufe belegen, dass Korpusretrievalzeiten maßgeblich von den Frequenzen der abzufragenden Phänomene abhängen. Je umfangreicher die zu verarbeitende Treffermenge, desto langwieriger gestaltet sich die Recherche. Dieser Zusammenhang zeigt sich unabhängig von Modellierungsvariante bzw. Indextyp und führt zu der naheliegenden Frage, ob hochfrequente Phänomene in Korpusdatenbanken nicht einer gesonderten Modellierung bedürfen.

In Anlehnung an das bereits erwähnte Zipf'sche Gesetz bietet sich etwa ein Rückgriff auf die bekannte 80-20-Regel – auch Pareto-Prinzip genannt – an. Diese entspricht mathematisch der Zipf-Verteilung und kann bei technischen bzw. organisatorischen Aufgabenbereichen zur Optimierung der Produktivität herangezogen werden.[118] Für die nachfolgenden Testläufe verzichten wir

[118] Die ursprünglichen Untersuchungen von Vilfredo Pareto (zusammengefasst in Pareto 2007) bezogen sich auf die Einkommensverteilung im Italien des ausgehenden 19. Jahrhunderts. Dabei wurde festgestellt, dass sich 80 Prozent des Gesamtvermögens in 20 Prozent der italienischen Familien konzentrierten und dass sich für Banken folglich bei einer Konzentration auf diese 20 Prozent mit vergleichsweise geringem Aufwand ein hoher Ertrag erwirtschaften ließe. Diese – in ihren sozialpolitischen Auswirkungen durchaus diskussionswürdige – Interpretation wurde seither zur Beschreibung diverser anderer empirischer Verteilungen herangezogen.

hinsichtlich der konkreten Verteilung zunächst auf tiefergehende empirisch-theoretische Analysen und wenden das Prinzip unmittelbar auf den vorhandenen Korpusbestand an. Dabei konzentrieren wir uns auf diejenigen hochfrequenten Token (nicht Types!), die in der Summe 20 Prozent des Inventars ausmachen, und implementieren für diese ein gesondertes Speicherungs- und Abfragemodell. Ob sich mit der Konzentration auf 20 Prozent der Wortformen tatsächlich 80 Prozent aller im wissenschaftlichen Alltag anfallenden Korpusabfragen optimieren lassen, muss an dieser Stelle dahingestellt bleiben und wäre ein lohnenswerter Gegenstand detaillierter Nutzungsstudien.

Ausgangspunkt ist die eingeführte Token-Tabelle TOKENTAB. Ca. 20 Prozent aller darin enthaltenen Zeilen (Types) beziehen sich auf genau acht Phänomene (Token), nämlich die Satzzeichen Komma, Punkt, öffnende Rundklammer, schließende Rundklammer sowie die Wortformen *der*, *und*, *die* und *in*. Diese isolieren wir nun in separaten Token-Tabellen (TOKENTAB_KOMMA etc.), die keine gesonderte WORD-Spalte mehr benötigen und lediglich fortlaufende Satz- und Tokennummern enthalten. Die restlichen 804.209.606 Datensätze speichert TOKENTAB_REST in den bekannten Spalten WORD, SID und ID (Frequenzen und Datenvolumen siehe Tab. 39). Auf Basis dieses modifizierten Datenmodells lassen sich nun Effekte einer physischen Segmentierung hochfrequenter Phänomene untersuchen.

Frequenz	Token	Tabelle	Tabellengröße in MB
49.872.845	,	TOKENTAB_KOMMA	956
44.230.812	.	TOKENTAB_PUNKT	848
24.561.941	*der*	TOKENTAB_DER	473
19.379.044	*und*	TOKENTAB_UND	374
18.540.671	*die*	TOKENTAB_DIE	357
13.526.161	*in*	TOKENTAB_IN	262
12.882.070	)	TOKENTAB_KLAMMERZU	251
12.796.748	(	TOKENTAB_KLAMMERAUF	249
804.209.606	restliche Token	TOKENTAB_REST	21.286

Tab. 39: Segmentierung des Korpusinventars in separate Token-Tabellen

Der Speicherbedarf der 804.209.606 Tabellenzeilen von TOKENTAB_REST liegt bei knapp 21,3 GB, verteilt auf 681.168 Datenblöcke. Das kumulierte Datenvolumen von TOKENTAB_ REST und den acht Einzeltoken-Tabellen von knapp über 25 GB fällt nicht nennenswert geringer aus als das der ursprünglichen

Token-Tabelle (25,5 GB), obwohl die WORD-Spalte in den Einzeltoken-Tabellen fehlt. Analog stellt sich das Bild für die angelegten Indizes dar: Der REST_SID-Index benötigt aufgrund der geringeren Zeilenanzahl zwar weniger Speicherplatz (26.499, 1.875 MB) als der ursprünglicher WSID-Index und die Einzeltoken-Tabellen beanspruchen vergleichsweise wenig Volumen, weil nur je zwei Spalten indiziert werden. Trotzdem fällt die Indexsumme (31.537 MB) lediglich unwesentlich kleiner aus als die Größe des TOKENTAB-WSID (32.107 MB).

Index	Indizierte Spalten	Größe in MB	Clustering Factor	Index-tiefe	Blattknoten
KOMMA_SID	SID, ID	1.279	30.392	2	40.626
PUNKT_SID	SID, ID	1.135	26.965	2	36.036
DER_SID	SID, ID	632	15.000	2	20.016
UND_SID	SID, ID	500	11.849	2	15.797
DIE_SID	SID, ID	478	11.320	2	15.095
IN_SID	SID, ID	350	8.290	2	11.031
KLAMMERZU_SID	SID, ID	334	7.919	2	10.527
KLAMMERAUF_SID	SID, ID	331	7.867	2	10.457
REST_SID	WORD,SID,ID	26.499	452.972.763	2	825.275

Tab. 40: Physikalischer Indexaufbau für die segmentierten Token-Tabellen

Tabelle 41 und Abbildung 28 stellen die gemittelten Abfragezeiten der acht separierten hochfrequenten Phänomene (gewonnen durch `select count(sid) from tokentab_komma` etc.) denen der unter Nutzung der ursprünglichen Token-Tabelle gewonnenen gegenüber.[119] Wiederum führen wir jeweils fünf Testläufen auf einem CPU-Kern ohne Buffercache-Nutzung durch. Die Ergebnisse belegen einen unverkennbar positiven Effekt: Die physische Segmentierung führt im Schnitt zu einer Beschleunigung der Einzeltoken-Abfragen um ca. 28,8 Prozent.

Tabelle	,	.	*der*	*und*	*die*	*in*	)	(
TOKENTAB	15,04	13,85	7,54	6,21	5,97	5,57	4,55	4,20
Einzeltoken-Tabellen	10,09	9,07	5,65	4,63	4,30	3,83	3,69	3,54

Tab. 41: Mittelwerte der Hochfrequenz-Abfragen in Sekunden

[119] Die TOKENTAB-Abfrage nach *der* bestätigt im Übrigen unsere Messung in Abschnitt 3.3.1.

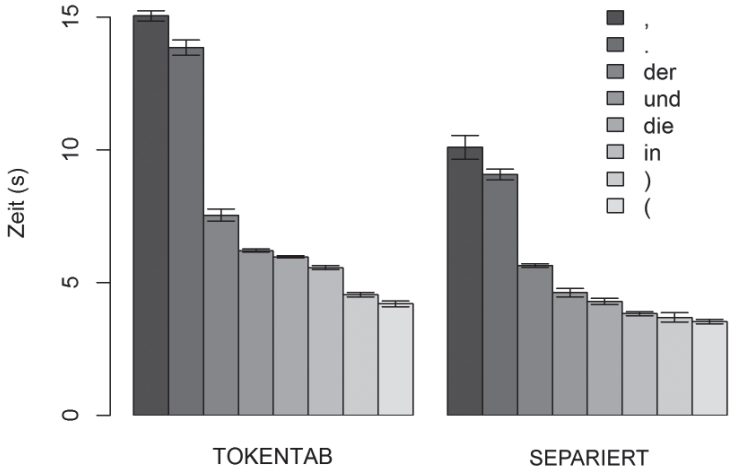


Abb. 28: Abfragezeiten der Hochfrequenz-Abfragen

Interessant ist weiterhin ein Blick auf die Auswirkungen der Zeilenreduktion von TOKENTAB_REST um ca. 20 Prozent auf die Abfragezeiten für Mono-, Bi- und Trigramme. Diese werden gemittelt in den Tabellen 42 bis 44 sowie in Abbildung 29 dokumentiert. Die Werte der ursprünglichen TOKENTAB stammen dabei aus Abschnitt 3.3.1; als Wert für die Monogramm-Häufigkeitsklasse HK-1 (Token *der*) wurde die Abfragezeit von TOKENTAB_DER eingetragen. Die Monogramm-Abfragen auf dem segmentierten Datenbestand führen im Schnitt zu einer Beschleunigung um 24,6 Prozent. Für Bigramme liegt der positive Effekt bei durchschnittlich 19,5 Prozent, für Trigramme noch bei 13,7 Prozent.

Tabelle	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
TOKENTAB	7,54	1,91	0,30	0,08	0,06	0,06	0,04
TOKENTAB_REST	5,65	1,53	0,21	0,07	0,04	0,02	0,01

Tab. 42: Mittelwerte der Monogramme in Sekunden

Tabelle	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
TOKENTAB	14,67	7,33	5,47	4,13	1,84	0,15	0,14
TOKENTAB_REST	11,71	6,20	4,08	3,63	1,25	0,14	0,15

Tab. 43: Mittelwerte der Bigramme in Sekunden

Tabelle	HK-1	HK-2	HK-3	HK-4	HK-5	HK-6	HK-7
TOKENTAB	8,42	6,51	5,59	4,13	3,46	4,22	2,03
TOKENTAB_REST	7,63	6,06	5,10	2,92	2,23	3,77	1,95

Tab. 44: Mittelwerte der Trigramme in Sekunden

Zusammenfassend lässt sich festhalten, dass die physische Auslagerung hochfrequenter Phänomene in separate Tabellen eine durchgehend vorteilhafte Entscheidung für das Design von Korpusdatenbanken mit authentischem Sprachmaterial darstellt. Daneben erscheint sie aufgrund der relativ wenigen betroffenen Types – im untersuchten Testkorpus sind es derer acht – auch praktisch handhabbar.

Die Interpretation der 80-20-Regel auf den Datenbestand führt zu einer Beschleunigung um beinahe 30 Prozent für ein geringes, aber mutmaßlich oft verwendetes Phänomeninventar. Für die übrigen Abfragen ist eine ebenfalls signifikante, mit Zunahme der Tabellenverknüpfungen bei Bi- und Trigrammen prozentual abnehmende Verbesserung erkennbar. Ob im alltäglichen Einsatz tatsächlich eine 80-20-Verteilung, eine weniger umfangreiche 90-10-Segmentierung oder sogar die Auslagerung weiterer Phänomene optimalere Ergebnisse generiert, bleibt Forschungsgegenstand individueller Testreihen. Vorstellbar ist im Übrigen die Anwendung des Segmentierungsgedankens auf weitere Inhaltstypen einer Korpusdatenbank, etwa auf Wortklassen-Angaben. Hier könnte an Stelle der Indizierung tokenbezogener POS-Werte (zur deren Verteilung im Korpus siehe Tab. 11 bis 14) eine Aufteilung in separate POS-Tabellen treten.

Anzumerken bleibt ein Hinweis auf potenzielle Nachteile der physischen Segmentierung: Wortformrecherchen mit Platzhalteroperatoren (z.B. im Suchmuster `der%`) oder unter Einbeziehung verschiedener Groß- und Kleinschreibungsvarianten (z.B. Suche nach `der` und `Der`) erfordern bei diesem Vorgehen zusätzliche Datenbankoperationen, da sowohl die Einzeltoken-Tabelle als auch die reduzierte Token-Tabelle abgefragt werden müssen. Gegebenenfalls wäre hierfür das Anlegen tabellenübergreifender Indizes eine Option, die allerdings nicht für alle Datenbanksysteme und Indextypen realisierbar ist und zudem wiederum das zu verwaltende Indexvolumen erhöhen würde.

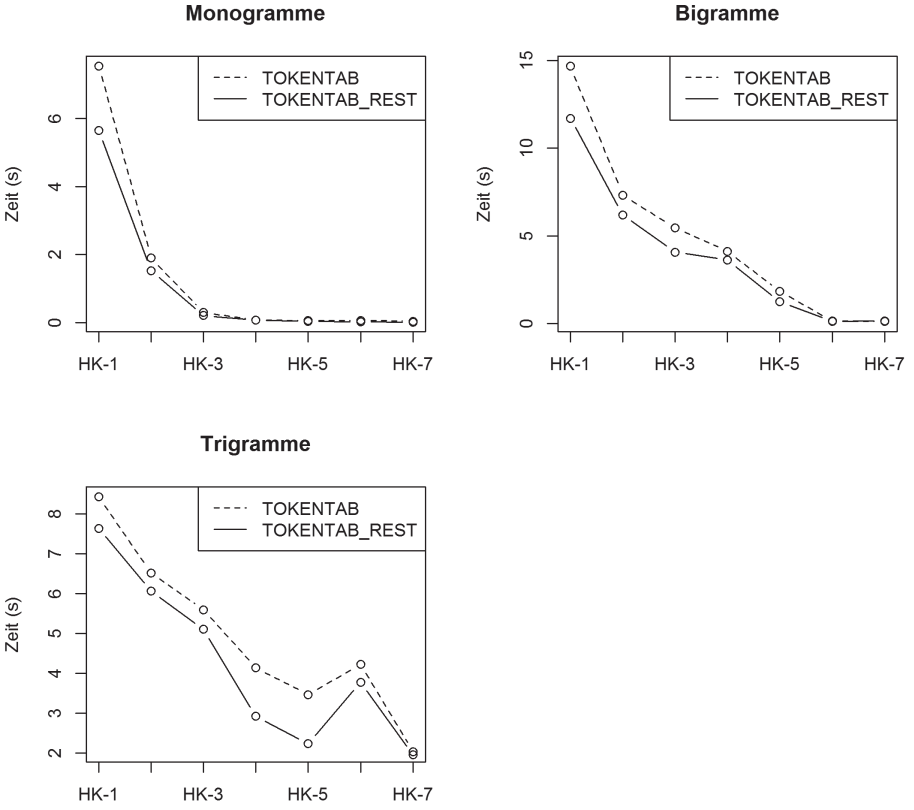


Abb. 29: Vergleich zwischen ursprünglicher und reduzierter Token-Tabelle

3.3.5 Fazit

Die durchgeführten Testreihen dokumentieren den Einfluss ausgewählter Designentscheidungen auf die Leistungsfähigkeit von Korpusdatenbanken, ausgedrückt durch Laufzeiten für linguistisch motivierte Recherchen. Im Einzelnen lassen sich folgende Tendenzen und Effekte nachweisen:

– Die tokenorientierte Relationierung von Korpusinhalten eröffnet im Vergleich zu n-Gramm-Tabellen deutlich umfangreichere Abfragemöglichen, wobei zum Auffinden aufeinander folgender Phänomene kombinierte Indizes – bestehend aus Wortform, Wort-ID und Satz-ID – die besten Ergebnisse liefern. Die Hinzunahme einer Wort-ID beeinträchtigt die Indexleistung dabei nicht. Für Recherchen mit mehr als zwei Suchattributen besteht dagegen Optimierungsbedarf hinsichtlich des Abfragemodells.

– Platzhalteroperatoren und reguläre Ausdrücke lassen sich effektiv nutzen, sofern Indizes angelegt werden, die gegebenenfalls zielführende Zusatzeinschränkungen unterstützen – etwa funktionale Reverse-Indizes.
– Die Nutzung rein numerischer Indizes bietet aufgrund der damit verbundenen zusätzlichen Tabellenverknüpfungen keine Vorteile hinsichtlich der Laufzeit.
– Die Abfrage hochfrequenter Phänomene profitiert von einer initialen physischen Segmentierung bzw. Isolierung gemäß des Pareto-Prinzips.

In der korpuslinguistischen Praxis nimmt, zumeist im Anschluss an das Auffinden der auf eine Phänomenbeschreibung passenden Belege, der Aspekt der Kontextgenerierung eine gewichtige Stellung ein. Besteht, wie in unserem Modell, das Ergebnis einer Korpusabfrage aus einer Satznummernliste, dann erfordert die nachfolgende Exploration in einem separaten Schritt die Ermittlung daran anknüpfender Informationen. Hierzu zählen in erster Linie eine geordnete Ausgabe sämtlicher Satzwörter sowie linguistisch relevanter Sekundärdaten (Lemma, Wortklasse usw). Auch dabei gilt es, selektionsoptimierte Indizes mit in passender Reihenfolgen indizierten Tabellenspalten vorzuhalten, also beispielsweise Strukturen mit der Satz-ID als erster oder alleiniger Indexspalte.

Im nachfolgenden Kapitel 4 soll unsere Referenzumgebung um Korpusinhalte variabler Größe ergänzt werden. Weiterhin wird ein leistungsstärkeres Hardwaresystem eingeführt, um Skalierungseffekte bei gleichermaßen zunehmender Korpusgröße und Anzahl einsatzfähiger CPU-Kerne messen zu können. Dabei kommt unser evaluiertes Datenmodell mit WSID-Indizes zum Einsatz.

4. Evaluation des Anforderungskatalogs

Der in Kapitel 2 vorgestellte Anforderungskatalog linguistisch motivierter Korpusabfragen umfasst zehn für die korpuslinguistische Praxis prototypische Phänomenbeschreibungen, formuliert unter Heranziehung unterschiedlicher Primär- und Sekundärdatentypen. Nachfolgend ziehen wir diesen Katalog zur empirischen Evaluation unseres eingeführten Referenzsystems und zur Offenlegung von Optimierungspotenzial heran. Dabei kommt das in Abschnitt 3.2.3 beschriebene relationale Datenmodell mit der zentralen Text- und Metadatentabelle TB_TEXT sowie den auf Wortebene dreifach annotierten Belegsammlungen zum Einsatz.

Das Untersuchungsziel liegt in der Aufdeckung von Zusammenhängen zwischen wachsenden Suchraum-Datenmengen, Tabellenverknüpfungen, Belegzahlen und Retrievalzeiten. Aus diesem Grund umfasst unser Evaluationsdatenbestand folgende Teilkorpora:

- Evaluationskorpus 1 (EK-1) mit 1 Million (10^6) Textwörtern (entspricht der Größenordnung von Korpora der ersten Generation, z.B. dem Brown-Korpus)
- Evaluationskorpus 2 (EK-2) mit 100 Millionen (10^8) Textwörtern (entspricht Korpora der zweiten Generation wie z.B. dem BNC)
- Evaluationskorpus 3 (EK-3) mit 1 Milliarde (10^9) Textwörtern (entspricht Korpora der dritten Generation)
- Evaluationskorpus 4 (EK-4) mit 2 Milliarden Textwörtern
- Evaluationskorpus 5 (EK-5) mit 4 Milliarden Textwörtern
- Evaluationskorpus 6 (EK-6) mit 8 Milliarden Textwörtern

Das umfangreichste Evaluationskorpus EK-6 umfasst dabei das in Kapitel 3 vorgestellte Gesamtkorpus, genauer gesagt das Untersuchungskorpus (UK) sowie Teile des Reservekorpus (RK) für die Aufstockung der Textwörteranzahl auf 8 Milliarden. Die kleineren Evaluationskorpora (EK-5 bis EK-1) wurden satzweise sukzessive per Zufallsauswahl aus dem Bestand der jeweils nächstgrößeren Kompilation generiert. Eine Ausnahme bildet die Zusammenstellung der Xerox-Variante von EK-6: Da das Gesamtkorpus zwar vollständig mit Connexor und TreeTagger analysiert wurde, aber nur gut zur Hälfte Xerox-annotiert vorliegt, konstituieren diese Annotationen die Xerox-Variante von EK-5. Xerox-EK-6 wurde durch einmalige Duplizierung der EK-5-Datenmenge generiert. Dies erscheint – trotz der mit der lexikalischen Vielfalt

verbundenen (höheren) Indexdichte bzw. (abnehmenden) Selektivität[120] – für unsere Auswertung als vertretbar, nicht zuletzt weil exemplarische Vergleichsabfragen zwischen den drei EK-6-Varianten keinerlei signifikante Abweichungen der Abfragezeiten aufweisen.

Indiziert werden auf Wortebene Token, Lemma oder Wortklasse gemeinsam mit Wort- und Satznummern in kombinierten Spaltenindizes. Für Token und Lemmata existieren darüber hinaus Reverse-Indizes für Suchanfragen mit Links-Trunkierung bzw. regulären Ausdrücken. Tabelle 45 gibt einen Einblick in den Speicherbedarf der Teilkorpora; Tabelle 46 ergänzt Angaben zum physischen Aufbau ausgewählter Indizes.[121]

Korpus	Zeilen/Wörter	Größe in MB	Blöcke
EK-1	1.000.000	34,45	1.116
EK-2	100.000.000	3.666,125	117.318
EK-3	1.000.000.000	38.370,563	1.227.867
EK-4	2.000.000.000	78.084,438	2.498.709
EK-5	4.000.000.000	158.544,125	5.071.785
EK-6	8.000.000.000	319.318,575	10.218.210

Tab. 45: Tabellengrößen der EK-Worttabellen (Connexor-Variante)

[120] Die für die Referenzabfragen wesentlichen kombinierten Indizes beinhalten Lemma- bzw. Tokenspalte an jeweils erster Position. Durch das Kopieren des Xerox-EK5-Inventars umfassen die Indizes des vom Volumen her doppelt so großen Korpus Xerox-EK-6 keinerlei neuen Lemma- oder Tokentypes, sondern ausschließlich zusätzliche Zeiger auf bestehende Werte. Indexdichte und -selektivität als Ausdruck der Verhältnisse zwischen eindeutigen Wertausprägungen und deren Gesamtzahl – in der Linguistik als *Type-Token Ratio* (kurz *TTR*) bekannt – ändern sich dadurch signifikant. Allerdings lässt sich eine vergleichbare, wenngleich weniger extreme Tendenz auch bei den beiden anderen Korpusvarianten (Connexor, TreeTagger) nach der Halbierung der Datenmenge von EK-6 auf EK-5 durch Zufallsauswahl beobachten. Dies entspricht der Beobachtung von Richards (1987), „dass das Type-Token-Verhältnis von der Korpusgröße abhängt: Wenn sonst alles gleich bleibt, nimmt der TTR-Wert mit steigender Korpusgröße ab“ (Perkuhn et al. 2012, S. E6-3).

[121] Die Angaben in Tabelle 45 beziehen sich ausschließlich auf die durch das Einlesen der Connexor-Standoff-Annotationen generierten Korpusdaten auf Wortebene. Tabelle 46 dokumentiert exemplarisch die Indizes für Connexor-Token. Lemma- und Wortklassenwerte sowie das „Shredding“ der TreeTagger- und Xerox-Annotationen konstituieren diverse zusätzliche, in Volumen und Aufbau jeweils vergleichbare Objekte. Darüber hinaus enthält das verwendete Datenbankschema Tabellen und Indizes für weitere Relationen, z.B. auf Wortgruppen-, Satz- und Textebene.

Index	Größe in MB	Clustering Factor	Index-tiefe	Blatt-knoten
TOKENTAB1_WSID	29,875	437.051	1	914
TOKENTAB1_WSID_REV	29,875	437.246	1	914
TOKENTAB2_WSID	3.093,75	41.063.896	2	103.589
TOKENTAB2_WSID_REV	3.093,813	40.035.070	2	95.145
TOKENTAB3_WSID	32.059,688	423.881.082	2	1.025.240
TOKENTAB3_WSID_REV	32.059,688	425.121.665	2	993.090
TOKENTAB4_WSID	65.017,625	957.851.352	2	2.053.849
TOKENTAB4_WSID_REV	65.017,375	981.458.722	2	2.082.820
TOKENTAB5_WSID	131.588,438	2.029.805.801	3	3.951.491
TOKENTAB5_WSID_REV	131.588,375	2.125.626.460	3	4.167.758
TOKENTAB6_WSID	264.737,625	4.314.542.648	3	8.457.043
TOKENTAB6_WSID_REV	264.737,563	4.410.356.331	3	8.643.062

Tab. 46: Physikalischer Aufbau der EK-Token-Indizes (Connexor-Variante)

Dem drastischen Anstieg der maximal zu verarbeitenden Datenmengen entsprechen wir mit der Einbeziehung einer zweiten, leistungsstärkeren Hardwareplattform (vertikale Skalierung). Neben dem ursprünglichen Referenzsystem kommt ein ebenfalls CentOS-basiertes Skalierungssystem zum Einsatz, auf dem sämtliche Korpusinhalte und -strukturen dupliziert vorliegen; vgl. Tabelle 47. Laufzeiten für einzelne Abfragen werden jeweils auf beiden Systemen gemessen, die Ergebnisse einander gegenüber gestellt. Auf diese Weise entsteht ein aussagekräftiger Evaluationsrahmen für die Verifizierung von Performanz und Skalierungsverhalten relational organisierter, datenintensiver Korpusretrievalsysteme.

Parameter	Referenzsystem	Skalierungssystem
CPU	1 Quadcore-Prozessor Intel i5 mit 2,67GHz Taktung (ca. 5.300 bogomips)	16 virtuelle CPUs AMD Opteron 8.356 mit 2,33GHz Taktung (ca. 5.700 bogomips)
RAM	16 GB	32 GB

Tab. 47: Hardware-Parameter von Referenz- und Skalierungssystem

Eine weitere Modifikation im Vergleich zu Kapitel 3 betrifft das Inventar der verwendeten SQL-Befehle. Während wir uns bislang ausschließlich auf Vorkommen zählende SELECT-Statements beschränkt haben, werden die gefundenen Belegnummern nachfolgend via INSERT[122]-Statement in temporäre Tabellenstrukturen eingefügt. Damit tragen wir der korpuslinguistischen Praxis Rechnung, dass Belege typischerweise nicht nur gezählt werden, sondern auch für weiterführende Aktionen zur Verfügung stehen sollen. Hierzu gehören statistische Analysen unter Einbeziehung zusätzlicher Metadaten ebenso wie das Auffinden und Anzeigen variabler Kontexte. Die Abfragen im vorliegenden Kapitel speichern aus diesen Gründen sämtliche eindeutigen Satznummern der recherchierten Phänomenbelege in einer Ergebnistabelle (RESTAB) ab.

Für jedes Evaluationskorpus EK-1 bis EK-6 kommen je fünf identische Queries sowohl auf dem Referenz- wie auf dem Skalierungssystem zur Ausführung, insgesamt also jeweils 6 × 5 × 2 = 60 SQL-Statements pro Referenzkatalog-Abfrage. Vor jeder Suche werden die Ergebnistabelle sowie der Buffer Cache der Datenbankinstanz explizit geleert.

4.1 Abfrage 1: Einfaches Suchmuster

Das Suchmuster der ersten Abfrage des Referenzkatalogs besteht aus dem Textwort „dabei“, nach dem unter Beachtung von Groß- und Kleinschreibung recherchiert werden soll. Durchsucht werden ausschließlich die Connexor-basierten Tokentabellen bzw. deren WSID-Indizes mit folgendem SQL-Statement:

```
insert into RESTAB
select unique CO_SENTENCEID
from <EK-WORTTABELLE>
where CO_TOKEN='dabei';
```

Beispielbeleg: *Die verbreitete Zuversicht dürfte **dabei** den Obstproduzenten gut getan haben*. [Textsigle A00/MAI.35415]

Abbildung 30 mit Konfidenzintervallen sowie Tabelle 48 dokumentieren die Abfragezeiten der jeweiligen INSERT-Statements. Ein Vergleich der unterschiedlich umfangreichen Evaluationskorpora ergibt, dass die Suchzeiten für anwachsende Tokenzahlen günstig skalieren. Die Steigerung fällt zwar stetig,

[122] Dabei verwenden wir durchgehend den APPEND-Hint zur Beförderung von direkten Einfügungen (*direct path inserts*) am Tabellenende sowie den NOLOGGING-Hint zur Vermeidung von Redo-Log-Einträgen. Beides beschleunigt das Befüllen der Ergebnistabelle.

allerdings durchgehend unterproportional aus. In Tabelle 49 werden die Steigerungsfaktoren einander gegenübergestellt: Während etwa EK-2 im Vergleich zu EK-1 ein um den Faktor 100 umfangreicheres Volumen sowie eine um den Faktor 105 höhere Belegzahl aufweist, wächst die entsprechende Abfragezeit lediglich mit einem Faktor < 2. Und auch bei sehr großen Datenvolumen – etwa EK-6 versus EK-5 – lässt sich eine unterproportionale Verlängerung der Abfragezeiten feststellen.

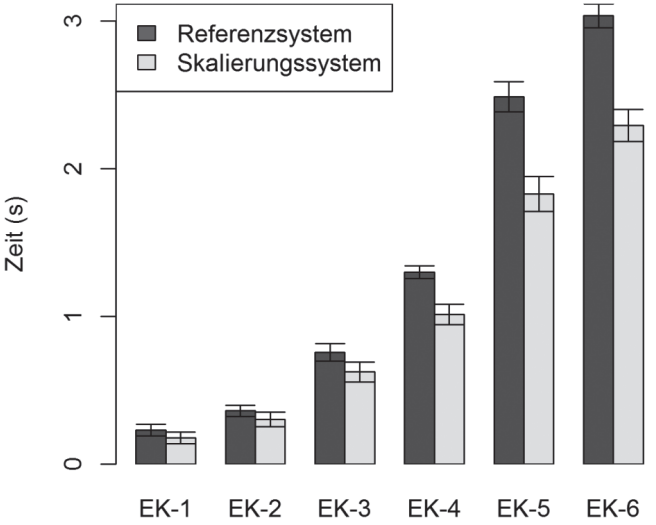


Abb. 30: Abfragezeiten für Abfrage 1 auf dem Referenz- und Skalierungssystem

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	357	0,23	0,18
EK-2	37.604	0,36	0,31
EK-3	400.005	0,76	0,62
EK-4	792.485	1,30	1,02
EK-5	1.616.680	2,49	1,83
EK-6	3.063.192	3,04	2,29

Tab. 48: Mittelwerte der Abfragezeiten für Abfrage 1 in Sekunden

Die unterschiedliche Hardware-Ausstattung auf Referenz- und Skalierungssystem fällt für das einfache Suchmuster der Abfrage hingegen nur marginal ins Gewicht. Bei kleineren Datenmengen sind kaum nennenswerte Unterschiede zwischen beiden Plattformen feststellbar, anschaulich visualisiert durch die Überschneidung der Konfidenzintervalle in Abbildung 30. Der vergrößerte Hauptspeicher sowie die zusätzlichen CPU-Kerne schlagen sich in diesen Fällen nicht in angemessen kürzeren Abfragezeiten nieder. Zwar ist eine stetige Verbesserung erkennbar, allerdings auf deutlich niedrigerem Niveau als es die Erhöhung der Prozessorkernanzahl nahelegen würde.

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 105,33	Faktor 1,57	Faktor 1,72
EK-3	Faktor 10	Faktor 10,64	Faktor 2,11	Faktor 2
EK-4	Faktor 2	Faktor 1,98	Faktor 1,71	Faktor 1,64
EK-5	Faktor 2	Faktor 2,04	Faktor 1,92	Faktor 1,79
EK-6	Faktor 2	Faktor 1,89	Faktor 1,22	Faktor 1,25

Tab. 49: Steigerungsfaktoren für Abfrage 1

4.2 Abfrage 2: Suffixsuche mit Platzhalterzeichen

Die zweite Abfrage des Referenzkatalogs verwendet wiederum ein einzelnes diskretes Element, das diesmal allerdings ein Platzhalterzeichen für Links-Trunkierung enthält. Das Suchmuster „*bezogen“ (in SQL-Syntax `'%bezogen'`) passt auf sämtliche Wortformen, die auf *-bezogen* enden. Recherchiert wird in den Connexor-basierten Korpustabellen bzw. per Index Range Scan in den Reverse-Indizes (WSID_REV) mit folgendem SQL-Statement:

```
insert into RESTAB
select unique CO_SENTENCEID
from <EK-WORTTABELLE>
where reverse(CO_TOKEN) like reverse('%bezogen');
```

Beispielbeleg: *Geprobt wird **projektbezogen**, also immer auf ein Konzert oder einen Auftritt hin.* [Textsigle M09/SEP.71468]

Die ermittelten Belegzahlen fallen niedriger als bei der ersten Katalogabfrage aus. Ein Vergleich der Laufzeiten (Abb. 31 und Tab. 50, Steigerungsfaktoren in

Tab. 51) zeigt dessen ungeachtet ähnliche Tendenzen wie die Wildcard-freie Recherche. Die Suchzeiten liegen durchgehend unter einer Sekunde und skalieren für ansteigende Korpusvolumen unverkennbar unterproportional. Dies wird besonders für die kleineren Evaluationskorpora deutlich. Aber auch bei Korpusgrößen im Milliarden-Token-Bereich und Volumensteigerungen um den Faktor 2 verlängern sich die Abfragezeiten stets maximal um den Faktor 1,7. Die Performanzsteigerungen auf dem Skalierungssystem durch zusätzlichen Hauptspeicher und weitere CPU-Kerne fallen zwar abermals vergleichsweise gering aus, allerdings diesmal ohne jede Überschneidung der Konfidenzintervalle und damit durchgehend signifikant.

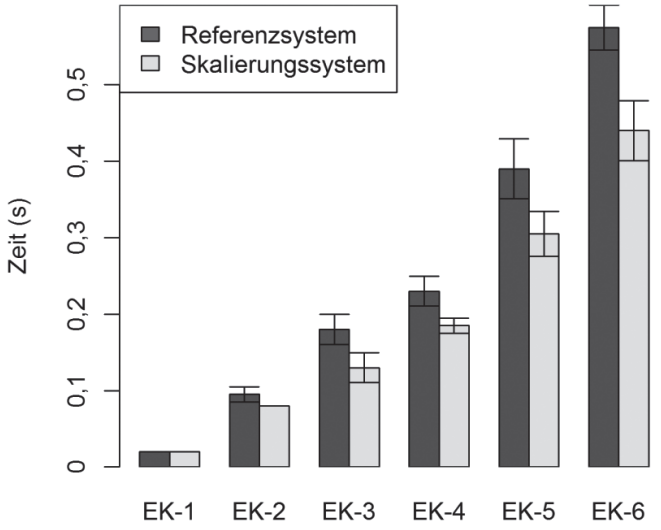


Abb. 31: Abfragezeiten für Abfrage 2 auf dem Referenz- und Skalierungssystem

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	31	0,02	0,02
EK-2	4.961	0,10	0,08
EK-3	42.705	0,18	0,13
EK-4	69.826	0,23	0,18
EK-5	134.790	0,39	0,30
EK-6	270.758	0,57	0,44

Tab. 50: Mittelwerte der Abfragezeiten für Abfrage 2 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 160,03	Faktor 5	Faktor 4
EK-3	Faktor 10	Faktor 8,61	Faktor 1,8	Faktor 1,63
EK-4	Faktor 2	Faktor 1,64	Faktor 1,28	Faktor 1,38
EK-5	Faktor 2	Faktor 1,93	Faktor 1,7	Faktor 1,67
EK-6	Faktor 2	Faktor 2,01	Faktor 1,46	Faktor 1,47

Tab. 51: Steigerungsfaktoren für Abfrage 2

4.3 Abfrage 3: Komplexes Relativsatz-Muster

Bei der Formulierung der dritten Referenzkatalog-Abfrage kommt erstmals ein komplexes Suchmuster in Form einer linearen Verkettung mehrerer Einzelelemente über deren relative Position zum Einsatz. Gesucht wird in Connexor-annotierten Subkorpora nach Relativsätzen mit einleitendem *was* an Stelle eines Relativpronomens. Dabei soll das Lemma *das*[123] am Satzanfang stehen, unmittelbar oder mit maximal einem Zwischenwort gefolgt von einem Nomen (Connexor-Wortklasse „N"), das wiederum unmittelbar vor einer Kombination aus Komma und *was* steht.

Beispielbeleg: ***Das** einzige **Argument, was** ich immer wieder höre, ist kein Argument, sondern ein Werturteil*. [Textsigle WDD13/B68.03108]

Insgesamt enthält das benötigte SQL-Statement mithin fünf Suchkriterien sowie Angaben zu deren Position im Belegsatz, realisiert durch relationale Verknüpfungen:

```
insert into RESTAB

select unique T1.CO_SENTENCEID

from <EK-WORTTABELLE> T1, <EK-SATZTABELLE> T2, <EK-WORTTABELLE>
T3, <EK-WORTTABELLE> T4, <EK-WORTTABELLE> T5

where T1.CO_LEMMA = 'das' and T3.CO_POS = 'N' and T4.CO_TOKEN =
',' and T5.CO_TOKEN = 'was'

and
T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_FIRST-
WORDID
```

[123] Durch die Verwendung der Lemmatisierung werden nicht nur Groß- und Kleinschreibungen des Artikels „das" gefunden, sondern z.B. auch „dem" oder „dessen".

```
and T1.CO_SENTENCEID = T3.CO_SENTENCEID and T1.CO_ID<T3.CO_ID and
T1.CO_ID>T3.CO_ID-3
and T3.CO_SENTENCEID = T4.CO_SENTENCEID and T3.CO_ID = T4.CO_ID-1
and T4.CO_SENTENCEID = T5.CO_SENTENCEID and T4.CO_ID = T5.CO_ID-1;
```

Die mit mehreren einschränkenden Kriterien versehene Relativsatzabfrage unterstreicht den Nutzen umfangreicher Korpora für die Recherche nach seltenen Phänomenen. Während das 8 Milliarden-Korpus zumindest eine vierstellige Anzahl von Belegen enthält, findet sich im kleinsten Evaluationskorpus trotz der isoliert betrachtet hochfrequenten Einzelkriterien gerade noch genau ein passender Satz. Selbst das vom Umfang her dem BNC entsprechende Evaluationskorpus 2 liefert lediglich 33 Belege und stellt damit kaum empirisch aussagekräftiges Datenmaterial für weitergehende linguistische Analysen dar.

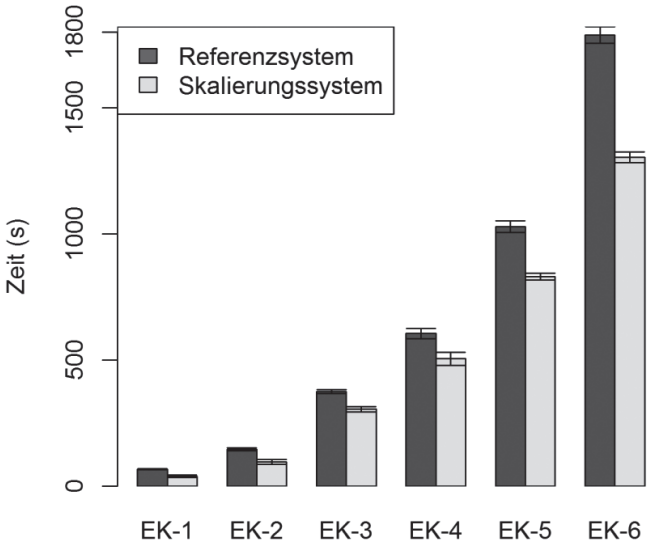


Abb. 32: Abfragezeiten für Abfrage 3 auf dem Referenz- und Skalierungssystem

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	1	68,28	39,62
EK-2	24	147,18	97,31
EK-3	212	376,24	305,78
EK-4	411	605,75	505,66
EK-5	780	1.028,99	831,84
EK-6	1.412	1.789,19	1.304,23

Tab. 52: Mittelwerte der Abfragezeiten für Abfrage 3 in Sekunden

Die Abfragezeiten skalieren wie gewohnt positiv, d.h. durchgehend unterproportional (Abb. 32 und Tab. 52, Steigerungsfaktoren in Tab. 53). Trotzdem wird erstmals während unserer Referenzkatalog-Evaluierung deutlich, dass die simple Verkettung mehrerer Suchkriterien auf authentischem Sprachmaterial rasch lange Wartezeiten generiert. Auf dem Referenzsystem dauert die Abfrage von EK-6 trotz bestmöglicher Indexnutzung eine halbe Stunde, und selbst die Suche auf dem Skalierungssystem terminiert durchschnittlich erst nach über zwanzig Minuten. Damit ist die Relativsatzsuche ein anschaulicher erster Präzedenzfall für den Bedarf an abfragespezifischen Optimierungen, die in Kapitel 5 vorgestellt werden sollen.

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 24	Faktor 2,16	Faktor 2,46
EK-3	Faktor 10	Faktor 8,83	Faktor 2,56	Faktor 3,14
EK-4	Faktor 2	Faktor 1,94	Faktor 1,61	Faktor 1,65
EK-5	Faktor 2	Faktor 1,9	Faktor 1,7	Faktor 1,65
EK-6	Faktor 2	Faktor 1,81	Faktor 1,74	Faktor 1,57

Tab. 53: Steigerungsfaktoren für Abfrage 3

4.4 Abfrage 4: ACI-Konstruktionen

Das in Abfrage 4 spezifizierte Suchmuster stellt eine Schablone für die Suche nach ausgewählten ACI-Konstruktionen im Deutschen mit zwei benachbarten Verbformen dar. Erstmals werden neben eindeutig benannten auch beliebige Wortabstände einbezogen. Auf die Schablone passen Sätze, in denen ein Infinitiv (Connexor-Subkategorie „INF“) unmittelbar vor einem Wahrnehmungsverb steht, dessen Lemma entweder *hören, sehen, spüren, fühlen* oder *riechen* lautet. Nach dieser Sequenz folgt wiederum unmittelbar ein Satzende-Punkt. Als zusätzliche Einschränkung sollen im selben Belegsatz mit beliebigem Abstand vor der Verbkombination ein Pluralnomen (Connexor-Subkategorie „PL“) sowie das Lemma *haben* ohne dazwischen stehendes Trennwort vorkommen.

Beispielbeleg: ***Hausbewohner hatten*** *auch am Montagnachmittag die Frau und einen Mann in der Wohnung* ***streiten gehört.*** [Textsigle PNN13/FEB.00627]

Das ausformulierte SQL-Statement kombiniert sechs Suchkriterien sowie deren Satz- bzw. Positionsspezifikationen:

```
insert into RESTAB

select unique T1.CO_SENTENCEID

from <EK-WORTTABELLE> T1, <EK- WORTTABELLE> T2, <EK-WORTTABELLE> T3,
<EK-WORTTABELLE> T4, <EK-WORTTABELLE> T5, <EK-SATZTABELLE> T6

where T1.CO_SUB = 'PL' and T2.LEMMA = 'haben' and T3.CO_SUB = 'INF'
and T4.CO_LEMMA in ('hören', 'sehen', 'spüren', 'fühlen', 'riechen')
and T5.CO_TOKEN = '.'

and T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_ID-1

and T2.CO_SENTENCEID = T3.CO_SENTENCEID and T2.CO_ID < T3.CO_ID

and T3.CO_SENTENCEID = T4.CO_SENTENCEID and T3.CO_ID = T4.CO_ID-1

and T4.CO_SENTENCEID = T5.CO_SENTENCEID and T4.CO_ID = T5.CO_ID-1

and
T5.CO_SENTENCEID = T6.CO_SENTENCEID and T5.CO_ID = T6.CO_LASTWORDID;
```

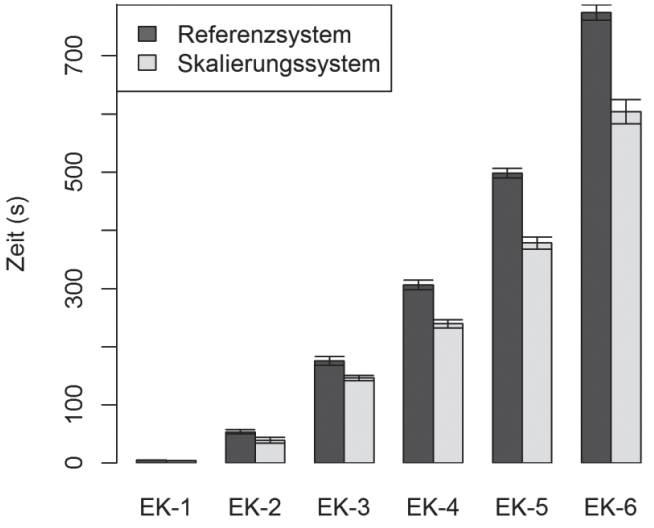


Abb. 33: Abfragezeiten für Abfrage 4 auf dem Referenz- und Skalierungssystem

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	0	5,46	4,17
EK-2	12	54	39,58
EK-3	137	175,75	146,38
EK-4	271	306,36	239,55
EK-5	583	498,43	378,26
EK-6	1.107	774,29	603,91

Tab. 54: Mittelwerte der Abfragezeiten für Abfrage 4 in Sekunden

In noch offensichtlicherem Ausmaß als bereits Abfrage 3 verdeutlicht die ACI-Recherche die unzureichende empirische Aussagekraft kleinerer Textkorpora für verallgemeinernde sprachwissenschaftliche Untersuchungen. Das Subkorpus EK-2 mit immerhin 100 Millionen Textwörtern liefert gerade einmal 12 passende Belege, EK-1 scheidet als Belegquelle sogar komplett aus.

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	N/A	Faktor 9,9	Faktor 9,5
EK-3	Faktor 10	Faktor 11,41	Faktor 3,25	Faktor 3,7
EK-4	Faktor 2	Faktor 1,98	Faktor 1,74	Faktor 1,64
EK-5	Faktor 2	Faktor 2,15	Faktor 1,63	Faktor 1,6
EK-6	Faktor 2	Faktor 1,9	Faktor 1,55	Faktor 1,6

Tab. 55: Steigerungsfaktoren für Abfrage 4

Die ermittelten Belegzahlen steigern sich von diesem sehr niedrigen Niveau aus mit teils über- und teils unterproportionalen Faktorwerten. Die Entwicklung der Abfragezeiten (Abb. 33 und Tab. 54, Steigerungsfaktoren in Tab. 55) bleibt positiv in dem Sinne, dass deren Steigerungsfaktoren ausnahmslos hinter denen von Token und Belegen zurückbleiben. Verbesserungswürdig fallen die absoluten Retrievalzeiten aus: Sowohl auf dem Referenz- wie auf dem Skalierungssystem messen wir maximale Wartezeiten von über zehn Minuten und identifizieren damit einen weiteren Kandidaten für die Optimierung des eingesetzten „all in one"- Suchalgorithmus, der sämtliche Suchkriterien in einem Abfragestatement zusammenfasst.

4.5 Abfrage 5: W-Fragen ohne Verb

Katalogabfrage 5 dient als dritte und letzte Repräsentantin von Recherchen, die ausschließlich unter Rückgriff auf die lineare Anordnung nach Kombinationen von Wortform-bezogenen Angaben fahnden. Gesucht wird in den mit TreeTagger annotierten Textsammlungen nach W-Fragen ohne Verb. In linguistischen Kategorien ausgedrückt: Passende Konstruktionen enthalten ein adverbiales Interrogativpronomen (TreeTagger-Wortklasse „PWAV") am Satzanfang sowie mit beliebigem Abstand ein Fragezeichen (Token *?*), jedoch kein dazwischen stehendes Verb (TreeTagger-Wortklasse „VRB"). „VRB" bezeichnet dabei eine kumulierte Klasse mehrerer STTS-Annotationselemente.[124]

Beispielbeleg: ***Warum*** *kein geeintes Deutschland****?*** [Textsigle WKD/MOD.12716]

Das dazu passende SQL-Statement verknüpft vier Suchkriterien:

[124] Zusammengefasst in „VRB" werden sämtliche Verb-Annotationen (finite Formen, Infinitive, Partizip Perfekt) des STTS-Tagsets: „VMFIN", „VAFIN", „VVFIN", „VAIMP", „VVIMP", „VVINF", „VAINF", „VMINF", „VVIZU", „VVPP", „VMPP" sowie „VAPP".

```
insert into RESTAB

select unique T1.CO_SENTENCEID

from <EK-WORTTABELLE> T1, <EK-SATZTABELLE> T2, <EK-WORTTABELLE> T4

where T1.CO_POS = 'PWAV' and T4.CO_TOKEN = '?'

a                                   n                                   d
T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_FIRSTWORD-
ID

and not exists (select null from <EK-WORTTABELLE> T3 where T3.CO_POS
= 'VRB' and T1.CO_SENTENCEID = T3.CO_SENTENCEID and T1.CO_ID < T3.
CO_ID)

and T1.CO_SENTENCEID = T4.CO_SENTENCEID and T1.CO_ID < T4.CO_ID;
```

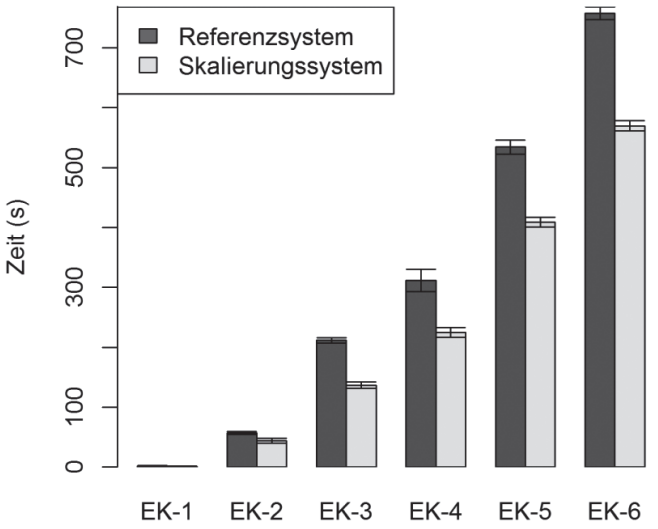


Abb. 34: Abfragezeiten für Abfrage 5 auf dem Referenz- und Skalierungssystem

Eine Besonderheit dieser Abfrage liegt, neben der Verwendung durchgehend hochfrequenter Suchkriterien (Satzanfang, Fragezeichen sowie zwei komplette Wortklassen), im erstmaligen Einsatz des NOT-Operators zum Ausschluss eines Phänomens (Vorhandensein der Wortklasse „Verb“). Aus empirischer Perspektive erfreulich entwickelt sich die Belegmenge: Die Anzahl passender Sätze liegt insgesamt unverkennbar höher als bei den vorhergehenden komplexen Recherchen, wobei die Steigerung der Treffer speziell zwischen EK-1 und EK-2 sowie zwischen EK-2 und EK-3 unterproportional ausfällt.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	3	2,36	1,31
EK-2	125	56,78	44,33
EK-3	738	211,7	136,91
EK-4	1.277	311,99	225,27
EK-5	2.595	534,48	409,27
EK-6	4.811	757,75	569,79

Tab. 56: Mittelwerte der Abfragezeiten für Abfrage 5 in Sekunden

Trotz hochfrequenter Suchkriterien bzw. vergleichsweiser hoher Trefferzahlen terminieren die Recherche-Statements deutlich rascher als bei Abfrage 3 und immer noch leicht schneller als bei Abfrage 4. Dabei macht sich die diesmal niedrigere Anzahl von nur vier Suchkriterien sicherlich positiv bemerkbar. Die maximalen Wartezeiten von deutlich über 10 Minuten für die größten Evaluationskorpora (vgl. Abb. 34 und Tab. 56, Steigerungsfaktoren in Tab. 57) klassifizieren nichtsdestoweniger auch Abfrage 5 als Anwärter für Optimierungsversuche.

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 41,7	Faktor 24,06	Faktor 33,84
EK-3	Faktor 10	Faktor 5,9	Faktor 3,73	Faktor 3,09
EK-4	Faktor 2	Faktor 1,73	Faktor 1,47	Faktor 1,65
EK-5	Faktor 2	Faktor 2,03	Faktor 1,71	Faktor 1,82
EK-6	Faktor 2	Faktor 1,85	Faktor 1,42	Faktor 1,39

Tab. 57: Steigerungsfaktoren für Abfrage 5

4.6 Abfrage 6: Movierung in virtuellen Subkorpora

Abfrage 6 erweitert die Suche nach linear angeordneten Wortkombinationen um eine thematische Beschränkung sowie um eine Eingrenzung hinsichtlich des Publikationsdatums. Damit greift sie nicht nur auf wortbezogene Sprachannotationen, sondern auch auf textbezogene außersprachliche Metadaten zu. Recherchiert wird nach movierten Anredeformen zur expliziten Kennzeichnung des grammatischen Genus (weiblich) in seit dem Jahr 2000 entstandenen Sprachquellen, die thematisch der Domäne „Politik/Wirtschaft/Gesellschaft“ zugeordnet sind. Zur Identifikation der Movierung dient die Kombination des Tokens *Frau* mit einem unmittelbar nachfolgenden Nomen (Connexor-Wortklasse „N“), das auf das Suffix *-in* endet und kein Eigenname (Connexor-Wortklasse „N Prop“) sein darf (um Kombinationen wie „Frau Hein“ auszufiltern).

Beispielbeleg: *„**Frau Kapitänin**, meine Herren“, grüßt Grobien jedes Mal sein Publikum, wenn er als Meister der Zeremonie etwas ansagen muss*. [Textsigle B04/FEB.11042]

Die passenden SQL-Statements umfassen insgesamt sechs Suchkriterien, darunter einen Ausdruck mit Platzhalterzeichen sowie extralinguistische Spezifikationen:

```
insert into RESTAB

select unique T1.CO_SENTENCEID

from <EK-WORTTABELLE> T1, <EK-WORTTABELLE> T2, <EK-WORTTABELLE> T3

where T1.CO_TOKEN = 'Frau' and reverse(T2.CO_TOKEN) like re-
verse('%in') and T3.CO_POS = 'N'

and T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_ID-1

and T2.CO_SENTENCEID = T3.CO_SENTENCEID and T2.CO_ID = T3.CO_ID

and not exists (select null from <EK-WORTTABELLE> T4 where T4.CO_
SUB = 'Prop' and T3.CO_SENTENCEID = T4.CO_SENTENCEID and T3.CO_ID
= T4.CO_ID)

and T1.CO_SENTENCEID in (select CO_SENTENCEID from <EK-SATZTABEL-
LE> T5 where T5.CO_TEXTID in (select CO_TEXTID from TB_TEXT T6
where T6.CO_DOMAIN = 4 and T6.CO_DATE >= 2000));
```

Die T4-Subabfrage zum Ausschluss von Eigennamen kombiniert Subkategorie (CO_SUB) und Satz-/Wort-IDs, da diese – analog zur Wortklassenindexierung – in einem gemeinsamen Index abgebildet sind. Für die übrigen Token- und Wortklassen-Suchwerte verwenden wir unsere eingeführten WSID- bzw. Reverse-Indizes. Erstmals abgefragt werden Metadaten aus der übergeord-

neten Texttabelle TB_TEXT. Die hierfür erforderlichen zusätzlichen Verknüpfungen greifen auf entsprechend implementierte Spaltenindizes zurück.

Auffallend sind die überproportionalen Zuwächse der gefundenen Belege für Subkorpus EK-2 im Vergleich zu EK-1 sowie für EK-3 im Vergleich zu EK-2. Den Steigerungen der Tokenzahlen um die Faktoren 100 bzw. 10 stehen um die Faktoren 327 bzw. knapp 17 angestiegene Trefferzahlen gegenüber. Dies kann als zufälliger Ausreißer für kleine Korpusvolumen abgebucht werden, eventuell spielen auch unterschiedliche Stratifikationsmerkmale der zufällig generierten Textsammlungen eine Rolle: In den kleinsten Evaluationskorpora weisen die Metadatenausprägungen für thematische Domäne und Publikationsdatum andere Verteilungen auf als bei den größeren Textsammlungen. Da dieser Umstand unser primäres Untersuchungsinteresse – Entwicklung von Abfragezeiten für abgestufte Korpusvolumen, unterschiedlich komplexe Suchausdrücke und variable Belegmengen – aber nicht maßgeblich berührt, verzichten wir an dieser Stelle auf eine weiterführende Interpretation. [125]

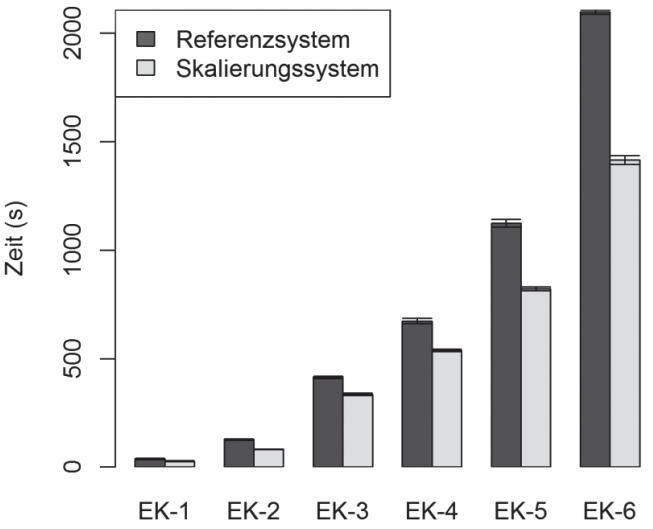


Abb. 35: Abfragezeiten für Abfrage 6 auf dem Referenz- und Skalierungssystem

[125] Im Großen und Ganzen korrespondieren die Belegzahlen sämtlicher Katalogsabfragen für wechselnde Korpusgrößen erwartungsgemäß mit den jeweiligen Gesamt-Tokenzahlen. Zur UK-Stratifizierung, d.h. der Verteilung der außersprachlichen Metadaten in den 8-Milliarden-Korpora, vgl. Kapitel 2.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	7	38,23	27,39
EK-2	2.291	126,77	82,37
EK-3	38.783	414,18	335,56
EK-4	81.005	672,86	538,38
EK-5	145.733	1.123,26	820,71
EK-6	281.129	2.096,78	1.415,85

Tab. 58: Mittelwerte der Abfragezeiten für Abfrage 6 in Sekunden

Auf die Retrievalzeiten der einzelnen Evaluationskorpora (Abb. 35 und Tab. 58, Steigerungsfaktoren in Tab. 59) wirken sich die überproportionalen Belegzuwächse vergleichsweise gering aus. Die entsprechenden Steigerungsfaktoren liegen hier für Referenz- und Skalierungssystem abermals deutlich unter den Steigerungsfaktoren für Korpusvolumen und Fundstellen. Zur Optimierung der absolut betrachtet langen Abfragezeiten, die für umfangreiche Korpora wie schon bei Abfrage 3 bei über einer halben Stunde liegen, verweisen wir auf Kapitel 5.

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 327,29	Faktor 3,32	Faktor 3,01
EK-3	Faktor 10	Faktor 16,93	Faktor 3,27	Faktor 4,07
EK-4	Faktor 2	Faktor 2,09	Faktor 1,62	Faktor 1,6
EK-5	Faktor 2	Faktor 1,8	Faktor 1,67	Faktor 1,52
EK-6	Faktor 2	Faktor 1,93	Faktor 1,87	Faktor 1,73

Tab. 59: Steigerungsfaktoren für Abfrage 6

4.7 Abfrage 7: Genitivobjekte

Ebenso wie die vorige Abfrage kombiniert auch unser siebtes Beispiel einzelwortbezogene Kriterien mit hierarchisch übergeordneten Annotationsmerkmalen, diesmal aus dem Datenmaterial des Xerox-Parsers. Gesucht wird nach Sätzen mit Genitivobjekten (Xerox-Dependenz „OBJ GEN“) zum Lemma *erfreuen*.

Beispielbeleg: *Sie **erfreuen sich ihrer Beliebtheit** zu Recht.* [Textsigle B00/JUL.59402]

Ergänzend zu den in Abschnitt 2.4 dargestellten Besonderheiten der Xerox-Dependenzanalyse nachfolgend ein Beispiel für die gesuchte Standoff-Annotation:

```
<DEPENDENCY name="OBJ" fts="GEN">
  <PARAMETER ind="0" num="4" word="erfreuen"/>
  <PARAMETER ind="1" num="10" word="Beliebtheit"/>
</DEPENDENCY>
```

Das passende Abfragestatement kommt mit nur einer Verknüpfung aus, greift aber erstmals auf die in Abschnitt 3.2.3 dokumentierte Dependenztabelle zu:

```
insert into RESTAB
select unique T1.CO_SENTENCEID
from <EK-WORTTABELLE> T1, <EK-DEPENDENZTABELLE> T2
where T1.CO_LEMMA = 'erfreuen'
and T2.CO_NAME = 'OBJ' and T2.CO_FTS = 'GEN'
and T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_PARA0;
```

Obwohl diesmal ein vergleichsweise einfaches Suchmuster mit nur zwei Suchattributen (Lemma und Objekttyp) vorliegt, findet sich im kleinsten Evaluationskorpus EK-1 zum wiederholten Mal keine einzige Fundstelle für das Kombinationsphänomen. Davon abgesehen bewegen sich die Belegzahlen in etwa auf dem Niveau von Abfrage 5, allerdings mit deutlich abweichenden Suchzeiten (Abb. 36 und Tab. 60, Steigerungsfaktoren in Tab. 61). Diese liegen für Abfrage 7 zum Teil massiv unter denen der bisherigen komplexen Abfragen 3 bis 6. Den Volumina der Dependenztabellen dürfte dies nicht geschuldet sein, da diese durchweg in etwa denen der Worttabellen entsprechen – nicht jedes Wort ist Bestandteil einer kodierten Dependenzrelation, aber viele Dependenzrelationen umfassen mehrere Wörter und einige Wörter gehören

zu mehreren Dependenzrelationen. Vielmehr schlägt hier, wie bereits in Kapitel 3 belegt, positiv zu Buche, dass die Korpusabfragen nur je eine SQL-Verknüpfung beinhalten.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	0	3,21	2,21
EK-2	70	11,46	9,2
EK-3	866	17,98	13,39
EK-4	1.512	32,73	20,77
EK-5	2.863	55,81	35,17
EK-6	5.726	98,94	58,94

Tab. 60: Mittelwerte der Abfragezeiten für Abfrage 7 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	N/A	Faktor 3,57	Faktor 4,16
EK-3	Faktor 10	Faktor 12,37	Faktor 1,57	Faktor 1,46
EK-4	Faktor 2	Faktor 1,75	Faktor 1,82	Faktor 1,55
EK-5	Faktor 2	Faktor 1,89	Faktor 1,71	Faktor 1,69
EK-6	Faktor 2	Faktor 2	Faktor 1,77	Faktor 1,68

Tab. 61: Steigerungsfaktoren für Abfrage 7

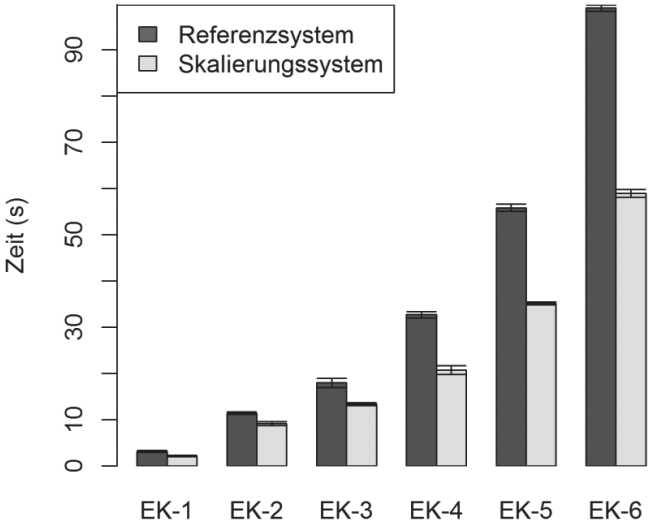


Abb. 36: Abfragezeiten für Abfrage 7 auf dem Referenz- und Skalierungssystem

4.8 Abfrage 8: Partizipialphrase vor niederfrequentem Nomen

Abfrage 8 kombiniert wortspezifische Kriterien (Lemma, Wortklasse, Häufigkeitsklasse) mit einer hierarchischen Suchspezifikation. Ermittelt werden Partizipialphrasen mit einem aus dem Verb *sehen* gebildeten Adjektiv (Partizip I) oder einer als Adjektiv gebrauchten Verbform (Partizip II; beides Xerox-Wortklasse „ADJ“) innerhalb einer Adjektivphrase. Auf diese soll unmittelbar ein niederfrequentes Nomen (Frequenzklasse > 9) folgen.

Beispielbeleg: *Auch hier gibt es schöne, bisher nicht* ***gesehene Töne***. [Textsigle M96/602.05744]

Die Konstruktion der Abfrage ist einer Besonderheit der Xerox-Annotationen geschuldet: „XIP vergibt im Gegensatz zu den weiteren NLP-Tools für Partizipien als Lemmaform fast immer die Infinitivform des Verbs, die sich durch das POS-Tag ADJ von der verbalen Lemmaform unterscheidet.“ (Stadler 2014, S. 23). Zur Illustration dieses Umstands hier die (gekürzte) XML-Standoff-Annotation des Beispielsatzes vor dem Import in die Korpusdatenbank:

```
<NODE num="23" tag="AP" start="19913955" end="19913976">
  <NODE num="12" tag="ADV" start="19913955" end="19913961">
    <TOKEN pos="ADV" start="19913955" end="19913961" surface="bisher">
      <READING lemma="bisher" pos="ADV"/>
```

```
    </TOKEN>
  </NODE>
  <NODE num="14" tag="NEGAT" start="19913962" end="19913967">
    <TOKEN      pos="NEGAT"      start="19913962"      end="19913967"
surface="nicht">
      <READING lemma="nicht" pos="NEGAT"/>
    </TOKEN>
  </NODE>
  <NODE num="16" tag="ADJ" start="19913968" end="19913976">
    <TOKEN      pos="ADJ"      start="19913968"      end="19913976"
surface="gesehene">
      <READING lemma="sehen" pos="ADJ"/>
    </TOKEN>
  </NODE>
</NODE>
<NODE num="18" tag="NOUN" start="19913977" end="19913981">
    <TOKEN      pos="NOUN"      start="19913977"      end="19913981"
surface="Töne">
      <READING lemma="Ton" pos="NOUN"/>
    </TOKEN>
</NODE>
```

Da kein kombinierter 4-Spalten-Index aus Satz-ID, Wort-ID, Lemma und Wortklasse zur Recherche nach *sehen*-Lemmata mit der Wortklasse „ADJ“ existiert, teilen wir diesen Suchausdruck in die beiden Subanfragen T1 (Wortklasse) und T2 (Lemma) auf. Die Zugehörigkeit zu einer AP prüfen wir mittels der Xerox-Knotentabelle, die Häufigkeitsklasse ist in der Lemmaliste hinterlegt. Abfragen für das aus insgesamt fünf Bestandteilen aufgebaute Suchmuster formulieren wir folgendermaßen:

```
insert into RESTAB
select unique T1.CO_SENTENCEID
from <EK-WORTTABELLE> T1, <EK-WORTTABELLE> T2, <EK-KNOTENTABELLE>
T3, <EK-WORTTABELLE> T4, <EK-LEMMALISTE> T5
where T1.CO_POS = 'ADJ'
and T2.CO_LEMMA = 'sehen' and T1.CO_SENTENCEID = T2.CO_SENTENCEID
and T1.CO_ID = T2.CO_ID
and T3.CO_PARENT = 'AP' and T2.CO_SENTENCEID = T3.CO_SENTENCEID and
T2.CO_ID = T3.CO_ID
```

```
and T4.CO_POS = 'NOUN' and T3.CO_SENTENCEID = T4.CO_SENTENCEID and
T3.CO_ID + 1 = T4.CO_ID
and T5.CO_LEMMA = T4.CO_LEMMA and T5.CO_FREQCLASS > 9;
```

Aus der Modellierperspektive ist diese Abfrage ein prototypisches Beispiel für den Einfluss von Tabellen- und Indexdesign auf die Suchzeiten. Diese würden mit einem kombinierten Index aus Satz-ID, Wort-ID, Lemma und Wortklasse vermutlich signifikant niedriger ausfallen. Ein solcher Index ginge indes mit einer ebenfalls nicht unerheblichen Steigerung des Speichervolumens einher und verhielte sich darüber hinaus suboptimal bei Abfragen, die Lemma und Wortklasse für unterschiedliche Textwörter recherchieren sollen. Optional könnte über eine physische Trennung der Daten nach Häufigkeitsklassen nachgedacht werden, um Verknüpfungen mit der Lemmaliste (T5) zu umgehen. Zur Überprüfung der prinzipiellen Eignung und Skalierfähigkeit unseres Modells erscheint dies allerdings nicht notwendig, deshalb bleiben wir beim eingeführten Design.

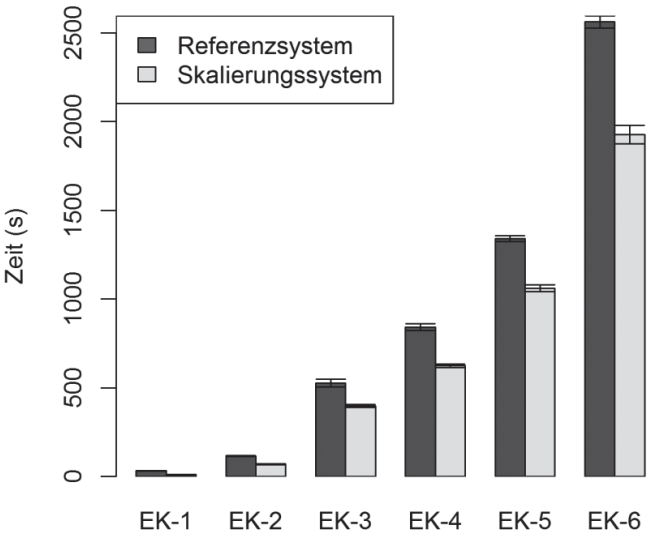


Abb. 37: Abfragezeiten für Abfrage 8 auf dem Referenz- und Skalierungssystem

Insgesamt bestätigen die Ergebnisse (Abb. 37 und Tab. 62, Steigerungsfaktoren in Tab. 63) die wesentlichen Tendenzen der bisherigen Katalogabfragen, nämlich eine unterproportionale Erhöhung der Abfragezeiten für ansteigende Korpusgrößen und Belegzahlen. Sowohl die minimalen als auch die maximalen Suchzeiten bewegen sich – angesichts der datenintensiven Verknüpfungen wenig verwunderlich – auf vergleichsweise hohem Niveau. Damit benen-

nen wir Abfrage 8 als weiteren Kandidaten für den Versuch einer Optimierung des Suchalgorithmus in Kapitel 5.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	2	33,3	11,27
EK-2	131	116,03	70,26
EK-3	1.294	526,3	398,04
EK-4	2.541	841,86	624,55
EK-5	5.621	1.341,21	1.061,35
EK-6	11.242	2.561,27	1.927,31

Tab. 62: Mittelwerte der Abfragezeiten für Abfrage 8 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	65,5	Faktor 3,48	Faktor 6,23
EK-3	Faktor 10	Faktor 9,88	Faktor 4,54	Faktor 5,67
EK-4	Faktor 2	Faktor 1,96	Faktor 1,6	Faktor 1,57
EK-5	Faktor 2	Faktor 2,21	Faktor 1,59	Faktor 1,7
EK-6	Faktor 2	Faktor 2	Faktor 1,91	Faktor 1,82

Tab. 63: Steigerungsfaktoren für Abfrage 8

4.9 Abfrage 9: Regulärer Ausdruck mit Rechts-Trunkierung

Abfrage 9 des Referenzkatalogs nimmt die in Abschnitt 3.3.2 thematisierte Recherche mit Platzhalteroperatoren bzw. regulären Ausdrücken wieder auf. Gesucht wird nach mit dem Substring „Will" beginnenden Straßennamen, realisiert als Aneinanderreihung mit mindestens zwei Durchkopplungsbindestrichen. Als Suchmuster verwenden wir: *Will*.+\-.+\-(*Straße*|*Weg*|*Platz*|*Allee*)$.

Passende Token enthalten also nach dem vorgegebenen Wortanfang zwei Sequenzen aus je mehreren beliebigen Zeichen (.) und einem (maskierten) Bindestrich, gefolgt von einem der vier Bezeichner „Straße", „Weg", „Platz" oder „Allee" am Wortende.

Beispielbeleg: *Mehr als 100 Beamte riegelten deshalb am Vormittag das Gelände an der **Willy-Brandt-Straße** ab*. [Textsigle B00/DEZ.01586]

Wir haben bereits gezeigt, dass für derartige reguläre Ausdrücke eine in das SQL-Statement aufgenommene zusätzliche Einschränkung durch Rechts-Trunkierung mit Platzhalteroperator vorteilhaft ist. Mit dieser Strategie gestaltet sich die Abfrage wie folgt:

```
insert into RESTAB
select unique T1.CO_SENTENCEID
from <EK-WORTTABELLE> T1
where T1.CO_TOKEN like 'Will%'
and REGEXP_LIKE (T1.CO_TOKEN,
'Will.+\-.+\-(Straße|Weg|Platz|Allee)$');
```

Das vorgeschaltete Prädikat „Will%“ ermöglicht die Nutzung des kombinierten Token-/SatzID-/WortID-Index und beschleunigt die Suche enorm. Im 8-Milliarden-Korpus EK-6 beispielsweise werden damit 1.412.768 Treffer ermittelt, auf die dann die REGEXP_LIKE-Filterung zur Anwendung kommt. Die finalen Belegzahlen fallen aufgrund des restriktiven regulären Ausdrucks natürlich geringer aus. Im kleinsten Evaluationskorpus finden sich keinerlei Treffer, allerdings steigen die Belegzahlen anschließend durchgehend überproportional zum Korpuswachstum.

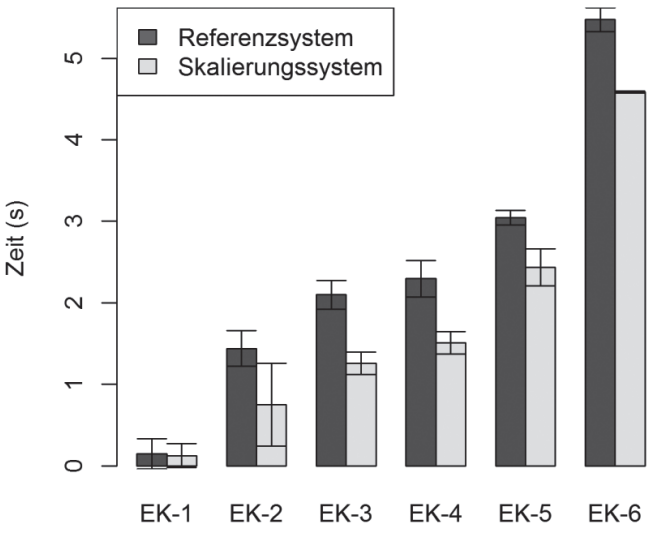


Abb. 38: Abfragezeiten für Abfrage 9 auf dem Referenz- und Skalierungssystem

Die Suchzeiten (Abb. 38 und Tab. 64, Steigerungsfaktoren in Tab. 65) gestalten sich durch die Indexnutzung erfreulich kurz und skalieren hinsichtlich des sukzessiv anwachsenden Korpusvolumens unterproportional. Insbesondere bei kleineren Evaluationskorpora fallen die geringen Steigerungsfaktoren auf. Der zeitliche Unterschied zwischen Referenz- und Skalierungssystem ist wie schon bei den vorigen Abfragen relativ gering, für EK-1 und EK-2 dokumentieren die überlappenden Konfidenzintervalle sogar keinerlei signifikante Abweichung.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	0	0,14	0,12
EK-2	10	1,44	0,75
EK-3	624	2,10	1,26
EK-4	1.317	2,29	1,51
EK-5	2.740	3,04	2,43
EK-6	5.693	5,47	4,58

Tab. 64: Mittelwerte der Abfragezeiten für Abfrage 9 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	N/A	Faktor 10,29	Faktor 6,25
EK-3	Faktor 10	Faktor 62,4	Faktor 1,46	Faktor 1,68
EK-4	Faktor 2	Faktor 2,11	Faktor 1,09	Faktor 1,2
EK-5	Faktor 2	Faktor 2,08	Faktor 1,33	Faktor 1,61
EK-6	Faktor 2	Faktor 2,08	Faktor 1,8	Faktor 1,88

Tab. 65: Steigerungsfaktoren für Abfrage 9

4.10 Abfrage 10: Regulärer Ausdruck mit Links-Trunkierung

Auch in Abfrage 10 kommt ein reguläres Suchmuster zum Einsatz, diesmal jedoch ohne festgelegtes Präfix. Recherchiert wird nach Internet-/Web-Domänen in Deutschland mit oder ohne explizite Protokollangabe: (*http:*\/\/)*?www*\..+*?*\.*de$*. Der maskierte Ausdruck „http://“ ist dementsprechend als optional mar-

kiert; passende Token müssen weiterhin den Servernamen „www“, eine beliebige Zeichensequenz als Subdomain sowie die Top-Level-Domain „de“ enthalten, jeweils mit regelgerechten Trennungspunkten.

Beispielbeleg: *Informationen finden sich im Internet unter: **www.uni-leipzig.de**.* [Textsigle B02/APR.25558]

Da an Stelle eines vorgegebenen Wortanfangs lediglich das Wortende bekannt ist, schränken wir die Suche mittels Links-Trunkierung ein und nutzen hierfür den Reverse-Index:

```
insert into RESTAB
select unique T1.CO_SENTENCEID
from <EK-WORTTABELLE> T1
where reverse(T1.CO_TOKEN) like reverse('%.de')
and REGEXP_LIKE (T1.CO_TOKEN, '(http:\/\/)?www\..+?\.de$');
```

Auf diese Weise wird ein Suchprädikat mit Platzhalteroperator vorgeschaltet, um den Index ansprechen zu können. Aus dem 8-Milliarden-Korpus EK-6 werden damit z.B. 1.278.586 Treffer für die REGEXP_LIKE-Filterung vorselektiert, dieser Wert liegt auf dem in Abfrage 9 erreichten Niveau. Die finalen Beleganzahlen fallen hingegen mit maximal 537.535 deutlich höher aus.

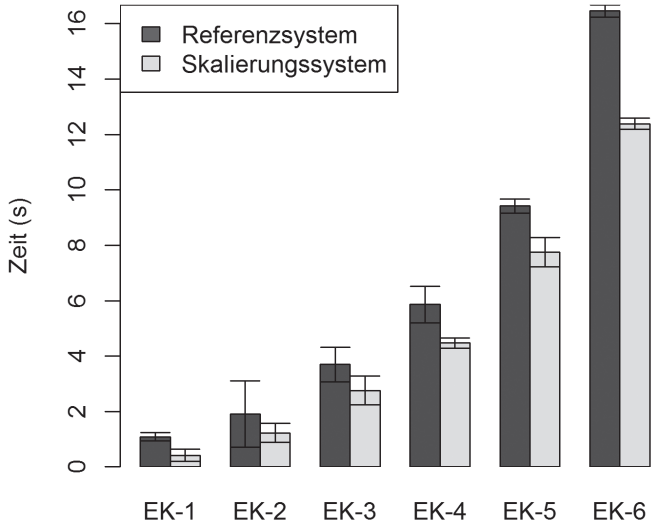


Abb. 39: Abfragezeiten für Abfrage 10 auf dem Referenz- und Skalierungssystem

Die gemessenen Suchzeiten liegen, trotz der ähnlich mächtigen Vorselektion, z.T. auffallend über denen der vorigen Abfrage. Hier spielen möglicherweise die optionalen Komponenten im regulären Ausdruck sowie das höhere Treffervolumen eine maßgebliche Rolle. Mit maximal 16 Sekunden auf dem Referenzsystem bewegen sich die Zeiten nichtsdestotrotz auch für umfangreiche Korpora in einem für die praktische Arbeit vertretbaren Bereich. Interessanter als diese absoluten Zahlen bleibt für uns ohnehin die relative Entwicklung der Suchzeiten im Verhältnis zu den Korpusgrößen: Hier ergibt sich das bereits bekannt positive Bild – auf beiden System fallen die Steigerungsraten eindeutig unterproportional aus.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	57	1,08	0,42
EK-2	5.926	1,91	1,23
EK-3	61.745	3,7	2,75
EK-4	137.702	5,86	4,47
EK-5	269.869	9,42	7,75
EK-6	537.535	16,45	12,38

Tab. 66: Mittelwerte der Abfragezeiten für Abfrage 10 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 103,96	Faktor 1,77	Faktor 2,93
EK-3	Faktor 10	Faktor 10,42	Faktor 1,94	Faktor 2,24
EK-4	Faktor 2	Faktor 2,23	Faktor 1,58	Faktor 1,63
EK-5	Faktor 2	Faktor 1,96	Faktor 1,61	Faktor 1,73
EK-6	Faktor 2	Faktor 1,99	Faktor 1,75	Faktor 1,6

Tab. 67: Steigerungsfaktoren für Abfrage 10

4.11 Einflussfaktoren auf die Abfrage-Laufzeiten

Ein übergeordnetes Erkenntnisinteresse im Zusammenhang mit der Evaluierung des Referenzkatalogs bestand – neben einer Bestätigung der grundsätzlichen Eignung relationaler Korpusdatenbanken für linguistisch motivierte Recherchen – in der Ausdifferenzierung elementarer Zusammenhänge: In welchem Ausmaß skalieren Suchzeiten auf dem Referenzsystem für unterschiedlich komplexe Korpusabfragen bei anwachsenden Korpusgrößen? Lässt sich eine regelgeleitete Steigerung der Suchzeiten in Abhängigkeit von Variablen wie Belegzahl oder Anzahl der Suchattribute – also der Anfrage-Komplexität – nachweisen? Und: In welchem Maße macht sich leistungsfähigere Hardware – auf unserem Skalierungssystems in erster Linie in Form zusätzlicher CPU-Kerne – positiv bemerkbar? Die nachfolgend diskutierten Antworten auf diese Fragen sollen zur Spezifizierung eines hinsichtlich der Eigenheiten natürlichsprachlicher Inhalte optimierten Abfragemodells beitragen.

Abfrageübergreifend steigt die Suchzeit (TIME) bei wachsenden Korpusgrößen (SIZE) streng monoton an; es gilt: aus SIZE1 < SIZE2 folgt TIME(SIZE1) < TIME(SIZE2). Gleichwohl fällt die Zunahme durchgängig unterproportional aus. Besonders augenfällig wird dies bei den kleineren Evaluationskorpora mit bis zu einer Milliarde Token. Abbildung 40 setzt die Steigerungsfaktoren von Tokenanzahl und Abfragezeit (mit Konfidenzintervallen zur Visualisierung der abfragespezifischen Intervallbreiten bei einem Konfidenzniveau von 95%) zueinander in Beziehung. Es wird deutlich, dass der relative Anstieg der Suchzeit stets unter dem des Korpusvolumens bleibt. Auf beiden Hardware-Plattformen entwickeln sich die Suchzeiten ähnlich unterproportional.

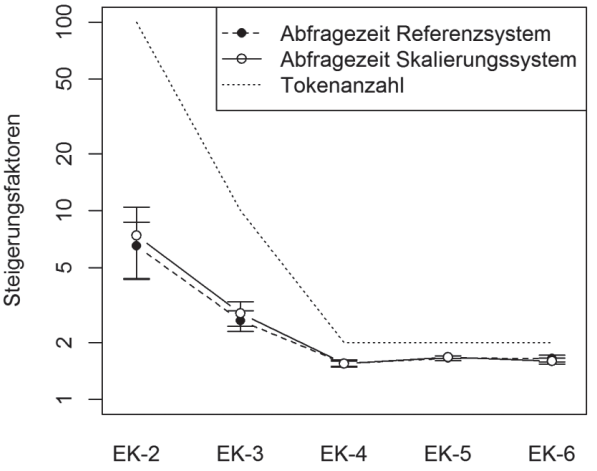


Abb. 40: Abfrageübergreifende Steigerungsfaktoren von Korpusgröße und Abfragezeit

4.11.1 Belegzahlen und Datenvolumen

Angesichts dieses für die korpuslinguistische Rechercheprаxis grundsätzlich günstigen Umstands stellt sich für eine weitere Laufzeitoptimierung die Frage nach den maßgeblich auf die Abfragedauer einwirkenden unabhängigen Variablen. Dabei scheint die ermittelte Belegzahl ein potenzieller Kandidat zu sein, schließlich korreliert üblicherweise mit einer größer ausfallenden Treffermenge auch ein erhöhter Verarbeitungsaufwand – etwa durch das Einfügen von Satznummern in Ergebnistabellen.

Abbildung 41 visualisiert die Zusammenhänge zwischen Belegzahlen und Abfragezeiten für die sechs unterschiedlich umfangreichen Evaluationskorpora. Graphen mit gestrichelten Linien repräsentieren die Werte des Referenzsystems, Graphen mit durchgezogenen Linien geben die Resultate des Skalierungssystems wieder. Interessanterweise legen die Plots keine konsistente funktionale Abhängigkeit nahe, sondern dokumentieren unregelmäßige Schwankungen: Gelegentlich geht eine Steigerung der Belegzahl mit einer Verlängerung der Suchzeit einher, ebenso häufig allerdings auch mit deren Verkürzung. Die niedrigsten Abfragezeiten korrespondieren sogar oftmals mit den jeweils höchsten Trefferzahlen.

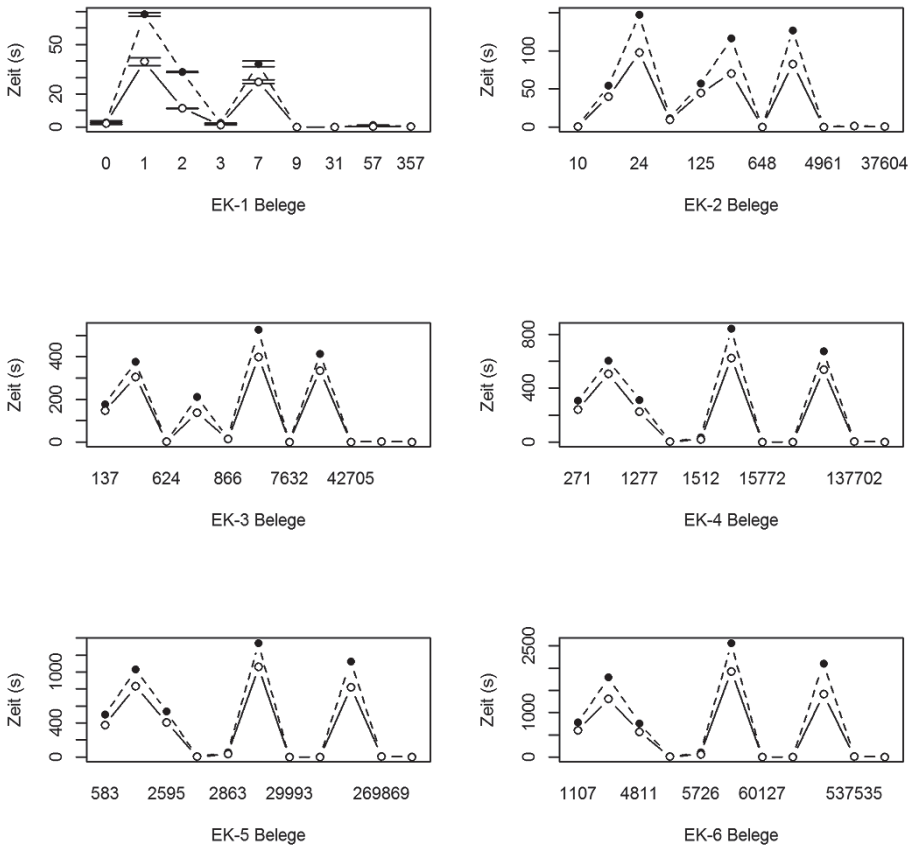


Abb. 41: Zusammenhang zwischen Belegzahlen und Abfragezeiten

Dieses uneinheitliche Bild ändert sich allenfalls, wenn wir an Stelle der ermittelten Belegsatzzahlen pro Abfrage die Summe aller durch die Einzelkriterien adressierten Datensätze betrachten. Plausibel scheint ein solcher Zusammenhang unter dem Gesichtspunkt des Rechenaufwands: Auch wenn die Schnittmenge derjenigen Korpussätze, auf die sämtliche Suchkriterien einer komplexen Abfrage gemeinsam zutreffen, letztlich klein ausfällt, müssen während der Suche doch ggf. beträchtlich mehr Datensätze selektiert und zueinander in Bezug gesetzt werden. Abbildung 42 visualisiert den Zusammenhang zwischen den Summen der zu einer Gesamtabfrage gehörenden Einzelkriteriumsfrequenzen und den Abfragezeiten für die fünf Katalogabfragen mit mehr als zwei Suchkriterien unter Heranziehung der jeweils maximalen Laufzeit für EK-6 auf dem Referenzsystem.

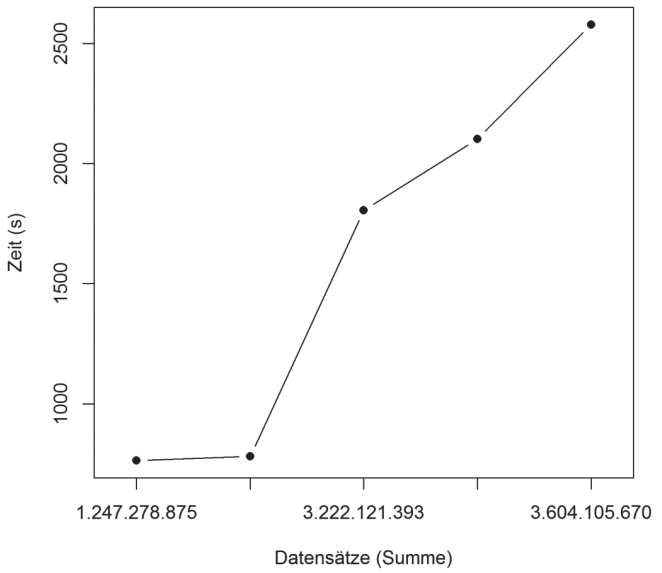


Abb. 42: Zusammenhang zwischen summierten Frequenzen von Suchkriterien und Abfragezeiten

Abfrage	Datensätze (Summe)	Datensätze (Durchschnitt)
3	3.222.121.393	644.424.279
4	1.429.942.915	238.323.819
5	1.247.278.875	311.819.718
6	3.309.693.272	551.615.545
8	3.604.105.670	720.821.134

Tab. 68: Anzahlen der durch Suchkriterien adressierten Datensätze für komplexe Abfragen

In eine ähnliche Richtung weist eine analoge Untersuchung mit Durchschnittswerten. Dabei teilen wir für jede Katalogabfrage die Summe der Einzelkriteriumsfrequenzen durch die Anzahl der Suchkriterien. Hohe Mittelwerte korrespondieren für die fünf analysierten Recherchen tendenziell mit vergleichsweise langen Laufzeiten; vgl. Tabelle 68 sowie Abbildung 43.

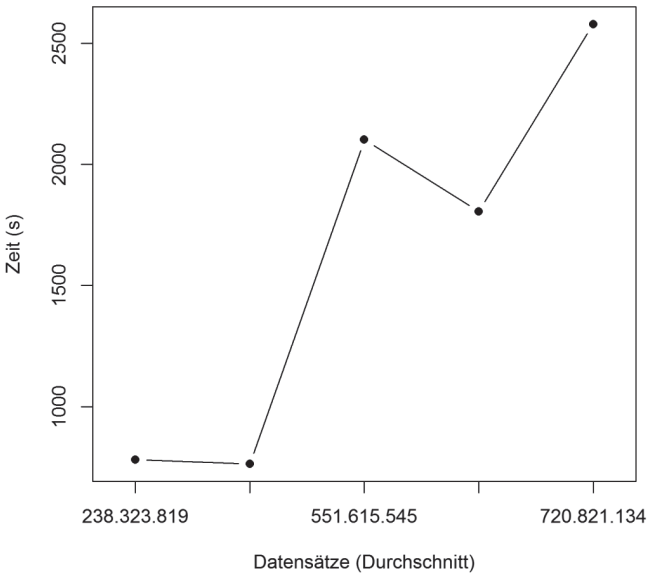


Abb. 43: Zusammenhang zwischen Durchschnittsfrequenzen von Suchkriterien und Abfragezeiten

4.11.2 Anzahl der Suchkriterien

Nachdem wenig auf finale Belegsatzzahlen als maßgebliche Einflussfaktoren hindeutet, könnte die Anzahl der Suchkriterien ein für weitere Analysen vielversprechender Wirkungsparameter sein. Vor diesem Hintergrund untersuchen wir nachfolgend Zusammenhänge zwischen Suchzeiten und der Komplexität sämtlicher evaluierten Queries. Hier erscheint bereits ein erster Blick auf die Visualisierung (Abb. 44, wiederum aufgeschlüsselt nach Korpusgröße und mit den Werten des Referenz- sowie des Skalierungssystems) unverkennbar aussagekräftiger als beim Wechselspiel zwischen Belegsatzzahlen und Suchzeiten.

Zwischen der Anzahl der Suchattribute und der zugeordneten Abfragedauer deutet sich ein zumindest partiell regelgeleiteter Zusammenhang ein: Die Abfragezeit (TIME) steigt bei zunehmender Komplexität des Suchausdrucks (COMPLEXITY) für bis zu fünf Suchkriterien streng monoton an; es gilt: aus COMPLEXITY 1 < COMPLEXITY 2 folgt TIME(COMPLEXITY 1) < TIME (COMPLEXITY 2). In diesem Bereich scheint sich eine Approximation durch eine lineare Funktion anzubieten. Einfache Suchmuster skalieren demnach effektiver als komplexe Suchmuster mit mehreren SQL-Verknüpfungen.

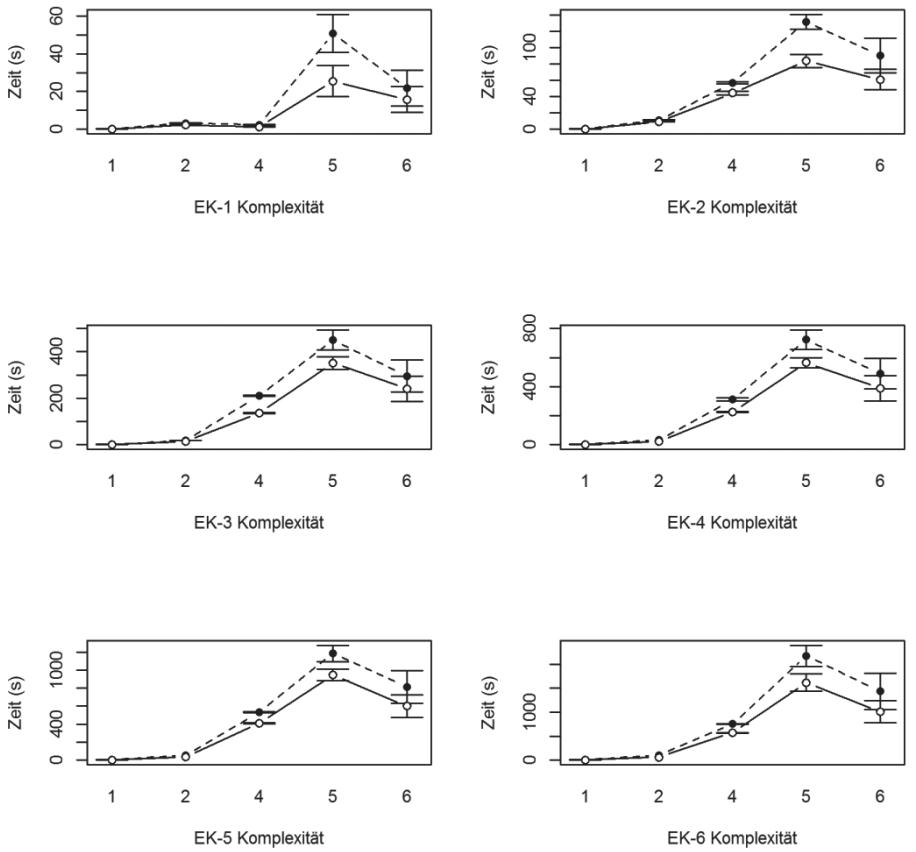


Abb. 44: Zusammenhang zwischen Komplexität des Suchausdrucks und Abfragezeiten

Schwieriger zu interpretieren erscheinen die Laufzeiten für Abfragen mit sechs Suchkriterien, die sich auf einem durchschnittlich niedrigeren Niveau als diejenigen für Abfragen mit fünf Suchkriterien bewegen. Dies mag zum einen mit individuellen Frequenz- und Verteilungsspezifika der im Referenzkatalog spezifizierten Phänomene zusammenhängen. Darüber hinaus weist dieser Umstand eventuell auf das Potenzial datenbankinterner Anfrageoptimierer hin, für die sich mit zunehmender Anzahl von Suchkriterien auch zusätzliche Optionen zur Berechnung alternativer Zugriffswege und effektiver Ausführungspläne auftun.

4.11.3 Modellierung der Abhängigkeiten

Um Effekte der unabhängigen Variablen FREQUENCY (Belegzahl) und COMPLEXITY (Komplexität des Suchausdrucks = Anzahl der Suchattribute) auf die abhängige Variable TIME (Abfragezeit) statistisch zu beschreiben, bietet sich der Rückgriff auf lineare Regressionsmodelle an. Aufgrund der sehr unterschiedlichen Werteausprägung (maximal mehrere hunderttausend Belege für FREQUENCY vs. maximal sechs Suchattribute für COMPLEXITY) testen wir kein gemeinsames Modell, sondern führen die Analysen separat für beide Variablen durch. Für die Berechnungen verwenden wir die Statistik-Software R und subsummieren sämtliche Messwerte im Datensatz *gesamt*.

In einem ersten Schritt berechnen wir ein lineares Regressionsmodell für TIME (Abfragezeit) und COMPLEXITY (Anzahl der Suchkriterien). Die R-Funktion *summary* liefert hierfür folgende detaillierte Auflistung:

```
Call:
lm(formula = toSeconds(as.character(gesamt$TIME)) ~ gesamt$COMPLEXITY)
Residuals:
        Min          1Q      Median          3Q             Max
    -590.35     -114.57        1.79        4.79         2104.03


Coefficients:
                       Estimate Std. Error t value             Pr(>|t|)
(Intercept)             -120.64      42.68  -2.827              0.00507 **
gesamt$COMPLEXITY        119.18      11.55  10.317 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 387.9 on 256 degrees of freedom
  (24 observations deleted due to missingness)
Multiple R-squared:  0.2937,    Adjusted R-squared:  0.2909
F-statistic: 106.4 on 1 and 256 DF,  p-value: < 0.00000000000000022
```

In einem zweiten Schritt setzen wir FREQUENCY als unabhängige Variable ein:

```
Call:
lm(formula = toSeconds(as.character(gesamt$TIME)) ~ gesamt$FREQUENCY)
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-239.19 -235.70 -200.83   61.82 2341.18


Coefficients:
                           Estimate   Std. Error t value             Pr(>|t|)
(Intercept)           239.20266348  27.79770676   8.605 0.000000000000000557
***
gesamt$FREQUENCY       -0.00009729   0.00004870  -1.998               0.0467
*
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 443.3 on 280 degrees of freedom
Multiple R-squared:  0.01406,    Adjusted R-squared:  0.01053
F-statistic: 3.992 on 1 and 280 DF,  p-value: 0.04669
```

Eine Interpretation der Analysen ergibt, dass COMPLEXITY deutlich mehr Werte von TIME abdeckt als FREQUENCY. Das korrigierte Bestimmtheitsmaß (*Adjusted R-squared*) von 0,2909 drückt aus, dass immerhin knapp 30 Prozent aller realen Ausprägungen durch die Anzahl der Suchattribute erklärt werden können. Wenig bis nichts zur Erklärung steuern dagegen die Belegzahlen bei, hier streuen die gemessenen Werte extrem. Auch der hohe t-Wert (*t value*) von über 10 für COMPLEXITY bestätigt den starken empirischen Einfluss der Suchattributsanzahl auf die Laufzeiten, während ein t-Betrag von ca. 1 für FREQUENCY[126] auf keinerlei bedeutsamen Einfluss hindeutet. Bei einem Signifikanzniveau von 0,05 ist die Variable FREQUENCY (0,0467) kaum signifikant, ganz im Gegensatz zu COMPLEXITY (<0.0000000000000002).

Beschränken wir uns bei der Analyse von COMPLEXITY auf Abfragen mit maximal fünf Suchattributen (R-Datensatz *ohnesechs*), verbessern sich die Werte für das korrigierte Bestimmtheitsmaß auf 0,3846 (d.h. knapp 40% der Suchzeiten lassen sich durch die Anzahl der Suchkriterien erklären) sowie der t-Wert auf > 11:

[126] Dass der t-Wert hier sogar ins Negative geht und damit grundsätzlich einen gegenteiligen Effekt impliziert, kann aufgrund des niedrigen Betrags vernachlässigt werden.

```
Call:
lm(formula = toSeconds(as.character(ohnesechs$TIME)) ~ ohnesechs$COMPLEXITY)
Residuals:
             Min    1Q Median      3Q             Max
         -600.32            -110.86   24.26   26.47 1967.96


Coefficients:
                             Estimate Std. Error t value             Pr(>|t|)
(Intercept)                  -182.75      40.46  -4.517            0.0000105
***
ohnesechs$COMPLEXITY          158.82      13.84  11.473 < 0.0000000000000002
***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


Residual standard error: 342.7 on 208 degrees of freedom
  (24 observations deleted due to missingness)
Multiple R-squared:  0.3876,     Adjusted R-squared:  0.3846
F-statistic: 131.6 on 1 and 208 DF,  p-value: < 0.00000000000000022
```

Eine separate statistische Analyse des COMPLEXITY-TIME-Modells mit dem Gesamtdatensatz (vgl. Tab. 69 mit Einzelsignifikanzen und Bestimmtheitsmaßen) für jede Korpusgröße liefert nochmals bessere Bestimmtheitsmaß-Werte. Für das kleinste Evaluationskorpus mit einer Million Token erklärt die Variable COMPLEXITY bereits über 37 Prozent der Suchzeiten, für größere Korpora steigt die Abdeckung auf über 73%. Die Signifikanz ist durchweg ausgezeichnet.

Korpus	Estimate	Signifikanz Pr(>|t|)	Adjusted R-squared
EK-1	5,457	<0,0001	0,3794
EK-2	19,123	<0,0001	0,7122
EK-3	70,007	<0,0001	0,7212
EK-4	113,280	<0,0001	0,7341
EK-5	185,190	<0,0001	0,7307
EK-6	322,020	<0,0001	0,6721

Tab. 69: Korpusspezifische Werte des linearen Regressionsmodells für COMPLEXITY

Der Vollständigkeit halber dokumentiert Tabelle 70 die erwartungsgemäß bescheidenen Ergebnisse unserer alternativen Modellierung unter Heranziehung der Belegzahlen. Der Einfluss der Variablen FREQUENCY auf die Laufzeiten von Korpusrecherchen erscheint bei einer niedrigen Werteabdeckung von maximal 11% weder für große noch für kleine Korpusgrößen besonders relevant.

Korpus	Estimate	Signifikanz Pr(>|t|)	Adjusted R-squared
EK-1	-0,0384300	0,0659	0,05257
EK-2	-0,0013231	0,0102	0,11860
EK-3	-0,0004249	0,0164	0,10180
EK-4	-0,0003431	0,0167	0,10110
EK-5	-0,0002787	0,0153	0,10420
EK-6	-0,0002533	0,0213	0,09262

Tab. 70: Korpusspezifische Werte des linearen Regressionsmodells für FREQUENCY

4.11.4 Fazit

Insgesamt lässt sich hinsichtlich des Einsatzes unserer relationalen Modellierung für die Recherche in authentischem Sprachmaterial ein zweigeteiltes Fazit ziehen: Die Evaluation des linguistisch motivierten Anforderungskatalogs liefert bereits auf dem Mid-Range-Referenzsystem für die Praxis ausreichend niedrige Laufzeiten. Wenig Optimierungsbedarf konstatieren wir für diejenigen Fälle, in denen maximal zwei Suchattribute zum Einsatz kommen, also für die Abfragen 1, 2, 7, 9 und 10. Dieser positive Befund gilt insbesondere für sukzessiv anwachsende Korpusvolumina; die Suchzeiten skalieren erfreulicherweise in allen Fällen deutlich unterproportional.

Erhöht sich allerdings die Anzahl linguistischer Suchattribute und damit der Verknüpfungen innerhalb einer Abfrage, dann erreichen die ermittelten Laufzeiten ein für die Rechercheпraxis weniger ansprechendes Niveau (Abfragen 3, 4, 5, 6 und 8). Ein signifikanter Zusammenhang zwischen Abfragekomplexität und Abfragedauer lässt sich statistisch nachweisen, die Anzahl der Belegtreffer scheint dagegen kaum eine Rolle zu spielen. Für Referenzkatalogabfragen mit mehr als zwei Suchattributen streben wir deshalb nachfolgend eine Modifikation des Recherchemodells an.

Auch die Evaluation der Abfragen auf einer leistungsfähigeren Hardwareplattform offenbart Verbesserungspotenzial. Eine Erhöhung der Anzahl nutzbarer CPU-Kerne (vertikale Skalierung) geht bei den getesteten SQL-Statements nicht mit einer linearen Verbesserung der Performanz einher. Die Suchzeiten auf dem Skalierungsserver liegen zwar durchgängig unter denen des Referenzsystems, aber der Sprung von vier auf sechzehn CPU-Kerne bleibt hinter den Erwartungen zurück. Das mag einerseits mit algorithmusunabhängigen Faktoren zu tun haben: Unterschiedliche L1/L2 Cachevolumen, Festplattendurchsatzwerte sowie diverse Datenbankparameter wirken sich zweifellos auf den direkten Vergleich aus. Weiterhin ließen sich die Dateien für unsere Datenbanktabellen auf beiden Systemen physikalisch auf maximal vier Festplatten verteilen, dies könnte auf dem Skalierungssystem einen potenziellen Flaschenhals darstellen. Neben der fachgerechten Balance sämtlicher Hard- und Softwareparameter übernimmt jedoch sicherlich der eingesetzte Abfragealgorithmus eine zentrale Rolle. Beim Versuch einer Optimierung unserer Retrievalstrategie im nachfolgenden Kapitel gilt es also nicht zuletzt, eine effiziente Hardwarenutzung im Blick zu behalten.

5. Versuch einer Laufzeitoptimierung durch segmentierte Abfragen

Korpusrecherchen profitieren von der abstufbaren Granularität einer relationierten Datenarchitektur und skalieren dabei, wie die Evaluation unseres Abfragekatalogs bestätigt hat, für authentisches Sprachmaterial grundsätzlich positiv. Die Steigerungsfaktoren der Laufzeiten entwickeln sich im Vergleich zu Korpusgröße und ermittelter Beleganzahl durchweg unterproportional und die absoluten Werte liegen in einem für den praktischen Einsatz angemessenen Bereich. Einzig Abfragen mit mehreren inner- und außersprachlichen Selektionskriterien weisen vergleichsweise lange Laufzeiten auf. Werden solche Suchattribute in einem komplexen Statement zusammengefasst, tendieren diese „all in one" implementierten Join-Konstruktionen bei umfangreichen Korpusgrößen zu eher unbefriedigenden Retrievalzeiten. Für praxistaugliche Recherchen auf Basis unseres Datenmodells soll deshalb versucht werden, durch einen automatisierbaren Eingriff in die Abfragemodalitäten günstigere Ergebnisse zu erzielen.

Eine mutmaßlich naheliegende Herangehensweise könnte sich auf die verstärkte Nutzung von leistungsfähigeren – hier explizit im Sinne von: schneller rechnenden – Mikroprozessoren (CPUs) konzentrieren. Die damit verbundene Idee bestünde grob gesagt darin, dass Leistungssteigerungen zukünftiger Prozessorgenerationen nicht nur kontinuierliche Volumensteigerungen auffangen, sondern auch unsere konstatierten Engpässe kompensieren sollten. Zur Beurteilung der Praktikabilität dieses Ansatzes hilft ein kurzer historischer Exkurs: Legt man die Entwicklung der Taktfrequenzen handelsüblicher CPUs während der letzten Jahrzehnte einer Vorhersage zukünftiger Leistungssteigerungen zugrunde, so erscheint die Sachlage vielversprechend. In den 1970er Jahren steigerte sich die maximale Anzahl von Rechenzyklen pro Sekunde (clock rate) von einigen hundert Kilohertz (KHz) auf ca. 8 Megahertz (MHz). Im darauf folgenden Jahrzehnt wurden sukzessive bis zu 100 MHz erreicht und zum Jahrtausendwechsel durchbrachen die Taktraten der führenden CPU-Hersteller die Grenzmarke von einem Gigahertz (GHz). Seither lässt sich indes eine deutliche Abflachung der Kurve beobachten; 2015 liegen die üblichen Taktraten bei ca. 2,3 bis 3,7 GHz. Nichtsdestotrotz: Sofern sich die skizzierte Entwicklung verlässlich fortsetzen würde, könnte man augenblicklichen Laufzeit-Engpässen beim Korpusretrieval sowie dem zu erwartenden Wachstum der Korpusgrößen gelassen begegnen.

Fälschlicherweise angeführt wird in diesem Zusammenhang gelegentlich das Mitte der 1960er Jahre formulierte Moore'sche Gesetz (Moore 1965), benannt nach dem kalifornischen Halbleiterpionier Gordon Earle Moore. Dieser prognostizierte eine dauerhafte Verdopplung der erreichbaren Prozessorleistung innerhalb von 12 Monaten. Seine Aussage wurde mehrfach korrigiert, zunächst auf 24, später auf immerhin noch 18 Monate. Für die Konzeption datenintensiver Korpusretrievalsysteme wäre dies ein bedeutsames Zukunftsversprechen:

> Moore's law is well know in the ICT [Information and Communication Technologie] […]. It states that roughly, the processor speed doubles every 1.5 years. This is an exponential improvement in speed and makes computers possible to perform more and more sophisticated tasks […]. (Jouis 2012, S. 441)

Doch sowohl eine Fortschreibung der Formel auf künftige Systeme wie überhaupt die Anwendung auf die Rechengeschwindigkeit erscheinen aus mehreren Gründen als problematisch: Zum einen stellt Moores Aussage kein naturwissenschaftlich hinlänglich belegtes Gesetz, sondern bestenfalls eine empirische Beobachtung über einen begrenzten Zeitraum dar. Physikalische und wirtschaftliche Grenzen – hinsichtlich der Integrationsdichte von CPU-Bauteilen, der Wärmeableitung, der Fabrikationskosten etc. – wirken einer kontinuierlichen exponentiellen Steigerung entgegen. Zum anderen – und damit nähern wir uns dem Kern des Problems für unseren intendierten Anwendungszweck – bezog sich Moores ursprüngliche Aussage primär auf die Entwicklung der Anzahl von auf einem Computerchip integrierbaren Transistoren, nicht auf die Geschwindigkeit. Die mit der zusätzlichen Komplexität einhergehende Beschleunigung ist eher als Nebeneffekt anzusehen, bewerkstelligt u. A. durch zusätzliche Einheiten auf dem Chip sowie kürzere Distanzen zwischen diesen. Und ebenso wenig wie aus der in PS/kW gemessenen Leistungsfähigkeit eines Automobilmotors dessen Maximaltempo ableitbar ist, lässt sich letztlich von der Transistorenzahl eines Mikroprozessors auf seine Rechengeschwindigkeit schließen.[127]

Während also die Steigerungskurve moderner CPU-Taktfrequenzen seit einigen Jahren tendenziell abflacht, lässt sich eine verstärkte Erhöhung der Anzahl von CPU-Kernen (*cores*) beobachten. Zur Erklärung lassen sich gleichermaßen physikalische wie wirtschaftliche Aspekte heranziehen: Ab einem bestimmten Punkt ergibt eine Maximierung der Taktrate immer weniger Sinn, weil der

[127] Einfluss auf die Arbeitsgeschwindigkeit nehmen zahlreiche andere Parameter und Merkmale, etwa die Datenbusbreite, Co-Prozessoren oder L2-Cachegrößen. Moore selbst hat seine Formel im Übrigen zwischenzeitlich als „self fulfilling prophecy" bezeichnet und optimistisch ausgeführt, dass „the law would extend for another 10 or 20 years, but that wouldn't be the end of the road" (The Inquirer 2005).

Umgang mit der damit einhergehenden erhöhten Betriebstemperatur deutlich aufwändiger als das Hinzufügen zusätzlicher Prozessorkerne wäre. Gängige CPUs sind mittlerweile mit vier bis acht Kernen bestückt, in absehbarer Zeit dürfte die Anzahl bei 64 oder sogar weit höher liegen. Diese Dynamik gilt es zu nutzen, damit zunehmend komplexere Software – wie Korpusretrievalsysteme mit diversen verschiedenartigen Suchattributstypen – in den Stand versetzt wird, optimal von der parallelen[128] Rechenleistung zu profitieren. Unsere Evaluation in Kapitel 4 hat aufgezeigt, dass die schlichte Bereitstellung zusätzlicher Prozessorkerne ohne Anpassung des Abfragealgorithmus die Korpusrecherche nicht wie theoretisch vorstellbar beschleunigt. Das Skalierungssystem mit 16 CPU-Cores übertrifft das Referenzsystem mit vier Kernen in keinem Fall mit den rein rechnerisch maximalen Ausmaßen.

Aus einem anderen Blickwinkel erscheint die Fokussierung auf alternative Suchstrategien ebenfalls sinnvoll: Solange Retrievalsysteme ihre Inhalte auf Festplatten speichern, weil In-Memory-Lösungen aus Volumen- und Kostengründen schwerlich realisierbar sind, bleibt deren Input-Output-Rate gerade bei weiter ansteigenden Prozessorgeschwindigkeiten ein veritabler Flaschenhals. Die Zugriffszeiten von Festplatten haben bislang in keinster Weise mit der Moore'schen Formel Schritt halten können und dürften dies auch in Zukunft kaum tun. Es empfiehlt sich folglich, gegenstandsspezifische Lösungen zu entwickeln, die Lese- und Schreibaktivitäten effektiver verteilen und nicht allein auf die schnellere Ausführung sequenzieller Rechenoperationen setzen. Ermutigend erscheinen die Experimente des Datenbankpioniers Michael R. Stonebraker, die belegen, dass „major RDBMS vendors can be outperformed by 1-2 orders of magnitude by specialized engines in the data warehouse, stream processing, text, and scientific database markets" (Stonebraker et al. 2007, S. 1150). Unter diesem Gesichtspunkt besteht unser nachfolgendes Ziel, ungeachtet der in anderen Forschungsbereichen vorangetriebenen Konzeptionen innovativer Datenbank-Engines, in der Modellierung und Implementierung eines auf bestehender Datenbanktechnologie basierenden Frameworks für die Optimierung komplexe Korpusrecherchen in einer ähnlichen Größenordnung.

[128] Gelegentlich werden die beiden Begriffe „Parallelität" und „Nebenläufigkeit" synonym verwendet. Wir grenzen beide im Folgenden dadurch voneinander ab, dass eine parallele Programmausführung zwingend mehrere Recheneinheiten (Mehrkernprozessor bzw. physisch getrennte Prozessoren auf einem Mainboard oder in einem Rechnerverbund) voraussetzt, während nebenläufige (*concurrent*) Abarbeitung auch durch softwaregesteuertes Wechseln zwischen einzelnen Programmschritten (*task switching*) erfolgen kann („scheinbare Parallelität").

5.1 Parallelisierung als Chance für das Korpusretrieval

Die parallele Bearbeitung komplexer Aufgabenstellungen auf großen Datenmengen ist in der Informatik ein ähnlich traditionsreiches Thema wie die Dynamik leistungsstärkerer Prozessoren. Und ebenso wie die Moore'sche Formel entstand auch das Amdahl'sche Gesetz zur parallelen Programmierung bereits Mitte der 1960er Jahre (Amdahl 1967). Sein Schöpfer, Gene M. Amdahl, setzte voraus, dass jedes informatische Problem nur zu einem gewissen Teil parallelisiert werden kann und bestimmte Teile stets sequenziell abgearbeitet werden müssen. Den erzielbaren Zeitgewinn definierte er als Quotient aus ursprünglicher Laufzeit (ohne Parallelisierung) und optimierter Laufzeit, die wiederum die Summe der nicht-parallelisierbaren und parallelisierbaren Teile ist. Die wichtigste theoretische Erkenntnis besteht darin, dass es eine Grenze der maximal erreichbaren Beschleunigung gibt. Ein System mit n Rechenkernen kann ein Problem nicht n-mal rascher abarbeiten als ein Einzelprozessor; der Zeitgewinn durch das Hinzufügen eines n-ten Rechenkerns fällt niedriger aus als noch beim (n-1)-ten Rechenkern. Verschärfend macht sich darüber hinaus der – von Amdahl seinerzeit gar nicht berücksichtigte – organisatorische Overhead durch den bei steigender Prozessorzahl erforderlichen Koordinationsaufwand (Jobkontrolle, Delegation von Teilaufgaben an die beteiligten Recheneinheiten, Synchronisierung der Resultate) bemerkbar.

Dass sich massive Parallelisierung ausreichend komplexer Berechnungen trotzdem auszahlen kann, zeigt John Gustafson mit seinem 1988 aufgestellten und nach ihm benannten Gesetz (Gustafson 1988). Darin bezieht er erstmals explizit die Problemgröße ein und argumentiert, dass bei gleichzeitig anwachsender Prozessorzahl und Problemgröße – in unserem Falle wären das beispielsweise Korpusvolumen oder Anzahl der Suchattribute – der parallel bearbeitbare Programmteil zunimmt und der sequenzielle Anteil abnehmend beschränkend wirkt. Der vorrangige Nutzen zusätzlicher Recheneinheiten liegt folglich weniger in der Beschleunigung einer konstanten Aufgabe als vielmehr in der Handhabbarkeit immer umfangreicherer Problemstellungen.

Eine kombinierte Interpretation beider Gesetze resultiert in der Einsicht, dass Parallelisierung komplexe informatische Aufgaben erheblich – wenn auch nicht grenzenlos – beschleunigen kann, und dass damit theoretisch bei gleichbleibendem Zeitaufwand beliebig große Probleme bewältigt werden können. Die Art und Weise, wie sich Computerprogramme in unabhängig voneinander ausführbare Bestandteile segmentieren lassen, und welche Programmiersprachen oder Frameworks dabei zum Einsatz kommen können, ist

Gegenstand einschlägiger Forschung.[129] Als zweckdienliches Programmiermodell hat sich seit einigen Jahren der von Google eingeführte und 2010 patentierte Map-Reduce-Ansatz[130] positioniert. Er beruht auf dem aus der funktionalen Programmierung wohlbekannten Konzept des „divide and conquer" bzw. dem Einsatz sogenannter „map"- und „reduce"-Funktionen. Ausreichend komplexe Probleme werden in singuläre Aufgaben aufgeteilt, diese parallel abgearbeitet und die Einzelergebnisse anschließend kombiniert. Entscheidend ist eine Algorithmisierung in drei bis vier aufeinander aufbauende Phasen. Um diese anhand einer originär sprachwissenschaftlichen Aufgabenstellung zu illustrieren, betrachten wir nachfolgend exemplarisch die Berechnung von Wortlängenverteilungen. Dabei soll für einen gegebenen Text eine geordnete Liste aller Einzelwortlängen sowie jeweils die Anzahl der betreffenden Wortformen ausgegeben werden. Als Eingabe verwenden wir ein aus drei Sätzen bestehendes „Lorem ipsum"-Pseudolatein (vgl. auch Abb. 45). Gemäß des Map-Reduce-Ansatzes lässt sich die Aufgabe in folgenden Phasen abarbeiten:

– Map-Phase: Die Eingabe wird in mehrere überschneidungsfreie Datensätze aufgeteilt. Diese werden an Map-Funktionen weitergereicht, die Worte segmentiert und deren Länge berechnet. Sämtliche Datensätze können parallel und unabhängig voneinander bearbeitet werden. Das Ergebnis besteht aus Schlüssel-Wert-Paaren, wobei die Wortlänge als Schlüssel dient.
– Shuffle-Phase: Hier gilt es, die Zwischenergebnisse so zu ordnen, dass die nachfolgenden Schritte ebenfalls vorrangig parallel ausgeführt werden können. Die Gruppierung erfolgt im Beispiel anhand des Wortlängen-Schlüssels. Dabei ist eine gleichermaßen effiziente wie zuverlässige Synchronisation der verteilt generierten Schlüssel-Wert-Paare wesentlich, d.h. gegebenenfalls muss auf ausstehende Map-Ergebnisse gewartet werden.
– Reduce-Phase: Sobald die Gruppierung für eine Wortlänge abgeschlossen ist, kann die entsprechende Liste an eine Reduce-Funktion weitergereicht werden. Diese berechnet für jede Wortlänge die Summe sämtlicher Vorkommen und lässt das Ergebnis in eine kombinierte Ausgabe einfließen.

[129] Zur Einführung vgl. z.B. Andrews (2000); Ben-Ari/Lutz (1985); Rauber/Rünger (2000); Ziesche (2005).

[130] Vgl. z.B. Dean/Ghemawat (2004); Lin/Dyer (2010); Miner/Shook (2013). Ranger et al. (2007) evaluieren Map-Reduce erfolgreich auf Mehrkern- und Multiprozessorsystemen. Stonebraker et al. (2010) diskutieren Architektur und Anwendungsszenarien von Map-Reduce- versus Datenbanktechnologien, während Dijcks (2009) einen konkreten Implementierungsansatz von Map-Reduce innerhalb eines RDBMS präsentiert. Yang et al. (2007) erweitern den Algorithmus für multiple relationale Datensets. Pavlo et al. (2009) thematisieren den Einsatz von Map-Reduce im DBMS-Umfeld, McCreadie et al. (2012) analysieren die Effizienz von Map-Reduce für die Optimierung langwieriger Textindizierungen.

Um Wartezeiten zu minimieren, ist die Reduce-Funktion idealerweise kommutativ und assoziativ, d.h. sowohl die Reihenfolge der zu kombinierenden Objekte als auch die Abfolge der Operationen sollten gleichgültig sein.

– Optionale Combine-Phase: Sofern Datenmaterial und Aufgabenstellung dies zulassen bzw. erforderlich machen, kann die zu transportierende Datenmenge zwischen Map- und Shuffle-Phase reduziert werden. Dabei werden üblicherweise gleichartige Schlüssel-Wert-Paare zusammengefasst. Kandidaten hierfür wären die beiden „2:in"-Paare in der dritten Map-Liste, worauf wir aus Gründen der Übersichtlichkeit verzichten.[131]

Die Implementierung der Map-Reduce-Phasen kann unter Verwendung beliebiger Programmierungssoftware erfolgen. Verbreitet ist der Einsatz des Big Data Frameworks Hadoop,[132] das sich als zusätzliche Abstraktionsschicht um eine ökonomische Lastverteilung und die Koordination der parallelen Teilaufgaben kümmert. Hadoop wurde ursprünglich von Yahoo entwickelt, seit 2008 nimmt es den Rang eines Top-Level-Projekts der Apache Software Foundation ein. Neben der Open Source-Variante existieren diverse produktspezifische Ableger für die Integration in kommerzielle Plattformen. Wesentlich ist die Verwendung eines verteilten Dateisystems, etwa des hochverfügbaren Hadoop Distributed File System (HDFS), mit dessen Hilfe sich Datendateien redundant und mit variabler Blocklänge auf Master- und Slave-Knoten innerhalb eines Computerclusters verteilen lassen. Darauf aufbauend stehen Erweiterungen für Datenbankoperationen (z.B. die spaltenorientierte Datenbank Apache HBase),[133] Data Warehousing (Apache Hive mit der Abfragesprache HiveQL)[134] sowie die High Level-Abfragesprache Apache Pig[135] zur Verfügung. Im Umfeld der datenintensiven Text- und Informationssuche kommen Hadoop und seine Komponenten wiederholt zum Einsatz: Lin et al. (2009) stellen mit Ivory einen Hadoop-basierten Ansatz für die Recherche in Webdokumenten vor, Hiemstra/Hauff (2010) evaluieren Hadoop für Retrievaloperationen über ca. 500 Millionen Webdokumenten, Verma et al. (2013) optimieren die Abfragezeiten des Frameworks unter partieller Aufhebung der Grenzen zwischen Map- und Reduce-Phasen.

---

[131] Damit einher ginge eine Änderung der Wertberechnung: Statt „2:in 2:in" gäbe die Map-Funktion „2:2" zurück, d.h. es würde bereits – wie später nochmals in der Reduce-Phase – gezählt und anstatt des Wortes eine Zwischensumme an die Shuffle-Phase übergeben.

[132] Vgl. z.B. Wartala (2012); White (2015) sowie http://hadoop.apache.org.

[133] Vgl. z.B. George (2015); Shriparv (2014) sowie http://hbase.apache.org.

[134] Vgl. z.B. Capriolo et al. (2012); Freiknecht (2014) sowie http://hive.apache.org.

[135] Vgl. z.B. Gates (2011); Pasupuleti (2014) sowie http://pig.apache.org.

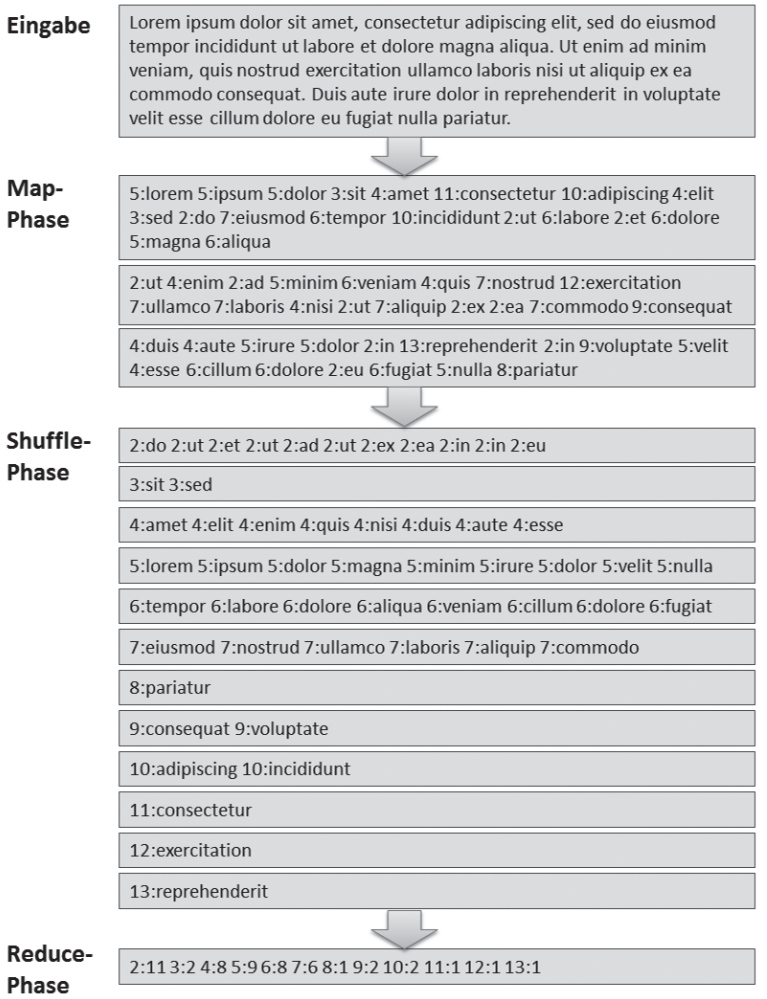


Abb. 45: Map-Reduce am Beispiel der Berechnung von Wortlängenverteilungen

Unbenommen seiner grundsätzlichen Tauglichkeit auch für das Korpusretrieval verzichten wir im Folgenden auf eine Einbeziehung des Hadoop-„Ökosystems“ in unsere Modellierung. Zum einen erreichen wir dadurch eine Reduzierung potenzieller verteilungsbedingter Overheads: Der bedeutsamste Nutzen von Hadoop liegt im Management umfangreicher Clusterstrukturen, d.h. der – zumeist redundant replizierten – Aufteilung von Input-Daten und Indizes auf unterschiedliche Rechner sowie in der Koordination der damit verbundenen Prozesse. Eine solche Systemarchitektur spielt ihre Stärken in erster Linie bei extrem zeitintensiven Operationen aus und ergäbe aus dieser

Perspektive ungeachtet der redundanzbedingten Multiplizierung des Gesamtdatenvolumens für die in Kapitel 4 kritisch evaluierten komplexen Abfragen Sinn. Daneben soll unser Referenzsystem allerdings auch vergleichsweise unproblematische Szenarien (z.B. Phänomene mit einem einzigen Suchattribut) zeitnah bedienen; in diesen Fällen stünde der Aufwand des Einsammelns von Zwischenergebnissen über diverse Cluster in keinem realistischen Verhältnis zur eigentlichen Datenabfrage.[136] Zum anderen, und dieser Aspekt ist für unsere Aufgabenstellung noch deutlich entscheidender, hilft die physische Verteilung der Ausgangsdaten nicht bei der Optimierung der initialen Abfrageformulierungen in der Map-Phase. Um hier zu einer überzeugenden Alternative zu gelangen, konzentrieren wir uns auf die angemessene Übertragung des Map-Reduce-Paradigmas auf die flexible Segmentierung komplexer sprachwissenschaftlich motivierter Korpusanfragen.[137]

5.2 Problemorientierte Algorithmisierung

Das grundlegende Prinzip von Map-Reduce, also die Segmentierung umfangreicher und mithin langwieriger Datenverarbeitungaufgaben in singuläre Tasks, erscheint grundsätzlich wie geschaffen für die Algorithmisierung komplexer Korpusrecherchen. Konkret bietet sich dieser Ansatz für solche Abfragen an, deren Laufzeiten unter alleiniger Zuhilfenahme etablierter Techniken überproportional skalieren. Abseits der reinen Recherche existieren bereits positive Evaluationen für korpusbasierte Operationen: Porta (2014) untersucht den Einsatz von Map-Reduce zur Indexierung und Berechnung von Korpusstatistiken, allerdings hauptspeicherbasiert unter massiver Nutzung von 256 GB Shared Memory. Berberich/Bedathur (2013) nutzen Map-Reduce für die Berechnung von n-Grammen aus ca. 50 Millionen Web-Dokumenten, verteilt auf mehrere Cluster Nodes.

---

[136] Vgl. z.B. Pol/Suryawanshi (2015, S. 112): „MapReduce is nice for scaling the process of huge datasets, but it is not designed to be responsive. [...] In the Hadoop implementation, as an example, the overhead of startup sometimes takes a group of minutes alone. The concept here is to take a processing job that would take days and bring it down to the form of hours, or hours to minutes, etc. But you would not start a new job in reply to a web request and expect it to finish in time to respond“. In eine ähnliche Richtung argumentiert auch Gray (2008) mit seiner Feststellung, dass die Kosten der Bereitstellung ausreichender Bandbreiten und der Kontrolle von über das Netzwerk verteilten Prozessen in vielen Anwendungsfällen über denen der lokalen Berechnung liegen („Put the computation near the data.“).

[137] Davon unberührt bleibt die grundsätzliche Option, die Abarbeitung von Teilproblemen zusätzlich unter Einsatz von Cluster-basierten Frameworks weiter zu optimieren.

Gleichwohl stellt sich speziell für Recherchen mit diversen heterogenen Suchattributen die Frage, was genau dabei aufgeteilt („gemappt") werden soll, und wie eine solche Aufteilung idealerweise durchzuführen wäre. In Schnober (2012) und Schneider (2012) finden sich Demonstrationen der grundlegenden Eignung von Map-Reduce für die Suche nach komplexen hochfrequenten Sprachphänomenen, jedoch ohne Systematisierung der Abfragesegmentierung. Prototypische Anwendungsszenarien von Map-Reduce – etwa das in der einschlägigen Literatur vielfach geschilderte Zählen von Worthäufigkeiten (vgl. z.B. Lin/Dyer 2010, S. 22ff.) oder auch unser obiges Wortlängen-Beispiel – mappen in erster Linie volumenorientiert. Aufgeteilt werden die Eingabedaten, und zwar in unabhängig voneinander behandelbare Teilmengen, für deren Abarbeitung jeweils identische Map-Funktionen bereit stehen. Üblicherweise nicht segmentiert werden dagegen die unter Umständen ebenfalls komplexen Operationen auf diesen Daten.

An diesem Punkt konstatieren wir für Korpusrecherchen mit mehreren Suchattributen ein bislang ungenutztes Optimierungspotenzial und schlagen ein alternatives, nämlich **problemorientiertes Mapping** vor. Aufgegliedert werden sollen nicht – jedenfalls nicht ausschließlich – die Eingabedaten, sondern die zu erledigenden Aufgabenstellungen. Zur Illustration kann unsere dritte Katalogabfrage zum Auffinden komplexer Relativsatzmuster herangezogen werden. Sie filtert Korpusbelege mittels folgender fünf Kriterien[138] aus dem Gesamtbestand heraus:

1) ein passender Belegsatz muss das Lemma *das* enthalten,
2) dieses soll unbedingt am Satzanfang stehen,
3) unmittelbar oder mit maximal einem Zwischenwort folgt ein Nomen,
4) anschließend steht unmittelbar ein Komma,
5) daran anschließend folgt unmittelbar die Wortform *was*.

Der zugehörige, in Abschnitt 4.3 evaluierte Suchbefehl lautete:

```
select unique T1.CO_SENTENCEID

from <EK-WORTTABELLE> T1, <EK-SATZTABELLE> T2, <EK-WORTTABELLE> T3,
<EK-WORTTABELLE> T4, <EK-WORTTABELLE> T5

where T1.CO_LEMMA = 'das' and T3.CO_POS = 'N' and T4.CO_TOKEN = ','
and T5.CO_TOKEN = 'was'

and
T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_FIRSTWORDID
```

[138] Die aufgeführten Suchkriterien beziehen sich ausschließlich auf die Wortebene, darüber hinaus gehende Phänomentypen werden später behandelt.

```
and T1.CO_SENTENCEID = T3.CO_SENTENCEID and T1.CO_ID<T3.CO_ID and
T1.CO_ID>T3.CO_ID-3
and T3.CO_SENTENCEID = T4.CO_SENTENCEID and T3.CO_ID = T4.CO_ID-1
and T4.CO_SENTENCEID = T5.CO_SENTENCEID and T4.CO_ID = T5.CO_ID-1;
```

Das Beispiel deutet bereits an, dass die Segmentierung komplexer Korpusabfragen in parallel ausführbare Teilprobleme alles andere als trivial ist. Im Gegensatz zur Volltextsuche besteht das Ziel eben nicht allein in der Auffindung von Wortsequenzen, sondern auch in der Einbeziehung von Annotationsdaten (Wortklasse). Darüber hinaus sind neben der UND-Verknüpfung gegebenenfalls die logischen Operatoren ODER bzw. NICHT für eine Abfrageformulierung relevant, d.h. eine Recherche könnte neben eindeutigen Inklusionskriterien auch alternative Ausprägungsoptionen sowie Ausschlusskriterien beinhalten. Neben strikt linearer Abfolge sind Positionsidentität oder sogar negative Abstände als Suchanforderungen vorstellbar. Weitere potenziell abfragerelevante Eigenschaften sind absolute Positionsangaben (z.B. Stellung am Satzanfang oder -ende), hierarchische syntaktische Strukturen (z.B. Zugehörigkeit zu einer Wortgruppe) sowie textbezogene Metadaten (z.B. Publikationsjahr oder regionale Zuordnung); zu Details vgl. Abschnitt 2.4.

Eine echte Abfrageparallelisierung von Teilproblemen ist vor diesem Hintergrund allein dann denkbar, wenn deren Ausführungsreihenfolge untereinander keine Rolle spielt und die gleichzeitige Abarbeitung keine unkalkulierten Seiteneffekte generiert – etwa in Form fälschlicherweise in das Endergebnis einfließender oder ausgefilterter Belege. Ein Map-Reduce-basierter Algorithmus für korpuslinguistische Zwecke umfasst deshalb folgende Bestandteile:

- Analyse der Gesamtabfrage und Segmentierung in lösbare Teilprobleme
- Zuordnung der Teilprobleme zu spezifischen Map-Prozessen
- Sammlung und Koordinierung der Zwischenergebnisse
- Reduzierung durch Bildung von Schnittmengen

Ein komplexitätsmindernder Nebeneffekt dieses Vorgehens ist der damit mögliche Verzicht auf die in Abschnitt 5.1 angesprochenen Shuffle- bzw. Combine-Phasen. Eine spezielle Anordnung der Zwischenergebnisse vor den folgenden Reduce-Aufrufen erscheint nicht notwendig, da unsere Reduce-Funktionen keine geordneten Satznummernlisten erwarten; ggf. ist lediglich eine finale Anordnung nach Text oder Korpus erforderlich. Auch eine zwischengeschaltete Vereinigung mehrerer Zwischenergebnisse kann und muss sogar entfallen, um sämtliche Fundstellen – von denen es potenziell mehrere innerhalb eines Satzes geben kann – bis zur Abarbeitung aller Teilprobleme verfügbar zu halten. Die Reduzierung mehrfach ermittelter Satznummern – also von

Sätzen in denen sämtliche Suchkriterien mehrfach bestätigt wurden – übernimmt ggf. implizit ein UNIQUE-Operator im finalen Reduce-Lauf.

Der erste Schritt, also die Segmentierung in Teilprobleme, ließe sich intuitiv durch eine Separierung der (in unserem Beispiel fünf) Einzelkriterien realisieren. Allerdings erscheint zweifelhaft, ob eine exakt darauf basierende Implementation von dann ebenfalls fünf Map-Prozessen für eine Optimierung des Laufzeitverhaltens zielführend wäre – die maximale Aufteilung komplexer SQL-Abfragen in kleinere und kleinste Einzelabfragen kann erfahrungsgemäß in ineffiziente Lösungen münden. Anders ausgedrückt: Die Übertragung von n singulären Teilproblemen auf n Map-Prozesse stellt bei ansteigendem n durch den linear zunehmenden Aufwand für Koordinierung und Reduzierung der gleichermaßen vielen wie voluminösen Zwischenergebnisse sicherlich nicht die bestmögliche Algorithmisierung dar.

Zu bestimmen gilt es daher, wie komplex die segmentierten Teilaufgaben bei der problemorientierten Algorithmisierung ausfallen sollen – und woran sich eine optimale Segmentierung orientieren kann. Bei der Antwortsuche liefern die Untersuchungen zu n-Gramm-Abfragen in Abschnitt 3.3.1 sowie das Fazit in Kapitel 4 wertvolle Anhaltspunkte. Wir haben dort für unser Datenmodell festgestellt, dass Abfragestatements mit mehr als zwei Suchattributen auf dem Referenzsystem überproportional ansteigende Laufzeiten generieren. Entsprechend beschränken wir uns nachfolgend auf die Konstruktion von Einfachjoins zwischen maximal zwei Korpustabellen.[139] Eine simple, ausschließlich an der linearen Abfolge der Suchattribute orientierte Aufteilung des Suchbefehls würde demnach für die fünf Einzelkriterien folgende drei Teilprobleme in der Map-Phase liefern:

– MAPQUERY1: Lemma *das* am Satzanfang (Suchkriterien 1 und 2)
– MAPQUERY2: Wortklasse *N* unmittelbar gefolgt von einem Komma (Suchkriterien 3 und 4)
– MAPQUERY3: Token *was* (Suchkriterium 5)

Bei der Übersetzung der beiden Zweierkombinationen sowie des bei ungerader Anzahl von Suchkriterien unvermeidlichen Einzelgängers in SQL gilt es zu beachten, dass nicht allein die Satznummern der gefundenen Belege zu-

[139] Die Beschränkung auf maximal zwei Suchkriterien pro Map-Funktion erhebt keineswegs den Anspruch, die bestmögliche Lösung für alle Systemarchitekturen zu sein. Je nach Rechenleistung und CPU-Anzahl sind durchaus andere Segmentierungsgrößen denkbar. Der Kern des propagierten Ansatzes besteht in der Aufteilung komplexer linguistischer Phänomenbeschreibungen in kleinere Aufgaben als Gegenmodell zur physischen Segmentierung des Datenbestands, unabhängig von der letztlichen Aufgabengröße.

rückgeliefert werden sollen. Als Input für nachfolgende Reduce-Operationen sind darüber hinaus die IDs der maximal zwei Suchattribute – zur Kennzeichnung der Anfangs- und Endpositionen des betreffenden Satzausschnitts – erforderlich. In SQL ausgedrückt bedeutet das:

MAPQUERY1: `select T1.CO_SENTENCEID, T1.CO_ID, T2.CO_ID from <EK-WORTTABELLE> T1, <EK-SATZTABELLE> T2 where T1.CO_LEMMA = 'was' and T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_FIRDTWORDID;`

MAPQUERY2: `select T3.CO_SENTENCEID, T3.CO_ID, T4.CO_ID from <EK-WORTTABELLE> T3, <EK-WORTTABELLE> T4 where T3.CO_POS = 'N' and T4.CO_TOKEN = ',' and T3.CO_SENTENCEID = T4.CO_SENTENCEID and T3.CO_ID = T4.CO_ID-1;`

MAPQUERY3: `select T5.CO_SENTENCEID, T5.CO_ID from <EK-WORTTABELLE> T5 where T5.CO_TOKEN = 'was';`

Abstrahierend von konkreten Teilproblemen illustriert folgender Pseudocode die Arbeitsweise einer entsprechenden Map-Funktion, nämlich die Ermittlung numerischer Tripel aus Satz-ID sowie den IDs der maximal zwei in einer MAPQUERY spezifizierten Suchkriterien:

```
mapper (MAPQUERY) =
  foreach (CO_SENTENCEID) in MAPQUERY
    output (CO_SENTENCEID, CO_ID_FIRST, CO_ID_LAST)
```

Die Ergebnisermittlung der Map-Funktionen kann parallel erfolgen. Der jeweilige Output wird in Zwischenergebnis-Tabellen (TB_MAP1, TB_MAP2 etc.) abgelegt. Jeweils zwei dieser Tabellen können anschließend von einer Reduce-Funktion weiterverarbeitet werden. Da sich das Beispiel auf der Wortebene bewegt, muss diese Reduce-Funktion zusätzlich eine exakte Angabe des gewünschten Wortabstands – oder zumindest der Reihenfolge – zwischen den beiden Satzausschnitten erhalten, um korrekte Schnittmengen zu berechnen. Die Ausgabe enthält neben der Satznummer die Start- und Endpunkte des neuen Satzausschnitts:

```
reducer (TB_MAP1, TB_MAP2, ABSTAND) =
  foreach (TB_MAP1.CO_SENTENCEID, TB_MAP2.CO_SENTENCEID)
    if abstandBerechnung (TB_MAP1.CO_ID_LAST, TB_MAP2.CO_ID_FIRST) = ABSTAND
    then output (CO_SENTENCEID, TB_MAP1.CO_ID_FIRST, TB_MAP2.CO_ID_LAST)
```

Die Übertragung des Pseudocodes in SQL ergäbe für eine Reduce-Funktion zur Abarbeitung der ersten beiden Zwischenergebnis-Tabellen mit maximal einem Zwischenwort folgendes SELECT-Statement:

REDUCEQUERY 1:
```
select T1.CO_SENTENCEID, T1.CO_ID_FIRST, T2.CO_
ID_LAST from TB_MAP1, TB_MAP2 where TB_MAP1.CO_SENTENCEID =
TB_MAP2.CO_SENTENCEID and TB_MAP1.CO_ID_LAST < TB_MAP2.CO_
ID_FIRST and TB_MAP1.CO_ID_LAST > TB_MAP2.CO_ID_FIRST-3;
```

Die illustrierte Methodik lässt sich auf beliebige Korpusrecherchen anwenden, denn komplexe Abfragen lassen sich stets in 1..n singuläre Teilabfragen oder n/2 bzw. n+1/2 Zweierkombinationen segmentieren. Jeder davon abgeleitete Map-Prozess nutzt idealerweise einen maßgeschneiderten Index. In der Reduce-Phase wird die Schnittmenge aller Zwischenergebnisse gemäß der vorgegebenen Wortabstandsangaben ermittelt. Gegebenenfalls findet die Reduzierung in mehreren aufeinander folgenden Schritten statt, d.h. der Output je zweier Reduce-Funktionen der Reduce-Phase r dient als Input einer Reduce-Funktion der nachfolgenden Reduce-Phase r+1. Werden mehr als drei initiale Map-Funktionen gestartet, ist eine Parallelisierung der Teilaufgaben während der ersten Reduce-Phase möglich. Wichtig ist die präzise logistische Kontrolle. So dürfen Reduce-Funktionen erst starten, wenn die zuliefernden Map-Funktionen vollständig abgeschlossen sind. Weiterhin muss sichergestellt sein, dass bei einer ungeraden Anzahl von Mapping-Aufrufen ein (finaler) Reduce-Schritt unter Einbeziehung des „Einzelgängers“ erfolgt. In unserem Beispiel würde also TB_MAP3 mit dem Ergebnis der REDUCEQUERY1 abgeglichen, diesmal mit UNIQUE-Operator.

Der MR-Suchalgorithmus lässt sich als Baumgraph visualisieren, der ausgehend von den Blättern zur Wurzel hin konstruiert wird. Eine Darstellung der einzelnen Schritte mitsamt Angabe der jeweiligen Abstandskriterien (natürliche Zahlen bei fixen Segmentabständen, Intervallangabe für variablen Abstand, Gleichheitszeichen für Positionsidentität) bietet Abbildung 46.

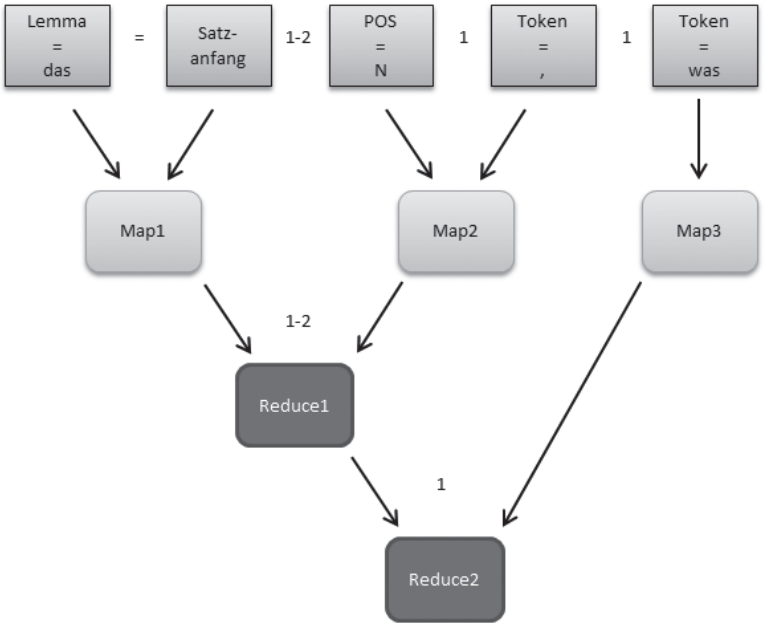


Abb. 46: Exemplarischer Algorithmus einer Abfrage auf Wortebene

Die Grundannahme des vorgeschlagenen Verfahrens besteht darin, dass die kumulierten Laufzeiten der einzelnen Phasen (Map-Phase mit drei parallel ausgeführten Map-Aufrufen, erste Reduce-Phase, zweite Reduce-Phase) trotz der Erhöhung des insgesamt zu verarbeitenden Datenvolumens[140] signifikant kürzer als die Abarbeitung des ursprünglichen Suchbefehls ausfallen. Denkbar sind auch andere Parallelisierungen, etwa sämtlicher jeweils zu einem Reduce-Ergebnis führenden Arbeitsschritte; entsprechende Evaluationen folgen weiter unten.

Daneben lenkt das Beispiel den Blick auf einen quantitativen Aspekt authentischer Sprachdaten, der ergänzend und gewinnbringend in die Logistik der Abfragesegmentierung einbezogen werden kann, nämlich die stark variierenden Häufigkeiten einzelner Phänomene:

– Für das erste Suchkriterium in MAPQUERY2 (Wortklasse=„N“) finden sich im Untersuchungskorpus über zwei Milliarden Entsprechungen, für die übrigen Suchkriterien in MAPQUERY1 und MAPQUERY2 jeweils mehrere hundert Millionen Fundstellen.

[140] Identische Satznummern für einzelne Teilprobleme werden redundant zwischengespeichert; nur Schlüsselwerte, die in sämtlichen Map-Ergebnissen enthalten sind, fließen in das Endergebnis ein.

– Dem gegenüber steht eine niedrigere Frequenz von nur ca. 5 Millionen für das Token „was“ (einzelnes Suchkriterium in MAPQUERY3).

Die Volumina der zu den Suchkriterien von MAPQUERY1 und MAPQUERY2 korrespondierenden Tabellen- bzw. Indexwerte fallen also deutlich umfangreicher aus als diejenigen von MAPQUERY3; gleiches gilt für die resultierenden Schnittmengen. In Abschnitt 3.3.4 haben wir realistische Auswirkungen solcher Verteilungsunterschiede auf Abfragezeiten einer Korpusdatenbank evaluiert, auch im Fazit von Kapitel 4 wurde der diesbezügliche Einfluss einzelner Suchkriteriumsfrequenzen beleuchtet. Übertragen auf das aktuelle Beispiel liegt darauf basierend die begründete Vermutung nahe, dass die beiden aus den Suchkriterien 1 und 2 bzw. 3 und 4 kombinierten Map-Abfragen deutlich langsamer terminieren als MAPQUERY3 mit dem Suchkriterium 5. Zur Vermeidung unproduktiver Leerlaufzeiten wäre deshalb möglicherweise eine Segmentierungsvariante günstiger, die je ein hochfrequentes Suchattribut mit einem weniger prominenten Phänomen kombiniert, also etwa Wortklasse=„N“ mit Token=„was“.

Hier stellt sich indes die Frage nach der praktischen Durchführbarkeit eines dergestalt problem- und frequenzorientierten Mappings: Da Zwischenergebnisse einzelner Map-Funktionen positionsbezogen miteinander abgeglichen werden müssen, um Konformität mit den Abstandsspezifikationen der originalen Suchanfrage zu gewährleisten, lassen sich beliebige Map-Kombinationen nicht in allen Fällen zielführend weiterverarbeiten. Schon bei einer Beschränkung auf linear organisierte Verknüpfungen mit variablen Entfernungsangaben[141] verbietet die Logik Baumkonstruktionen, bei denen bestimmte Positionsabstände zwischen einzelnen Ergebnissen der Map-Reduce-Prozesse nicht mehr zu verifizieren wären. Zu unterscheiden sind für einen Map-Reduce-Baum dabei die im Folgeabschnitt genauer spezifizierten Intervallverhältnisse der Abfolge, Überschneidung und Überdachung mit charakteristischen Implikationen auf Praktikabilität und Effizienz einer Baumvariante. Komplizierend kommen Kombinationen unterschiedlicher Annotationsebenen hinzu, etwa zu verifizierende Einbettungen in hierarchische syntaktische Strukturen.

Eine laufzeitbezogen ideale Segmentierung von Map-Funktionen konkurriert also problemabhängig mit den Anforderungen der nachfolgenden Reduce-Phase142 bzw. der Gesamtaufgabe. Diese Problematik soll nachfolgend de-

[141] Hier lassen sich drei Möglichkeiten unterscheiden: a) positionsidentisch, b) mit beliebigem Abstand, c) mit fixem positivem oder negativem Abstand.

[142] Eine explizite Shuffle-Phase darf für das beschriebene Szenario entfallen, wenn die im Zwischenspeicher abgelegten Ergebnisse (Belege) nicht nach einem bestimmten Schlüssel (z.B.

tailliert beleuchtet und idealerweise aufgelöst werden. In Anlehnung an die Kategorien unseres Anforderungskatalogs sowie an die für eine Segmentierung potenziell besonders geeigneten Abfragen – dies sind die laufzeitintensiven Abfragen Nr. 3, 4, 5, 6 und 8 – versuchen wir uns explizit an der Optimierung folgender Abfragetypen:

- Abfrage auf Wortebene (Abfragen 3, 4 und 5)
- Abfrage unter Einbeziehung textbezogener Metadaten (Abfrage 6)
- Abfrage unter Einbeziehung syntaktischer Strukturen (Abfrage 8)

Dabei werden wir zunächst logische Kombinationsbeschränkungen und phänomenspezifische Besonderheiten der Abfragetypen betrachten. Daran anschließend evaluieren wir die ausgewählten Abfragen mit dem modifizierten Algorithmus und überprüfen, ob das optimierte Modell signifikante Leistungssteigerungen generiert. Abschließend beschreiben wir eine exemplarische Implementierung in einem Online-Framework.

5.2.1 Modellierung auf Wortebene

Die Formulierung von Filterkriterien auf Wortebene stellt vermutlich die meistgenutzte Art der Korpusrecherche dar. Das mag zum einen an der Verbreitung wortbasierter Segmentierungs- und Annotationswerkzeuge in der maschinellen Sprachverarbeitung und der damit einher gehenden Dominanz wortbasierter Forschungsdaten liegen, vielleicht auch an der intuitiven Eignung der Wortebene für ein breites Anwendungsspektrum zwischen Wortschatz- und Bedeutungsanalyse. Auf jeden Fall erscheint die Recherche mit linear angeordneten Wortkriterien exemplarisch für die Handhabung hierarchisch untergeordneter Beschreibungsebenen, namentlich der Silben-, Morphem- oder Lautebene (vgl. Abschnitt 2.4). Sofern neben der reinen Kriterienabfolge keine weiteren Bedingungen – etwa konkrete Abstandswerte zwischen einzelnen Segmenten – spezifiziert werden, lässt sich ein Suchverfahren wie nachfolgend theoretisch modellieren.

Problembeschreibung: Gegeben ist für eine Korpusabfrage eine Menge $K_1 \ldots K_n$ von n Suchkriterien. Diese korrespondieren mit einer Menge $W_1 \ldots W_m$ von m Suchwörtern, wobei sich mehrere Kriterien auf dasselbe Wort beziehen dürfen; es gilt also $m <= n$.[143] Die Wörter $W_1 \ldots W_m$ liegen, mit beliebigem po-

Text- oder Satznummer) angeordnet werden müssen. Für eine Zählung der Gesamtvorkommen oder eine Zufallsauswahl ist eine entsprechende Anordnung entbehrlich.

[143] Beziehen sich mehrere Suchkriterien auf ein gemeinsames Wort, so sprechen wir von Positionsidentität. Dies ist beispielsweise der Fall, wenn gleichermaßen Lemma und Wortklasse oder eine satzinitiale Stellung eines Worts spezifiziert werden, und betrifft in unserem Anfor-

sitivem Abstand[144] zueinander, in genau dieser linearen Abfolge innerhalb jeweils eines Korpussatzes. Gesucht ist die Menge aller Sätze, auf die diese Anforderungen zutreffen. Wir unterstellen, dass jeder Satz als eine fortlaufend durchnummerierte Abfolge von Wörtern beschrieben werden kann. Die Nummer eines Wortes (Wort-ID) werde auch als dessen Position bezeichnet; die Positionen der Wörter eines Satzes bilden jeweils ein geschlossenes Intervall von natürlichen Zahlen. Jedes Wort sei genau einem Korpussatz zugeordnet, der eineindeutig durch eine natürliche Zahl (Satz-ID) identifiziert wird. Jeder Satz sei genau einem Korpustext zugeordnet, der eineindeutig durch eine alphanumerische Textsigle (Text-ID) identifiziert wird.

Eine minimale Abfrage beinhaltet ein einziges Suchkriterium, das unmittelbar auf einen Index angewendet werden kann, beispielsweise „Finde Belegsätze, in denen die Wortform *modern* vorkommt". Komplexere Abfragen verbinden mehrere Suchkriterien: „Finde Belegsätze, in denen die Wortform *modern* mit beliebigem Abstand gefolgt von der Wortform *Gebäude* vorkommt". Eine weitere typische Verwendungsweise ist die Kombination wechselnder Typen von Suchkriterien, die beim physischen Zugriff mit der Nutzung unterschiedlicher Indizes einhergeht: „Finde Belege, in denen das Lemma *modern* als Verb markiert ist und mit beliebigem Abstand von der Wortform *Gebäude* gefolgt wird". Der für solche Suchanfragen erwartete Output besteht aus der Schnittmenge der Belege für sämtliche Einzelkriterien unter zusätzlicher Sicherstellung der korrekten linearen Abfolge der entsprechenden Wortformen.

Suchverfahren: Die durch die Suchkriterien spezifizierten m Suchwörter bilden die Blätter eines gewurzelten und ungeordneten Binärbaums, des Map-Reduce-Baums (MR-Baums). Sämtliche nichtterminale Knoten korrespondieren mit den Zwischenergebnissen einzelner Map- bzw. Reduce-Funktionen. Die Baumwurzel entspricht im Erfolgsfall dem finalen Suchergebnis, d.h. einer Menge von Korpussätzen, die sämtliche n Suchkriterien und deren lineare Anordnung erfüllen.

---

derungskatalog die Abfragen 3, 5, 6 und 8. Für das hier skizzierte Suchverfahren gälte es, positionsidentische Suchkriterien auf Blattebene zusammenzufassen, so dass unterschiedliche Blätter stets auch mit unterschiedlichen Satzwörtern einhergehen. Auf diese Weise würde nebenher auch sichergestellt, dass oberhalb der Blattebene die Anfangs- und Endpositionen von Positionstripeln stets paarweise verschieden sind.

144 Grundsätzlich sind auch negative Abstände behandelbar: Bei zwei durch einen negativen Abstandsoperator („W_j vor W_i") verknüpften Wörtern kann durch Vertauschung ein positiver Abstand („W_i nach W_j") generiert werden.

Der MR-Baum beschreibt demnach das Suchverfahren vollumfänglich. Sämtliche Baumknoten sind jeweils einer Menge von geordneten Positionstripel (s, a, e) aus einer Satz-ID s sowie zwei Wortpositionen (a = Anfangsposition, e = Endposition mit a = e für Blätter sowie a < e für die übrigen Knoten) zugeordnet.

Der MR-Baum muss bestimmte Bedingungen erfüllen. Um diese formulieren zu können, vereinbaren wir folgende Terminologie. Seien $t_1 = (s, a_1, e_1)$ und $t_2 = (s, a_2, e_2)$ zwei Positionstripel mit gleicher Satz-ID s; a_1, a_2, e_1, e_2 seien paarweise verschieden. Wir nehmen ohne Beschränkung der Allgemeinheit an, dass $a_1 < a_2$ (und sagen, t_1 sei zu t_2 präzedent). Dann trifft genau eine der drei folgenden Aussagen zu:

- $a_1 < e_1 < a_2 < e_2$ (Abfolge)
- $a_1 < a_2 < e_1 < e_2$ (Überschneidung)
- $a_1 < a_2 < e_2 < e_1$ (Überdachung)

Die drei genannten Fälle bezeichnen wir im Weiteren als Intervallverhältnisse der beiden Positionstripel. Zur Illustration vgl. die Teilbäume in den Abbildungen 47 bis 49 für m = 4 Suchwörter (K_1=*A*, K_2=*B*, K_3=*C*, K_4=*D*). Die Intervallverhältnisse „Abfolge“, „Überschneidung“ und „Überdachung“ bestehen zwischen den Map-Funktionen „Map1“ und „Map2“ bzw. den von ihnen generierten Positionstripel-Mengen – nicht zwischen A, B, C und D, die trivialerweise stets aufeinander folgen.

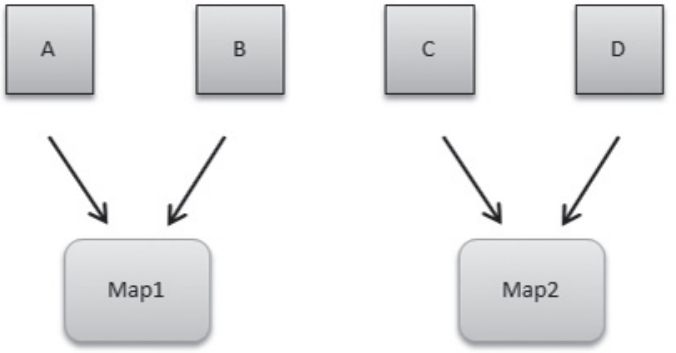


Abb. 47: Intervallverhältnis „Abfolge“ in einem binären Map-Reduce-Baum

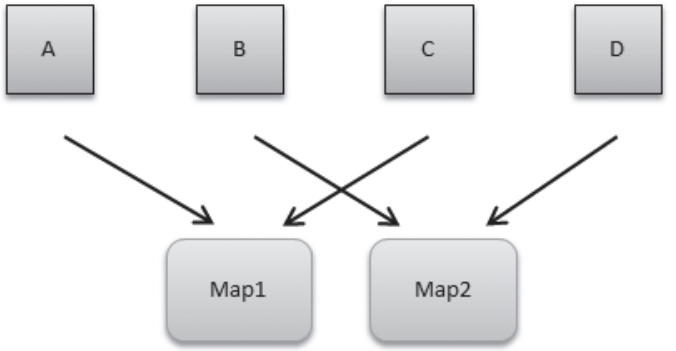


Abb. 48: Intervallverhältnis „Überschneidung“ in einem binären Map-Reduce-Baum

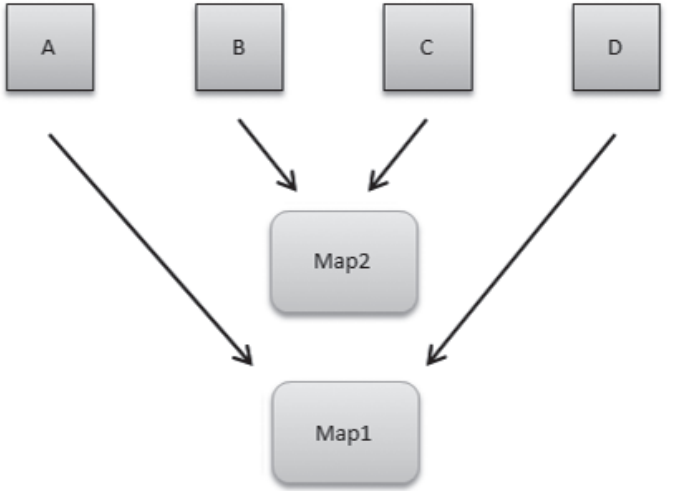


Abb. 49: Intervallverhältnis „Überdachung“ in einem binären Map-Reduce-Baum

Für jeden Mutterknoten M mit zwei Tochterknoten T_1 und T_2 (T_1 sei o. B. d. A. präzedent zu T_2) gibt es ein Intervallverhältnis V, so dass gilt: Das Positionstripel t = (s, a, e) ist Element von M genau dann, wenn es ein Positionstripel t_1 = (s, a_1, e_1) in T_1 und ein Positionstripel t_2 = (s, a_2, e_2) in T_2 gibt, so dass die Tochtertripel t_1 und t_2 im besagten Intervallverhältnis V stehen und a = a_1 sowie e = e_2 (falls V Abfolge oder Überschneidung ist) und bzw. e = e_1 (falls V Überdachung ist) sind. V heiße das Intervallverhältnis der beiden Tochterknoten; T_1 heiße präzedent zu T_2.

Definition „Ergebnis“: Jedem Positionstripel eines jeden Knotens eines gegebenen MR-Baums kann eine Menge von *Ergebnissen* zugeordnet werden. Als Ergebnis gilt eine Menge von Wortpositionen eines Satzes, die durch dieses Positionstripel abgedeckt ist. Das Ergebnis des Positionstripels (s, a, e) ist eine Menge von geordneten Paaren aus terminalen Knoten und Wortpositionen. Wir definieren rekursiv:

1) Zu jedem Positionstripel (s, a, e) eines Blattknotens K (mit a = e) gehört als einziges Ergebnis die Menge {(K, a)}.
2) Zu jedem Positionstripel (s, a, e) eines nichtterminalen Knotens gehört für jedes Paar von Tochtertripeln je ein Ergebnis, das die Vereinigungsmenge der Positionstupelmengen der beiden Tochtertripel ist.

Jedes Ergebnis eines Positionstripels eines Knotens K bezeichnen wir kurz auch als Ergebnis von K. Intuitiv ist ein Ergebnis die Menge der zu K „passenden“ Wörter – bzw. genauer deren Positionen – aus sämtlichen Korpussätzen.

Definition „relative Reihenfolge“: Ein Blatt K_1 eines MR-Baums liegt relativ zu einem gemeinsamen Vorfahrenknoten K vor einem Blatt K_2, sofern für alle Ergebnisse E von K gilt: Wenn E ein Paar (K_1, p_1) und ein Paar (K_2, p_2) enthält, dann ist $p_1 < p_2$. Abstrahierend sprechen wir von der *Reihenfolge zweier Blätter relativ zu einem gemeinsamen Vorfahrenknoten*. Es kann, je nach konkreter Baumstruktur und den damit einhergehenden Intervallverhältnissen der Knoten, nicht für jeden Knoten K verifiziert werden, dass ein Blatt K_1 relativ zu K vor einem anderen Blatt K_2 liegt. Ist dies jedoch der Fall, so sagen wir, die Reihenfolge sei relativ zu K bestimmt.

Wie bereits in den Abbildungen 47 bis 49 deutlich wurde, existieren bei mehr als zwei Blättern (= Suchwörtern) unterschiedliche Möglichkeiten der Baumgestaltung (= der Kombination der Suchkriterien für die Map- bzw. Kombination der Zwischenergebnisse in den Reduce-Funktionen). Damit einher gehen unterschiedliche Mächtigkeiten hinsichtlich der Verifizierungen der korrekten Reihenfolge von Ergebnissen einzelner Blattknoten. Ein kompletter MR-Baum erfüllt nur dann die gegebene Problembeschreibung, wenn er allein aufgrund seines Aufbaus neben sämtlichen Suchkriterien auch die korrekten linearen Abfolgen sämtlicher Ergebnisse garantiert.

Definition „reihenfolgedefiniter Knoten“: Ein nichtterminaler Knoten K eines gegebenen MR-Baums heiße *reihenfolgedefinit* gdw. gilt: Nur aus der Kenntnis sämtlicher Positionstripel der unmittelbaren Tochterknotenpaare lässt sich verifizieren, dass die Reihenfolge von irgend zwei Blättern K_1 und

K_2, die Nachfahren von K sind, relativ zu K bestimmt ist, und welche Reihenfolge bei ihnen vorliegt.

Definition „reihenfolgekorrekter Baum“: Ein MR-Baum heißt *reihenfolgekorrekt* gdw. sein Wurzelknoten reihenfolgedefinit ist, d.h. seine Ergebnisse ausschließlich aus solchen Mengen $\{(B_1, a_1), (B_2, a_2) \ldots (B_m, a_m)\}$ besteht, bei denen $a_1 < a_2 < \ldots < a_m$.

Zur Entscheidung, ob ein nichtterminaler Knoten reihenfolgedefinit bzw. ein MR-Baum reihenfolgekorrekt ist, stellen wir folgenden Satz auf:

Satz: Ein nichtterminaler Knoten K eines Binärbaums der geschilderten Konstruktion ist reihenfolgedefinit dann und nur dann, wenn seine beiden Tochterknoten T_1 und T_2 jeweils reihenfolgedefinit oder terminal sind[145] und eine der drei folgenden Bedingungen erfüllt ist:

1) Das Intervallverhältnis V der Tochterknoten ist Abfolge.

   Begründung: Im einfachsten Falle sind beide Tochterknoten T_1 und T_2 Blätter, dann ist deren Reihenfolgeinformation in der Problembeschreibung definitionsgemäß eindeutig gegeben. Sind T_1 und T_2 präzedente nicht-terminale Knoten, so gilt $a_1 < e_1 < a_2 < e_2$ und folglich, dass sämtliche Anfangs- und Endpositionen der Tochtertripel von T_1 vor denen von T_2 liegen. Im Ergebnis entsprechen die zugeordneten Wortpositionen der linearen Reihung. Abbildung 46 zeigt eine Baumstruktur, in der sich sämtliche Intervallverhältnisse als Abfolgen charakterisierten lassen.

2) Das Intervallverhältnis V der Tochterknoten ist Überschneidung und beide Tochterknoten sind Präblätter.[146]

   Begründung: Sind beide Tochterknoten T_1 und T_2 Präblätter (d.h. Knoten, deren beide Töchter Blätter mit jeweils identischer Anfangs- und Endposition sind; vgl. Abb. 48) mit $a_1 < a_2 < e_1 < e_2$, dann lässt sich die Reihenfolge der insgesamt genau vier betroffenen terminalen Blattknoten anhand der in den Zwischenergebnissen enthaltenen Positionsangaben bestimmen. Abbildung 50 skizziert ein Negativbeispiel: Hier ist „Reduce1“ kein Präblatt

[145] K kann nicht reihenfolgedefinit sein, sofern mindestens ein Tochterknoten nicht reihenfolgedefinit ist, weil die Schnittmenge der beiden Tochterknoten stets leer ist. Eine für T_1 bzw. darunter liegende Blätter fehlende Reihenfolgeinformation kann also in keinem Fall durch eine zu T_2 gehörende Reihenfolgeinformation „geheilt“ werden. Reihenfolgeinformationen zwischen terminalen Knoten (Blättern) sind per definitionem immer bekannt.

[146] Überschneidungen zwischen Blattknoten sind definitionsgemäß nicht möglich; zwischen diesen besteht durchgängig das Intervallverhältnis der Abfolge.

und in „Reduce2“ können nicht mehr die relativen Reihenfolgen sämtlicher terminalen Nachfahren sicher bestimmt werden.

3) Das Intervallverhältnis V der Tochterknoten ist Überdachung und der überdachende der beiden Knoten ist ein Präblatt.

   Begründung: Sei T_1 überdachend zu T_2 – also sämtliche Positionstripel von T_1 überdachend zu sämtlichen Positionstripel von T_2 mit $a_1 < a_2 < e_2 < e_1$ – sowie T_1 ein Präblatt; vgl. Abbildung 49. Dann liegen die Ergebnisse des ersten Tochterblatts von T_1 per definitionem vor allen Ergebnissen von T_2 ebenso wie alle Ergebnisse des zweiten Tochterblatts von T_1 nach allen Ergebnissen von T_2. Abbildung 51 skizziert ein Gegenbeispiel, bei dem der B überdachende Knoten „Reduce1“ kein Präblatt und eine Bestimmung aller relativen Reihenfolgeinformationen nicht gewährleistet ist.

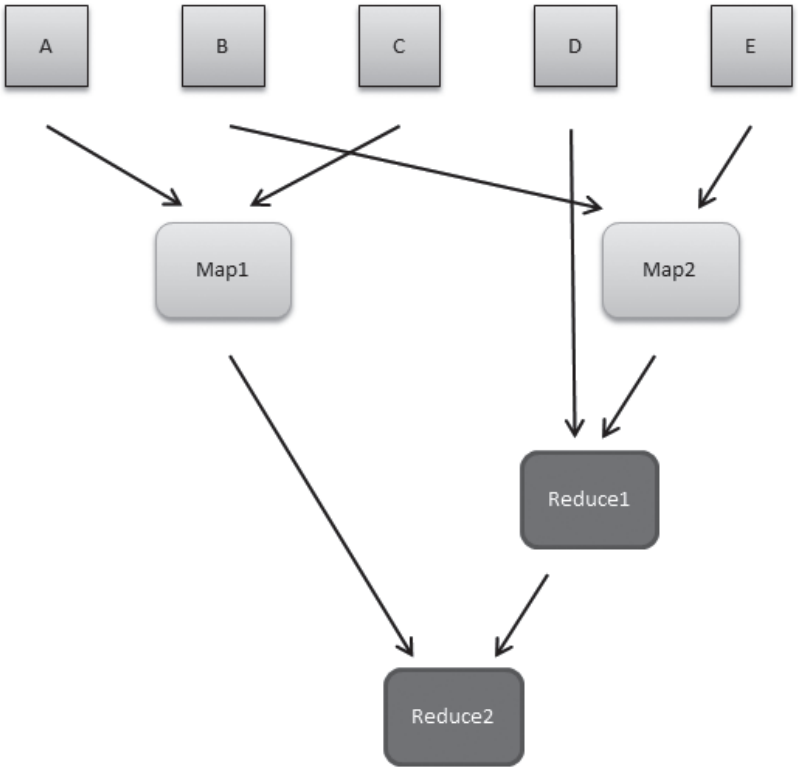


Abb. 50: Nicht reihenfolgekorrekter MR-Baum mit Überschneidung

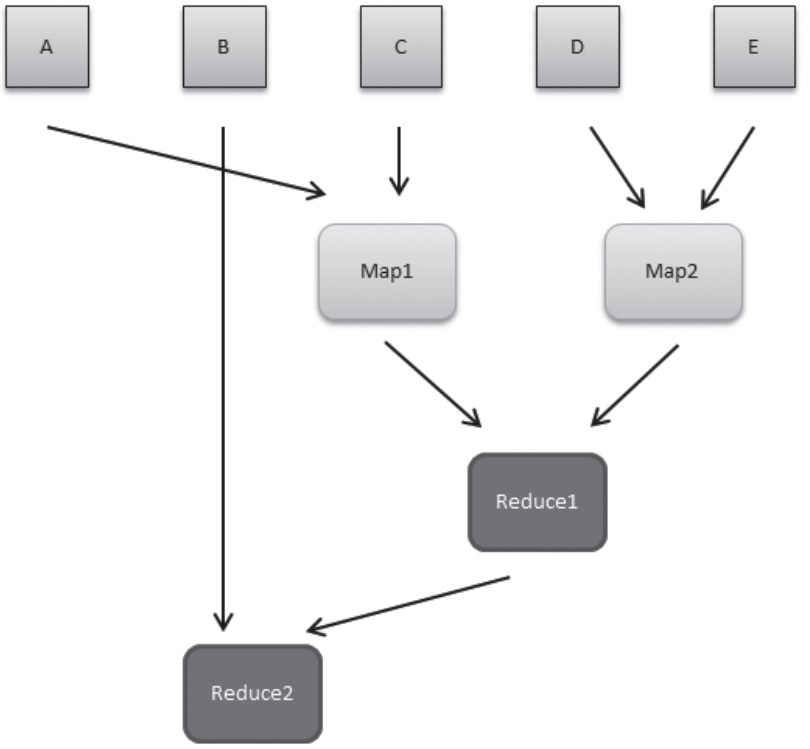


Abb. 51: Nicht reihenfolgekorrekter MR-Baum mit Überdachung

Unter Zuhilfenahme der obigen Definitionen sowie des Satzes lässt sich – etwa durch logische Programmierung[147] – bereits ein Suchalgorithmus implementieren, der für beliebige n Suchkriterien bzw. m Suchwörter sämtliche Graphen berechnet, die reihenfolgekorrekten MR-Bäumen entsprechen. Die Abarbeitung der segmentierten Teilprobleme gemäß der jeweiligen Graphenstruktur ist demnach ein hinreichendes Kriterium zur Sicherstellung der gewünschten linearen Abfolge. Die Auswahl eines konkreten MR-Baums aus der Liste aller generierten Baumvarianten würde an die spezifische Suchanfrage gekoppelt, um in der Praxis möglichst effiziente Mapping-Kombinationen zu erhalten. In Anknüpfung an unsere empirischen Evaluationen käme also idealerweise ein Baummodell zum Einsatz, das je ein hochfrequentes mit einem niederfrequenten Phänomen innerhalb eines Map-Knotens bzw. einer Map-Funktion vereint.

[147] An dieser Stelle gilt mein herzlicher Dank meinem Kollegen Dr. Peter Meyer, der als Machbarkeitsnachweis das beschriebene Problem unter Verwendung der logischen Programmiersprache Prolog implementiert und darüber hinaus in konstruktiven Diskussionen diverse wertvolle Anregungen beigesteuert hat.

5.2.2 Abfrage auf Wortebene mit spezifizierten Abständen

Der vorstehend skizzierte Ansatz zur Bestimmung von für die Segmentierung und Abarbeitung komplexer Korpusabfragen geeigneten Baumstrukturen beruht auf der Prämisse, dass allein die lineare Abfolge bestimmter Phänomene überprüft werden soll. Nicht einbezogen wurden Phänomene, bei denen numerisch ausgedrückte Abstände zwischen einzelnen Wortsegmenten explizit vorgegeben sind. Dies ist dann der Fall, wenn im Rahmen einer Korpusrecherche nicht nur spezifizierbar sein soll, dass innerhalb eines Belegsatzes ein Wort W1 „irgendwo" vor einem Wort W2 vorkommt, sondern beispielsweise unmittelbar davor, getrennt durch genau ein weiteres Wort oder mit mindestens zwei und maximal vier Trennwörtern dazwischen. Bezogen auf den Anforderungskatalog sowie die Evaluierungen in Kapitel 4 betrifft dies die Abfragen 3, 4 und 5.

Weiterhin soll das Ziel verfolgt werden, positionsidentische Suchkriterien – also die Charakterisierung eines einzelnen Wortes durch annotierte linguistische Merkmale (Lemma, Wortklasse etc.) – effektiver in den Suchalgorithmus zu integrieren. Das obige Verfahren fasst mehrere positionsidentische Suchkriterien innerhalb eines gemeinsamen Blattknotens zusammen, um für zwei Positionstripel $t_1 = (s, a_1, e_1)$ und $t_2 = (s, a_2, e_2)$ mit gleicher Satz-ID s sicherzustellen, dass a_1, a_2, e_1 und e_2 stets paarweise verschieden ausfallen. In der praktischen Umsetzung ginge dies einher mit dem zusätzlichen Einsatz von SQL-Verknüpfungen innerhalb einer Map-Funktion, was im Widerspruch zu der von uns angestrebten Beschränkung auf Einfachjoins zwischen maximal zwei Korpustabellen steht.

Deshalb betrachten wir nachfolgend unter Erweiterung unserer Terminologie eine alternative Algorithmisierung. Ausgangspunkt ist die bereits eingeführte Menge $K_1 \ldots K_n$ von n Suchkriterien, die eine Korpusanfrage vollumfänglich spezifizieren. Weiterhin gegeben ist eine Menge $A_1 \ldots A_m$ von m Abstandsbezeichnern mit m = n-1. Jeder Abstandsbezeichner ist ein Paar aus zwei numerischen Abstandswerten (min = minimaler Wortabstand, max = maximaler Wortabstand) mit $0 <= min <= max$. A_j definiert den erlaubten Abstand zwischen zwei Suchkriterien K_j und K_{j+1}. Wenn für A_j gilt, dass sowohl min = 0 und max = 0, dann beziehen sich K_j und K_{j+1} auf das selbe Satzwort.

Gesucht werden MR-Baummodelle – und damit Abfragealgorithmen –, die derart exakt spezifizierte Abstände zwischen Einzelergebnissen der Suchkriterien verifizieren. Die Modelle sollen nicht allein aufgrund ihres Aufbaus hinreichend für die Sicherstellung der gewünschten Abstände sein, sondern diese mit Hilfe der generierten Positionstripel an jeweils passenden Knoten

überprüfbar machen. Einzelne Verifizierungen können während der Map-Phase oder auch während eines Reduce-Schritts erfolgen. Erlaubt ist die Kombination von maximal zwei Blattknoten (Suchkriterien auf der Map-Ebene) bzw. nichtterminalen Knoten (Zwischenergebnisse auf der Reduce-Ebene), d.h. das Suchverfahren lässt sich wiederum durch einen gewurzelten und ungeordneten Binärbaum beschreiben, der von den Blättern ausgehend konstruiert wird.

An die Stelle der Betrachtung von Intervallverhältnissen zwischen Wörtern tritt eine sukzessive Überprüfung der Erfüllung von $A_1 \ldots A_m$ durch die Map- und Reduce-Funktionen. Deshalb kann zur Vereinfachung der späteren Implementierung auf eine Sonderbehandlung von Überdachungen verzichtet werden. Für jeden Mutterknoten M mit zwei Tochterknoten T_1 und T_2 soll unter Wegfall der oben angegeben Sonderfälle für die verschiedenen Intervallverhältnisse vereinfacht gelten: Das Positionstripel t = (s, a, e) ist Element von M genau dann, wenn es ein Positionstripel $t_1 = (s, a_1, e_1)$ in T_1 und ein Positionstripel $t_2 = (s, a_2, e_2)$ in T_2 gibt mit $a = a_1$ sowie $e = e_2$.

Die Verlagerung der Abstandsüberprüfungen in beliebige Knoten erweitert – neben den bereits genannten Vorteilen (Umgang mit numerischen Abständen und positionsidentischen Kriterien) – das Inventar der nutzbaren Suchalgorithmen. So stellt der in Abbildung 50 skizzierte nicht reihenfolgekorrekte MR-Baum zwar wie gesehen nicht qua Traversierung die korrekte lineare Wortabfolge sicher, eröffnet aber Möglichkeiten zur Verifizierung in der Reduce-Phase. Im Zuge der Generierung der Positionstripel für den Knoten „Reduce1“ lässt sich der Abstand zwischen Ergebnissen von Blatt D und Blatt E bestimmen, denn hier fließen die Positionen von D mit den in „Map2“ als Endpositionen enthaltenen Positionen von E zusammen. Im finalen „Reduce2“ wiederum können die übrigen Blattpositionen verifiziert werden: Die Positionstripel von „Map1“ enthalten Positionsangaben für die Blätter A und C, die Positionstripel von „Reduce1“ diejenigen von B und D.

Erwartungsgemäß erweisen sich einzelne MR-Kombinationen weiterhin als strukturell ungeeignet für konkrete Korpusrecherchen. Exemplarisch hierfür steht der in Abbildung 51 visualisierte Suchbaum. Die Abstände zwischen den Ergebnissen der Blätter D und E können zwar bereits unmittelbar in „Map2“ verifiziert werden. In „Reduce1“ kommen Positionsangaben für C und D zusammen, in „Reduce2“ Positionsangaben für A und B, so dass die korrespondierenden Abstände überprüfbar werden. Abstandswerte zwischen B und C bleiben jedoch zwingend unverifiziert, da an keiner Stelle im Baum – und mithin weder in der Map- noch in einer Reduce-Phase – entsprechende Anfangs- oder Endpositionen zueinander in Bezug gesetzt werden können.

Bevor wir die Implementierung unserer modifizierten Suchstrategie näher beleuchten, erscheint eine verbindliche Feststellung der maximalen Komplexität potenzieller MR-Bäume sinnvoll. Insbesondere soll die Frage geklärt werden, wie viele und welche Kombinationen von Zweierpärchen aus Blattknoten möglich sind. Diese Pärchen entsprechen den Verknüpfungen singulärer Suchkriterien auf Map-Ebene und sollen zur Laufzeit unter dem Gesichtspunkt der optimalen Verteilung konkreter Abfragekosten kombiniert werden. Dabei gilt es, gleichermaßen Fälle mit gerader und ungerader Kriterienanzahl einzubeziehen. Die Reihenfolge der beiden Suchkriterien eines Pärchens darf unberücksichtigt bleiben.

Definition „Zweierpärchen“: Für jede gegebene Menge $K_1 \ldots K_n$ von n Suchkriterien lassen sich p ungeordnete Zweierpärchen ohne Einerrest (d.h. echte Zweierpärchen aus jeweils zwei Suchkriterien, p >= 0) sowie q ungeordnete Zweierpärchen mit optionalem Einerrest (d.h. echte Zweierpärchen plus maximal ein einzelnes Suchkriterium, q >= 1) bilden. Bei geradem n gilt q = p = n/2, bei ungeradem n gilt q = p+1 = (n+1)/2.

Die Kombinationen der q Zweierpärchen mit optionalem Einerrest bilden das Inventar für die Ausgestaltung von Map-Funktionen. Abbildung 52 veranschaulicht die Herangehensweise zur Bestimmung der Kombinationsanzahl k in Abhängigkeit von der Suchkriterienanzahl n mit $K_1 = A$, $K_2 = B$ usw. Für n = 1 und n = 2 existiert jeweils genau ein Zweierpärchen mit optionalem Einerrest, d.h. q = 1 und trivialerweise k = 1. Für n = 3 und n = 4 mit q = 2 sind die jeweils drei Kombinationsmöglichkeiten [AB][C], [AC][B], [BC][A] bzw. [AB][CD], [AC][BD], [AD][BC] explizit ausbuchstabiert.

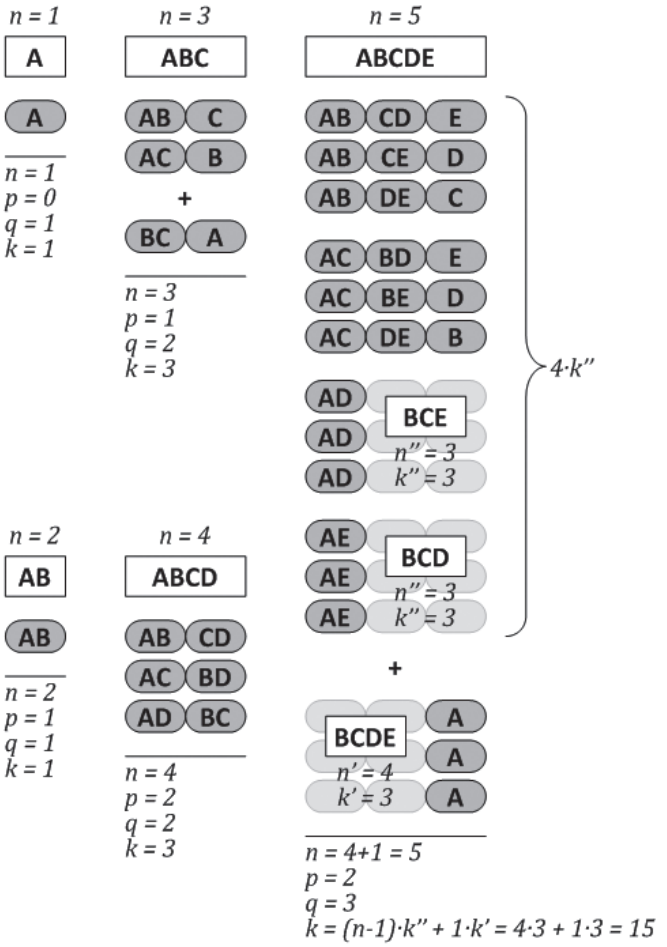


Abb. 52: Darstellung der extrahierbaren Zweierpärchen für 1 bis 5 Suchkriterien

Für n = 5 wird k erstmals formal berechnet, indem die Kombinationsmenge überschneidungsfrei aufgeteilt wird in:

– vier Submengen aus den Zweierpärchen [AB], [AC], [AD], [AE] und zugehörigen Kombinationen für die jeweils verbleibenden drei Suchkriterien
– eine fünfte Submenge aus dem singulären Suchkriterium [A] und den zugehörigen Kombinationen für die verbleibenden vier Suchkriterien

Die Kombinationsanzahl für ungerade n berechnet sich demnach als Summe aus (n-1) * k'' (mit k'' = Kombinationsanzahl für n'' = n-2) und 1 * k' (mit k' = Kombinationsanzahl für n' = n-1).

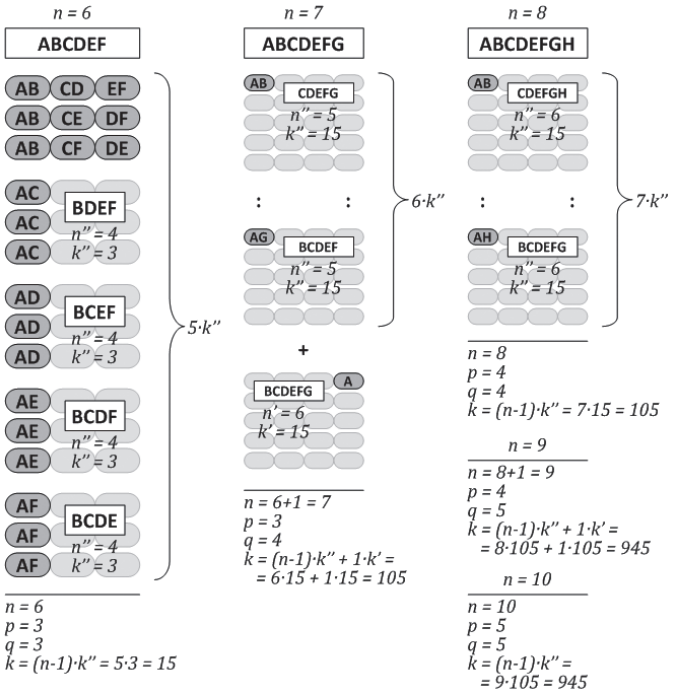


Abb. 53: Darstellung der extrahierbaren Zweierpärchen für 6 bis 10 Suchkriterien

Abbildung 53 führt die Beispiele mit ungeradem n für sieben und neun Suchkriterien fort. Weiterhin wird die Berechnung für gerade n illustriert. Für n = 6 gilt beispielsweise, dass sich die Menge der ungeordneten Pärchenkombinationen aus fünf Submengen der Zweierpärchen [AB], [AC], [AD], [AE], [AF] und den zugehörigen Kombinationen der jeweils verbleibenden vier Suchkriterien zusammensetzt.

Für gerade n lässt sich verallgemeinern: k = (n-1) * k'' (mit k'' = Kombinationsanzahl für n'' = n-2).

$$p = \left\lfloor \frac{n}{2} \right\rfloor$$

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
p	0	1		2		3		4		5		6		7		8		9

$$q = \left\lceil \frac{n}{2} \right\rceil$$

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
q	1		2		3		4		5		6		7		8		9	

$$k = \prod_{i=1}^{q} (2i-1) = (2q-1)!!$$

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
k	1		3		15		105		945		10.395		135.135		2.027.025		34.459.425	

Abb. 54: Iteration der Formeln für Zweierpärchen und Zweierpärchen-Kombinationen

Abbildung 54 iteriert die Formeln zur Bildung von Zweierpärchen bzw. Zweierpärchen-Kombinationen bis n = 18. Für die Bestimmung der Anzahl ungeordneter Kombinationen von Zweierpärchen mit optionalem Einerrest wird eine vereinfachte Formel abgeleitet. Die zur Berechnung herangezogene doppelte Fakultät ist für eine gerade Zahl z das Produkt aller geraden Zahlen kleiner gleich z bzw. für ungerade z das Produkt aller ungeraden Zahlen kleiner gleich z.[148]

Es wird deutlich, dass k für ansteigende Suchkriterienanzahlen rasch in Größenordnungen expandiert, die eine exhaustive Berechnung sämtlicher Kombinationen – zuzüglich davon abhängiger Baumverläufe zur Wurzel hin – zur Laufzeit einer Korpusabfrage nicht empfehlenswert machen. Aus diesem Grund lagern wir diesen Vorgang in ein externes, rekursiv arbeitendes Programmpaket aus. Die Suchkriterien A, B, C, ... werden dabei zur einfacheren Verarbeitung und Rezeption auf natürliche Zahlen 1, 2, 3, ... abgebildet.

Das Programm generiert zu einem numerischen Eingabeparameter (Anzahl der Suchkriterien) sämtliche ungeordnete Kombinationen von Zweierpärchen (für die Map-Funktionen) sowie rekursiv sämtliche passenden Reduce-Varianten. Die Ergebnisse bestehen folglich aus kompletten MR-Bäumen. Für jede Map- und Reducevariante wird geprüft, welche Abstände dort sicher verifizierbar sind. Nur wenn letztlich sämtliche Abstände verifiziert sind, ist eine MR-Variante (=Zweierpärchen-Kombination) für die Implementierung geeignet. Die Liste aller retrievaltauglichen Baummodelle kann abschließend in der Korpusdatenbank hinterlegt und für das problemorientierte Mapping genutzt werden.

Nachfolgend die Programmausgabe für fünf Suchkriterien:

```
MAPVARIANTE 1: [1:2][3:4][5:5]
Verifiziert durch Mapping: {1-2} {3-4}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:4] [5:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [3:4]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:2] [3:5]
```

[148] Darüber hinaus entspricht die Formel der Lösung für Schröders drittes Problem („Wieviele binäre Klammerungen sind für ein Wort der Länge n möglich?"); vgl. Schröder (1870), Pitman/Rizzolo (2015) sowie die „On-Line Encyclopedia of Integer Sequences" (https://oeis.org/A001147). Mein herzlicher Dank geht an dieser Stelle an meinen Kollegen Peter M. Fischer für diverse wertvolle Hinweise und Unterstützung bei der Ausdifferenzierung der Kombinationsmöglichkeiten.

```
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 2: [1:2][3:5][4:4]
Verifiziert durch Mapping: {1-2}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:5] [4:4]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:4] [3:5]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:2] [3:4]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {2-3} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 3: [1:2][3:3][4:5]
Verifiziert durch Mapping: {1-2} {4-5}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:3] [4:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {2-3} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert:
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [3:3]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:3]
Durch diese Reducevariante verifiziert:
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3} {3-4}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:2] [3:5]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {2-3} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 4: [1:3][2:4][5:5]
Verifiziert durch Mapping:
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:4] [5:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [2:4]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3} {3-4}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:3] [2:5]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
MR verifiziert sämtliche Grenzen.
```

```
MAPVARIANTE 5: [1:3][2:5][4:4]
Verifiziert durch Mapping:
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:5] [4:4]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:4] [2:5]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:3] [2:4]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 6: [1:3][2:2][4:5]
Verifiziert durch Mapping: {4-5}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:2] [4:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {2-3}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [2:2]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:2]
Durch diese Reducevariante verifiziert: {1-2} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:3] [2:5]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {2-3}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
MR-Kombinationen nicht geeignet!

MAPVARIANTE 7: [1:4][2:3][5:5]
Verifiziert durch Mapping: {2-3}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:3] [5:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert: {4-5}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [2:3]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:3]
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:4] [2:5]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
MR-Kombinationen nicht geeignet!

MAPVARIANTE 8: [1:4][2:5][3:3]
Verifiziert durch Mapping:
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:5] [3:3]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:3]
```

```
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3} {3-4}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:3] [2:5]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert: {4-5}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:4] [2:3]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:3]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert: {4-5}
MR-Kombinationen nicht geeignet!

MAPVARIANTE 9: [1:4][2:2][3:5]
Verifiziert durch Mapping:
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:2] [3:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {2-3}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4} {4-5}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [2:2]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:2]
Durch diese Reducevariante verifiziert: {1-2} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:4] [2:5]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
MR-Kombinationen nicht geeignet!

MAPVARIANTE 10: [1:5][2:3][4:4]
Verifiziert durch Mapping: {2-3}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:3] [4:4]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert: {4-5}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:4] [2:3]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:3]
Durch diese Reducevariante verifiziert: {1-2} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:5] [2:4]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 11: [1:5][2:4][3:3]
Verifiziert durch Mapping:
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:4] [3:3]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:3]
Durch diese Reducevariante verifiziert: {1-2} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:3] [2:4]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:4]
```

```
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert: {4-5}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:5] [2:3]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:3]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert: {4-5}
MR-Kombinationen nicht geeignet!

MAPVARIANTE 12: [1:5][2:2][3:4]
Verifiziert durch Mapping: {3-4}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:2] [3:4]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {2-3}
Durch diese Map-Reduce-Kombination nicht verifiziert: {4-5}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:4] [2:2]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:2]
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:5] [2:4]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 13: [1:1][2:3][4:5]
Verifiziert durch Mapping: {2-3} {4-5}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:3] [4:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert:
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [2:3]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:3]
Durch diese Reducevariante verifiziert: {1-2}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:1] [2:5]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {3-4}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 14: [1:1][2:4][3:5]
Verifiziert durch Mapping:
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:4] [3:5]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:5] [2:4]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3} {3-4}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:1] [2:5]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:5]
```

```
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.

MAPVARIANTE 15: [1:1][2:5][3:4]
Verifiziert durch Mapping: {3-4}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:5] [3:4]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:4] [2:5]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:1] [2:4]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
MR verifiziert sämtliche Grenzen.
```

Von den 15 errechneten Varianten sind demnach für eine Abfrage mit spezifizierten Wortabständen folgende zehn Konstruktionen geeignet:[149]

MR-Baum	Mapping 1	Mapping 2	Mapping 3
MAPVARIANTE 1	[1:2] bzw. [AB]	[3:4] bzw. [CD]	[5:5] bzw. [E]
MAPVARIANTE 2	[1:2] bzw. [AB]	[3:5] bzw. [CE]	[4:4] bzw. [D]
MAPVARIANTE 3	[1:2] bzw. [AB]	[3:3] bzw. [C]	[4:5] bzw. [DE]
MAPVARIANTE 4	[1:3] bzw. [AC]	[2:4] bzw. [BD]	[5:5] bzw. [E]
MAPVARIANTE 5	[1:3] bzw. [AC]	[2:5] bzw. [BE]	[4:4] bzw. [D]
MAPVARIANTE 10	[1:5] bzw. [AE]	[2:3] bzw. [BC]	[4:4] bzw. [D]
MAPVARIANTE 12	[1:5] bzw. [AE]	[2:2] bzw. [B]	[3:4] bzw. [CD]
MAPVARIANTE 13	[1:1] bzw. [A]	[2:3] bzw. [BC]	[4:5] bzw. [DE]
MAPVARIANTE 14	[1:1] bzw. [A]	[2:4] bzw. [BD]	[3:5] bzw. [CE]
MAPVARIANTE 15	[1:1] bzw. [A]	[2:5] bzw. [BE]	[3:4] bzw. [CD]

Tab. 71: Retrievaltaugliche MR-Bäume für fünf Suchkriterien

[149] Manche Blattkombinationen eignen sich in keiner Variante zur umfassenden Abstandsverifizierung, etwa [1:4] bzw. [AD]. Für die nachfolgende Evaluation der optimierten Referenzabfrage 4 wird gezeigt, wie sich diese Beschränkung im Einzelfall umgehen lässt.

Mapvariante 5 entspricht dem in Abbildung 50 skizzierten MR-Baum, die nicht in die Tabelle aufgenommene Mapvariante 6 dem für die Abstandsverifizierung weiterhin untauglichen Retrievalbaum aus Abbildung 51.

Die Auswahl einer konkreten Mapvariante für eine Recherche erfolgt idealerweise unter Heranziehung der in der Korpusdatenbank hinterlegten Frequenzangaben für Token, Lemmata, Wortklassen usw. Auf diese Weise lässt sich zur Optimierung der Abfragekosten von Fall zu Fall entscheiden, ob etwa Kriterium A initial mit Kriterium B, C oder E kombiniert – die Kombination [AD] ist mangels vollständiger Abstandsverifizierung keine Option – oder als Einzelgänger behandelt wird. Die Zielvorgabe unseres Verfahrens besteht im Zusammenziehen des Suchkriteriums mit der umfangreichsten Ergebnismenge und des Kriteriums mit der kleinsten Ergebnismenge ab, gefolgt vom Zusammenziehen des Suchkriteriums mit der zweitgrößten Ergebnismenge und des Kriteriums mit der zweitkleinsten Ergebnismenge usw. In Anbetracht des in Kapitel 3 thematisierten Umstands, dass gemäß des Zipf'schen Gesetzes in natürlicher Sprache vergleichsweise wenige Wörter sehr häufig vorkommen, ist zur diesbezüglichen Entscheidungsfindung ein übersichtliches, leicht handhabbares Inventar ausreichend: Für die Evaluationen im weiteren Verlauf dieses Kapitels halten wir eine Liste mit knapp 100 Einträgen im Hauptspeicher vor, die sämtliche den Häufigkeitsklassen 1 bis 5 zugeordneten Textwörter umfasst. Findet sich ein Suchkriterium nicht darin, so wird es für die Pärchenbildung als niederfrequent eingestuft.

Erwähnenswert ist darüber hinaus die Behandlung weiterer logischer Verknüpfungsoperatoren, namentlich von ODER und NICHT:

- Nicht-ausschließende ODER-Disjunktionen zwischen zwei oder mehreren positionsidentischen Suchkriterien des gleichen Typs lassen sich problemlos zusammenfassen. Aus der Abfrage „Finde Belege, in denen die Wortform *Haus* oder die Wortform *Gebäude* unmittelbar nach einem Adjektiv vorkommen" resultieren dann beispielsweise die beiden Suchkriterien A: Wortklasse = „ADJ" sowie B: Lemma in („Haus", „Gebäude") mit Abstand 1.
- Komplizierter stellt sich der Umgang mit der Negation NOT in Abfragen wie „Finde Belege, in denen die Wortform *modern* mit einem maximalen Abstand von fünf Wörtern gefolgt von der Wortform *Geruch*, aber bis zum Satzende nicht gefolgt von der Wortform *Gebäude* vorkommt" dar. Da für mit NOT verknüpfte Suchkriterien (hier: *Gebäude*) keine Fundstellen und mithin keine Positionsangaben ermittelbar sind, gilt es diese stets mit einem benachbarten Suchkriterium zu kombinieren. Ideale Kandidaten sind Einzelgänger oder Suchkriterien mit niedriger Frequenz.

Komplexere Schachtelungsmöglichkeiten, die über die oben beschriebenen Fälle hinausgehen, berücksichtigen wir zur Vereinfachung unseres Retrievalmodells sowie der verwendeten Abfragesyntax nicht.

5.2.3 Abfrage unter Einbeziehung textbezogener Metadaten

Die Metadaten sämtlicher Korpustexte (für das Referenzsystem insbesondere Medium, Register, Domäne, Land, Region, Jahr; vgl. Abschnitt 3.2.1) sind gemäß unseres Datenmodells in einer gemeinsamen Texttabelle (TB_TEXT) versammelt und nicht unmittelbarer Bestandteil der darunter liegenden Segmentierungen. Soll also für ein bestimmtes Textwort oder einen Belegsatz das zugeordnete Metadatum ermittelt werden, geschieht dies vermittels relationaler Verknüpfungen. Exemplarisch hierfür steht Abfrage 6 unseres Anforderungskatalogs, die nach Belegen in nach 2000 entstandenen Quellen sucht, die thematisch der Domäne „Politik/Wirtschaft/Gesellschaft“ zugeordnet sind. Gekürzt lautete das ursprüngliche SQL-Statement:

```
select unique T1.CO_SENTENCEID from <EK-WORTTABELLE> T1 … where …
T1.CO_SENTENCEID in (select CO_SENTENCEID from <EK-SATZTABELLE> T5
where T5.CO_TEXTID in (select CO_TEXTID from TB_TEXT T6 where T6.
CO_DOMAIN = 4 and T6.CO_DATE >= 2000));
```

Dieses Vorgehen zeichnet sich einerseits durch logische Konsequenz – Metadaten bleiben ausschließlich an diejenigen Objekte gebunden, auf die sie sich originär beziehen – und vergleichsweise geringe Speicherplatzanforderungen aus. Andererseits steigern die zur Abfragezeit erforderlichen Mehrfachverknüpfungen die Kosten bzw. Laufzeiten der betreffenden SQL-Statements nicht unerheblich. Aus diesem Grund sollen an dieser Stelle kurz drei optionale Modifizierungen des Datenmodells angeführt werden:

1) Aufnahme textbezogener Metadaten in untergeordnete Tabellen: Die Ausprägungen der abfragerelevanten Metadaten erscheinen als zusätzliche Spalten in den Wort- bzw. Satztabellen und können dort analog zu Token, Lemma, Wortklasse etc. indiziert und recherchiert werden. Auf diese Weise lassen sich Laufzeiten potenziell verringern, allerdings verbunden mit einer beträchtlichen Erhöhung des Speicherbedarfs.
2) Anlegen von Lookup-Tabellen: Für jeden Metadatentyp oder auch für jede Ausprägung eines Metadatentyps werden Umsetzungstabellen angelegt, die Satznummern sämtlicher zugehöriger Korpussätze aufnehmen. Da Satznummern als Fremdschlüssel ebenfalls in Worttabellen hinterlegt sind, reduzieren sich die Abfragekosten aufgrund des Wegfalls je eines Joins mit der Texttabelle auf Kosten vergleichsweise moderater Volumensteigerungen.

3) Physische Aufteilung der untergeordneten Abfrageebenen: Die Tabellen auf der Wortebene werden feingranular nach sämtlichen existierenden Kombinationen der abfragerelevanten Textmetadaten aufgeteilt. Dies kann etwa durch das Anlegen separater Worttabellen geschehen. Prinzipielle Vorteile dieser Strategie sind die Vermeidung zusätzlicher Speicherplatzanforderungen sowie eine Reduzierung des bei einer Abfrage zu verarbeitenden Datenvolumens aufgrund der dann möglichen Beschränkung auf exakt zu den Suchkriterien passenden Datenausschnitten. Dies erscheint jedoch nur bis zu einem bestimmten Maße zweckmäßig. Für unser Referenzsystem mit fünf unterschiedlichen Ausprägungen des Metadatentyps Medium, drei Ausprägungen des Metadatentyps Register, sechs Ausprägungen des Metadatentyps Domäne, fünf Ausprägungen des Metadatentyps Land, zehn Ausprägungen des Metadatentyps Region und – im günstigsten Fall – sechs Ausprägungen des Metadatentyps Jahr/Jahrzehnt[150] wären bereits $5 \times 3 \times 6 \times 5 \times 10 \times 6 = 27.000$ Worttabellen pro Annotationsvariante erforderlich. Der mit der Separierung, Indizierung und insbesondere Abfrageausgestaltung verbundene Overhead lässt ein derartiges Vorgehen als wenig praktikabel erscheinen.

Vor Implementierung einer der Varianten gilt es, konkrete Auswirkungen der genannten Vor- und Nachteile zu evaluieren und gegeneinander abzuwägen. Auf unserem Referenzsystem durchgeführte Testreihen mit Lookup-Tabellen verliefen durchgehend laufzeitverkürzend und dürften in Anbetracht von Anzahl und Verteilung textbezogener Metadaten im Gesamtkorpus eine sinnvolle Alternative darstellen. Für die Evaluierung unserer Retrievalstrategie verzichten wir gleichwohl auf entsprechende Modifizierungen der Datenbasis, um eine Vergleichbarkeit mit den in Kapitel 4 dokumentierten Laufzeiten zu gewährleisten.

[150] Jahreszahlen einzelner Texte lassen sich entweder unter Nutzung der Tabellenspalte CO_DATE exakt recherchieren oder zur übersichtlicheren empirischen Faktorenauswertung zu Dekaden zusammenfassen. Da die frühesten Quellen unserer Datenbasis aus den 1960er-Jahren stammen, ergäbe dieses Vorgehen an Stelle der ca. 60 Einzeljahre lediglich sechs Merkmalsausprägungen.

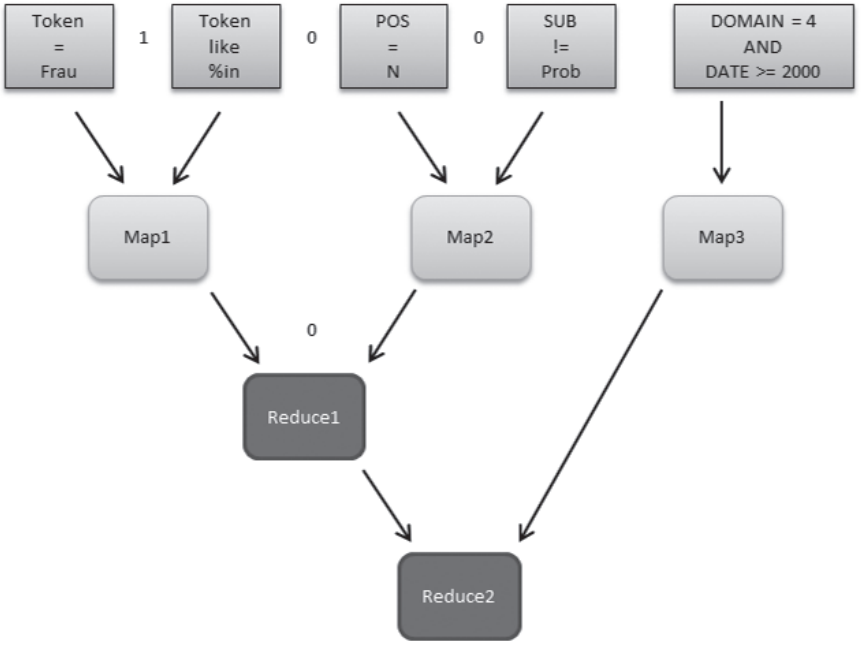


Abb. 55: Retrieval-Algorithmus mit textbezogenen Metadaten in einem Blattknoten

Die Integration textbezogener Suchkriterien in einen MR-Baum erfolgt im einfachsten Fall durch Hinzufügung eines spezifischen Blattknotens. Dieser subsummiert sämtliche Filterungen, im Falle der erwähnten Abfrage 6 also die Einschränkung der thematischen Domäne sowie des Publikationsjahrs, und operiert auf spaltenspezifischen Indizes der Texttabelle TB_TEXT. Das zusätzliche Blatt darf in der Map-Phase mit beliebigen anderen Blättern kombiniert werden, da seine Ergebnisse keine in der linearen Abfolge zu verifizierenden Wortpositionen, sondern ausschließlich Satznummern enthalten. Die Anfangs- und Endpositionen in den durch solche Mapping-Aufrufe generierten Positionstripeln (s, a, e) entsprechen den a- und e-Werten des jeweiligen Partnerknotens. Auch eine Selektion der Textmetadaten als Einzelgänger ist möglich, Abbildung 55 illustriert einen solchen Fall exemplarisch für Abfrage 6.

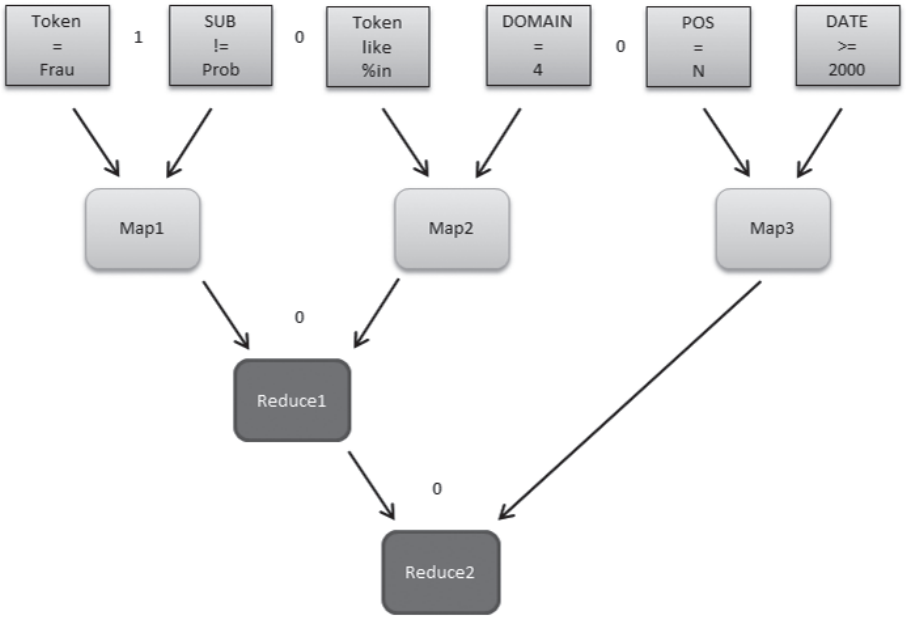


Abb. 56: Retrieval-Algorithmus mit textbezogenen Metadaten in separaten Blattknoten

Eine alternative Vorgehensweise bestünde in der Aufteilung textspezifischer Suchkriterien auf separate Blattknoten. Analog zum Vorgehen bei Textwörtern sind dabei Einteilungen in frequenzbasierte Gruppen sinnvoll. In der Map-Phase werden dann wiederum idealerweise niederfrequente Textmerkmale an hochfrequente wortbezogene Suchkriterien gekoppelt. Das damit verbundene Ziel besteht in der Reduzierung des zu verarbeitenden Datenvolumens an der frühestmöglichen Stelle des Abfragealgorithmus. Für Abfrage 6 würde sich demnach eine Kombination der beiden Textmerkmale (Einschränkung der Domäne bzw. des Publikationszeitpunkts) mit den Suchattributen „Token like %in“ bzw. „POS = N“ anbieten; vgl. Abbildung 56.

5.2.4 Abfrage unter Einbeziehung syntaktischer Strukturen und Frequenzen

Durch die umfassende Relationierung der Korpusbasis eröffnet sich die Möglichkeit, Merkmale unterschiedlicher linguistischer Beschreibungsebenen kombiniert abzufragen. Dies gilt in gleicher Weise für die Integration von Frequenzdaten, sei es in Form absoluter Werte oder durch die Subsummierung in Häufigkeitsklassen. Exemplarisch hierfür steht Abfrage 8 des Anforde-

rungskatalogs, die wortspezifische Suchkriterien (Lemma, Wortklasse, Häufigkeitsklasse) mit einer hierarchischen Spezifikation (Zugehörigkeit zu einer Adjektivphrase) vereint:

```
select unique T1.CO_SENTENCEID from <EK-WORTTABELLE> T1,
<EK-WORTTABELLE> T2, <EK-KNOTENTABELLE> T3, <EK-WORTTABELLE> T4,
<EK-LEMMALISTE> T5

where T1.CO_POS = 'ADJ'

and T2.CO_LEMMA = 'sehen' and T1.CO_SENTENCEID = T2.CO_SENTENCEID
and T1.CO_ID = T2.CO_ID

and T3.CO_PARENT = 'AP' and T2.CO_SENTENCEID = T3.CO_SENTENCEID
and T2.CO_ID = T3.CO_ID

and T4.CO_POS = 'NOUN' and T3.CO_SENTENCEID = T4.CO_SENTENCEID and
T3.CO_ID + 1 = T4.CO_ID

and T5.CO_LEMMA = T4.CO_LEMMA and T5.CO_FREQCLASS > 9;
```

Der entscheidende Unterschied zur Einbeziehung textbezogener Metadaten in unseren optimierten Suchalgorithmus besteht darin, dass die genannten – durchgehend hochfrequenten – Merkmale durchgängig an linear angeordnete Textwörter gebunden sind. Hierarchische Suchkriterien wie die Zugehörigkeit zu einer Wortgruppe liefern als Ergebnisse die eingeführten Positionstripel (s, a, e) und können analog zu anderen wortbasierten Blattknoten an beliebiger Stelle in den MR-Baum integriert werden, sofern die Verifizierung der korrekten Abstandswerte gewährleistet ist. Dies gilt in gleicher Weise für Überprüfungen der Stellung im Satz (Satzanfang oder -ende bzw. Entfernungen davon); entsprechende Angaben der Wort-IDs sind in den Satztabellen kodiert.

Abweichend gestaltet sich der Umgang mit Frequenzwerten. Da diese in einer separaten Lemmaliste hinterlegt sind und naheliegenderweise keine Positionsangaben liefern können, müssen entsprechende Suchkriterien zwingend mit dem zugehörigen Textwort verknüpft werden. In der Praxis folgt daraus, dass Blattknoten mit Frequenzbedingungen stets in eine MR-Baum-Variante integriert werden müssen, bei der das zu diesem Textwort passende Blatt als Einzelgänger erscheint; Abbildung 57 illustriert eine entsprechende Abfragesegmentierung.

Syntaktische Schachtelungen können sich naturgemäß über mehrere Hierarchieebenen erstrecken. Abfragerelevant könnte beispielsweise eine Konstruktion sein, bei der ein Wort innerhalb einer Nominalphrase (NP) auftritt, die sich ihrerseits innerhalb einer Präpositionalphrase (PP) befindet und diese wiederum Teil eines Einschubs (INS) ist; vgl. das Xerox-Annotationsbeispiel

in Abschnitt 2.4. Auch beim Umgang mit solchen Phänomenen zeigt sich die Leistungsfähigkeit unseres Datenmodells, das in der Knotentabelle stets Satz- und Wort-ID mitführt. Geschachtelte Wortgruppenspezifikationen lassen sich dadurch in separate Blattknoten aufteilen und können sogar gezielt dazu eingesetzt werden, durch Kombination mit potenziell hochfrequenten Einzelkriterien das Datenvolumen für nachfolgende Reduce-Schritte zu reduzieren.

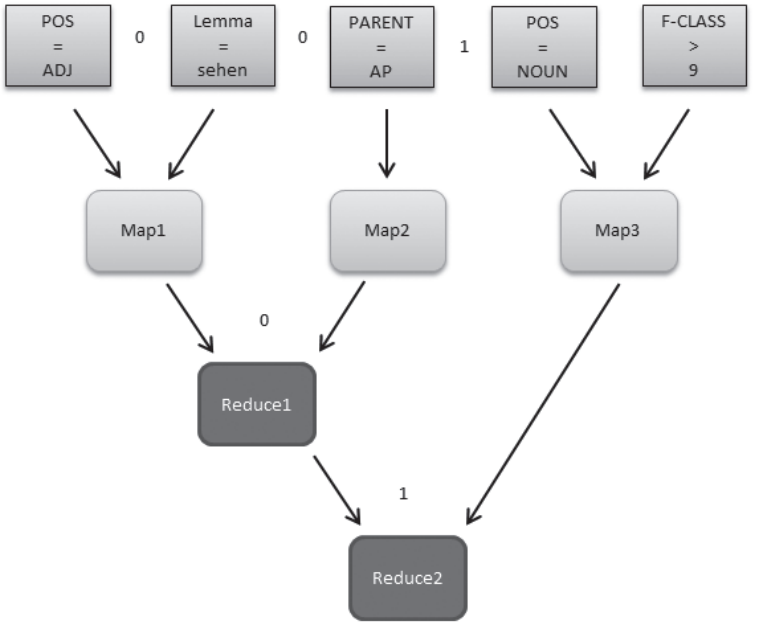


Abb. 57: Retrieval-Algorithmus mit Wortgruppenhierarchie und Frequenzkriterien

5.3 Evaluation des alternativen Suchalgorithmus

Nachdem die Evaluation des Referenzkatalogs nahegelegt hat, dass insbesondere die kombinierte Recherche nach mehreren heterogenen Suchattributen mit verbesserungsbedürftigen Laufzeiten einhergeht, widmet sich der folgende Abschnitt den entsprechenden Kandidaten. Unter Verwendung des problemorientierten Mappings sollen die Abfragen 3 (fünf Suchkriterien), 4 (sechs Suchkriterien), 5 (vier Suchkriterien), 6 (sechs Suchkriterien) und 8 (fünf Suchkriterien) modifiziert und erneut für unterschiedliche Korpusgrößen überprüft werden. Die Hoffnung auf bessere Resultate gründet sich dabei auf folgende Innovationen:

– Segmentierung in Subabfragen mit maximal zwei Suchattributen und entsprechend weniger komplexen Ausführungsplänen

– Auswahl optimaler Mapping-Pärchen, die sich an den Verteilungseigenschaften natürlichsprachlicher Phänomene (Wörter, Wortklassen, Wortgruppen) orientieren

Das Ziel der Übertragung unserer problemorientierten Segmentierungen von Suchkriterien auf SQL-basierte Recherchen in authentischem Sprachmaterial ist die Unterstützung etablierter interner Abfrageoptimierungstechnologien, nicht deren Ablösung. Dies geschieht durch Aufteilung und – soweit möglich – parallele Ausführung komplexer linguistisch motivierter Probleme in überschaubare Teilaufgaben sowie durch die explizite Steuerung der Pärchenbildung auf Basis sprachspezifischen Weltwissens.

Als Testplattformen dienen das bereits eingeführte Referenzsystem mit vier sowie das Skalierungssystem mit 16 CPU-Kernen. Auch die Korpusgrößen (EK-1 mit 1 Million Textwörtern bis EK-6 mit 8 Milliarden Textwörtern) sowie die dazugehörigen, für den datenbankinternen Query-Optimierer vorab analysierten Tabellen und Indizes entsprechen dem Datenbestand aus Kapitel 4. Neu hinzu kommen Ergebnistabellen zur Speicherung der Ausgabe einzelner Reduce-Funktionen; die Tabellennamen in den nachfolgenden Beispielen setzen sich aus dem Präfix TB_REDUCE plus der Reducenummer zusammen (TB_REDUCE1, TB_REDUCE2 etc.). Diese Tabellen enthalten folgende Spalten:[151]

– CO_SENTENCEID (Satz-ID)
– CO_ID_FIRST (Wort-ID des ersten Suchkriteriums bzw. Anfangsposition)
– CO_ID_LAST (Wort-ID des zweiten Suchkriteriums bzw. Endposition)

Eine naheliegende Implementierungsvariante der Segmentierung in Mapping-Pärchen besteht in der Verteilung auf isolierte SELECT-Statements, die ggf. sogar auf physikalisch getrennter Hardware ausgeführt werden könnten. Um den Koordinierungsaufwand überschaubar zu halten, fassen wir nachfolgend allerdings maximal zwei Pärchen mit Hilfe einer WITH-Klausel zusammen und schreiben die durch den Reduce-Schritt zusätzlich eingeschränkte Schnittmenge in eine Reduce-Tabelle. Die Zwischenergebnisse der Mapping-Funktionen werden in temporären Tabellen mit identischer Struktur hinterlegt, die gemäß SQL-99-Standard im Rahmen von WITH-Klauseln zum Einsatz kommen können. Der Grundgedanke hinter diesem auch *subquery factoring* genannt Vorgehen liegt in der Aufspaltung komplexer SQL-Abfragen in leicht administrierbare Teilbereiche. Erfahrungsgemäß tragen WITH-Klauseln in besonderem Maße zur Reduzierung der Abfragezeiten bei mehrmals durchzuführenden Subselects bei, eignen sich jedoch auch zur Abbildung von MR-Teil-

[151] Für die finale Ergebnistabelle genügt die Speicherung von Satznummern ohne Wortpositionen.

bäumen. Für die in Abbildung 57 visualisierte Korpusabfrage lässt sich die Berechnung der Ergebnisse von „Reduce1“ folgendermaßen formalisieren:

```
insert into TB_REDUCE1 (CO_SENTENCEID, CO_ID_FIRST, CO_ID_LAST)
with
      MAP1 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T2.
      CO_ID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-WORTTABELLE> T2
      where T1.CO_POS = 'ADJ' and T2.CO_LEMMA = 'sehen' and T1.CO_
      SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_ID),
      MAP2 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T1.
      CO_ID CO_ID_LAST from <EK-KNOTENTABELLE> T1 where T1.CO_PARENT
      = 'AP')
select T1.CO_SENTENCEID, T1.CO_ID_FIRST, T2.CO_ID_LAST
from MAP1 T1, MAP2 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID_LAST =
T2.CO_ID_FIRST;
```

Die Berechnung der Ergebnisse von „Reduce2“ und mithin der finalen Satznummern erfolgt anschließend in einem zweiten Schritt. Bei Abfragen mit sieben oder mehr Suchkriterien bzw. mindestens vier temporären Map-Tabellen lassen sich mehrere Reduce-Statements zeitgleich ausführen, so dass sich der Anteil parallelisierbarer Aktionen erhöht. Zur Vermeidung von Deadlocks sind daran anschließende Arbeitsschritte darauf angewiesen, dass alle relevanten temporären Tabellen mit Zwischenergebnissen gefüllt sind. Deshalb übernimmt ggf. ein Datenbank-Scheduler das Anstoßen der SQL-Statements; Start- und Endzeiten werden global protokolliert.

5.3.1 Neuevaluation Abfrage 3

In Abschnitt 4.3 wurde die Suche nach Relativsätzen mit einleitendem *was* an Stelle eines Relativpronomens ohne problemorientiertes Mapping evaluiert. Die verwendeten fünf Suchkriterien sind:

1) Lemma „das“
2) Satzanfang
3) Connexor-Wortklasse „N“
4) Token „,“ (Komma)
5) Token „was“

Ziehen wir zur Auswahl einer konkreten Abfragesegmentierung die retrievaltauglichen MR-Bäume aus Tabelle 71 heran, entspräche eine rein an der linearen Abfolge der Suchkriterien orientierte Vorgehensweise der Mapvariante 1 ([1:2][3:4][5]) mit den Pärchen (das/Satzanfang) (n/,) und (was). Um die

mit den Suchkriterien verbundenen unterschiedlichen Einzelhäufigkeiten vorteilhafter zu verteilen und damit die Vorzüge unseres optimierten Abfragealgorithmus gewinnbringender auszuschöpfen, bietet sich hingegen Mapvariante 2 ([1:2][3:5][4]) an. Diese kombiniert ebenfalls Suchkriterium 1 mit Suchkriterium 2 in TB_MAP1 und validiert dabei bereits deren Wortabstand. Abweichend verknüpft sie jedoch das teuerste Suchkriterium „N“ mit dem gemäß unserer Frequenzdaten seltensten Phänomen (Token „was“) ohne expliziten Entfernungstest.

Im ersten Reduceschritt erfolgt ein Abgleich der Ergebnisse von „MAP2“ und „MAP3“ (Reducevariante 3), im zweiten Reduceschritt kommen die Ergebnisse aus „MAP1“ hinzu. Für diesen MR-Baum lauten die zu evaluierenden SQL-Statements, die sukzessive sämtliche erforderlichen Entfernungsangaben validieren:

```
insert into TB_REDUCE1 (CO_SENTENCEID, CO_ID_FIRST, CO_ID_LAST)
with
      MAP2 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T2.
      CO_ID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-WORTTABELLE> T2
      where T1.CO_POS='N' and T2.CO_TOKEN = 'was' and T1.CO_SEN-
      TENCEID = T2.CO_SENTENCEID),
      MAP3 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T1.
      CO_ID CO_ID_LAST from <EK-WORTTABELLE> T1 where T1.CO_TOKEN =
      ',')
select T1.CO_SENTENCEID, T1.CO_ID_FIRST, T2.CO_ID_LAST
from MAP2 T1, MAP3 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID_FIRST+1 =
T2.CO_ID_FIRST and T1.CO_ID_LAST-1 = T2.CO_ID_FIRST;


insert into TB_REDUCE2 (CO_SENTENCEID)
with
      MAP1 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T2.
      CO_FIRSTWORDID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-SATZ-
      TABELLE> T2 where T1.CO_LEMMA = 'das' and T1.CO_SENTENCEID =
      T2.CO_SENTENCEID and T1.CO_ID = T2.CO_FIRSTWORDID)
select unique T1.CO_SENTENCEID
from MAP1 T1, TB_REDUCE1 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID_LAST < T2.
CO_ID_FIRST and T1.CO_ID_LAST > T2.CO_ID_FIRST-3;
```

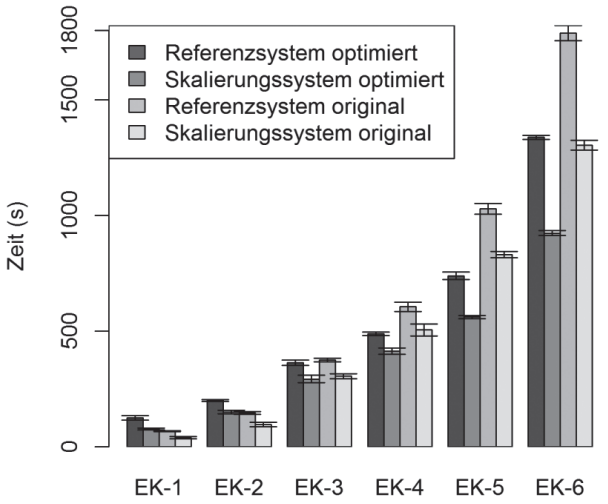


Abb. 58: Vergleich der Abfragezeiten für Abfrage 3 (optimiert, original)

Abbildung 58 sowie die Tabellen 72 und 73 dokumentieren Änderungen im Laufzeitverhalten gegenüber den Testläufen ohne Problemsegmentierung. Die angegebenen optimierten Abfragezeiten bestehen aus der Summe der Werte beider Query-Statements. Insgesamt lässt sich eine signifikante Reduzierung der Suchzeiten für mittlere und große Korpora feststellen, während die Abfragen der kleinen Korpora EK-1 und EK-2 kontraproduktiv auf die Modifikation des Algorithmus reagieren. Die Aufteilung der Suchkriterien in separate Map-Schritte sowie die Hinzufügung einer zusätzlichen Integration durch den nicht parallel organisierbaren „Reduce2"-Schritt verlängern hier die Laufzeiten. Für Korpora mit über einer Milliarde Wortformen dagegen beschleunigen sich Abfragen auf dem Referenzsystem um etwa ein Viertel der ursprünglichen Messwerte.[152]

[152] Die Verbesserung lässt sich auch durch einen Vergleich von logischem und physischem I/O nachvollziehen. Während das in Kapitel 4 eingesetzte „all in one"-Statement für EK-6 auf dem Referenzsystem 8.076.348 *consistent gets* und 1.754.319 *physical reads* benötigte, reduzieren sich die Werte für die optimal segmentierte MR-Variante drastisch auf 4.899.606 + 553.032 *consistent gets* bzw. 1.482.387 + 281.852 *physical reads*.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	1	126,17 (+57,89)	78,08 (+38,46)
EK-2	24	201,23 (+54,05)	151,08 (+53,77)
EK-3	212	364,52 (-11,71)	293,78 (-12)
EK-4	411	489,67 (-116,08)	414,13 (-91,53)
EK-5	780	739,7 (-289,29)	561,64 (-270,19)
EK-6	1.412	1.338,53 (-450,66)	923,87 (-380,37)

Tab. 72: Mittelwerte und Änderungen der optimierten Abfragezeiten für Abfrage 3 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 24	Faktor 1,59	Faktor 1,93
EK-3	Faktor 10	Faktor 8,83	Faktor 1,81	Faktor 1,94
EK-4	Faktor 2	Faktor 1,94	Faktor 1,34	Faktor 1,41
EK-5	Faktor 2	Faktor 1,9	Faktor 1,51	Faktor 1,36
EK-6	Faktor 2	Faktor 1,81	Faktor 1,81	Faktor 1,64

Tab. 73: Steigerungsfaktoren für optimierte Abfrage 3

5.3.2 Neuevaluation Abfrage 4

In Abschnitt 4.4 haben wir das Auffinden von ACI-Konstruktion, d.h. von Satzbelegen mit einer Infinitivform unmittelbar vor bestimmten Wahrnehmungsverben, ergänzt um weitere linguistisch motivierte Einschränkungen, mit einem Mehrfachjoin-Statement evaluiert. Die verwendeten sechs Suchkriterien lauten:

1) Connexor-Subkategorie „PL“
2) Lemma „haben“
3) Connexor-Subkategorie „INF“
4) Lemma „hören“, „sehen“, „spüren“, „fühlen“, „riechen“
5) Token „.“ (Punkt)
6) Satzende

In dieser Liste repräsentieren die beiden Suchkriterien 5 und 6 die Phänomene mit den höchsten Häufigkeitswerten. Deshalb überrascht auch nicht, dass eine rein linear vollzogene Map-Segmentierung ([1:2] [3:4] [5:6]) wenig überzeugende Laufzeiten generiert und für die meisten Korpusgrößen sogar deutlich langsamer terminiert als die Originalabfrage. Ideal im Sinne der Kombination von jeweils höchsten und niedrigsten, zweithöchsten und zweitniedrigsten usw. Häufigkeitswerten wäre die Kombination [1:3] [2:5] [4:6]. Leider lässt sich diese nicht für eine vollständige Verifizierung der Abstandswerte verwenden; unser Skript (vgl. Abschnitt 5.2.2) berechnet die möglichen Reduce-Pfade als insgesamt nicht geeignet:

```
MAPVARIANTE 5: [1:3][2:5][4:6]
Verifiziert durch Mapping:
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:5] [4:6]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:6]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5} {5-6}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:6] [2:5]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:5]
Durch diese Reducevariante verifiziert: {1-2} {3-4} {5-6}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3} {4-5}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:3] [2:6]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:6]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5} {5-6}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
MR-Kombinationen nicht geeignet!
```

Alternativ bietet sich eine Segmentierung in [1:3][2:4][5:6] an, die sämtliche Abstände zwischen den sechs Suchkriterien verifiziert:

```
MAPVARIANTE 4: [1:3][2:4][5:6]
Verifiziert durch Mapping: {5-6}
REDUCEVARIANTE 1 REDUCELEVEL 1 [1:4] [5:6]
REDUCEVARIANTE 1 REDUCELEVEL 2 [1:6]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {3-4} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert:
REDUCEVARIANTE 2 REDUCELEVEL 1 [1:6] [2:4]
REDUCEVARIANTE 2 REDUCELEVEL 2 [1:4]
Durch diese Reducevariante verifiziert: {1-2}
Durch diese Map-Reduce-Kombination nicht verifiziert: {2-3} {3-4}
{4-5}
REDUCEVARIANTE 3 REDUCELEVEL 1 [1:3] [2:6]
REDUCEVARIANTE 3 REDUCELEVEL 2 [1:6]
Durch diese Reducevariante verifiziert: {1-2} {2-3} {4-5}
Durch diese Map-Reduce-Kombination nicht verifiziert: {3-4}
MR verifiziert sämtliche Grenzen.
```

Die bestmögliche Zuordnung der Suchkriterien zu den Map-Pärchen gelingt in diesem Fall, sofern Kriterium 4 und Kriterium 5 vorab vertauscht werden. Ein solcher Austausch gestaltet sich problemlos, weil der geforderte Abstand zwischen den korrespondierenden Phänomenen in der Anfrage eindeutig spezifiziert ist. Nicht möglich wäre dieses Vorgehen bei einer beliebig großen Entfernung. Letztlich lauten die optimierten Abfragestatements:

```
insert into TB_REDUCE1 (CO_SENTENCEID, CO_ID_FIRST, CO_ID_LAST)
with
      MAP1 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T2.
      CO_ID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-WORTTABELLE> T2
      where T1.CO_SUB='PL' and T2.CO_SUB = 'INF' and T1.CO_SEN-
      TENCEID = T2.CO_SENTENCEID),
      MAP2 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T2.
      CO_ID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-WORTTABELLE> T2
      where T1.CO_LEMMA = 'haben' and T2.CO_TOKEN = '.' and T1.CO_
      SENTENCEID = T2.CO_SENTENCEID)
select T1.CO_SENTENCEID, T1.CO_ID_FIRST, T2.CO_ID_LAST
from MAP2 T1, MAP2 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID_FIRST =
T2.CO_ID_FIRST-1 and T2.CO_ID_FIRST < T1.CO_ID_LAST and T1.CO_ID_
LAST = T2.CO_ID_LAST-2;

insert into TB_REDUCE2 (CO_SENTENCEID)
with
      MAP3 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T2.
      CO_LASTWORDID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-SATZTA-
      BELLE> T2 where T1.CO_LEMMA in ('hören', 'sehen', 'spüren',
      'fühlen', 'riechen') and T1.CO_SENTENCEID = T2.CO_SENTENCEID
      and T1.CO_ID = T2.CO_LASTWORDID)
select unique T1.CO_SENTENCEID
from TB_REDUCE1 T1, MAP3 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID_LAST =
T2.CO_ID_FIRST+1;
```

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	0	7,82 (+2.36)	6,48 (+2.31)
EK-2	12	58,48 (+4.48)	47,3 (+7.72)
EK-3	137	132,32 (-43.43)	117,28 (-29.11)
EK-4	271	176,91 (-129.45)	131,44 (-108.11)
EK-5	583	306,69 (-191.74)	243,37 (-134.9)
EK-6	1.107	468,17 (-306.12)	361,1 (-242.81)

Tab. 74: Mittelwerte und Änderungen der optimierten Abfragezeiten für Abfrage 4 in Sekunden

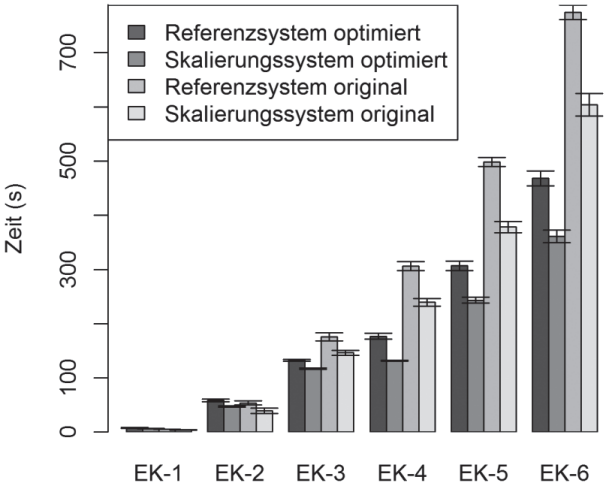


Abb. 59: Vergleich der Abfragezeiten für Abfrage 4 (optimiert, original)

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	N/A	Faktor 7,48	Faktor 7,3
EK-3	Faktor 10	Faktor 11,41	Faktor 2,26	Faktor 2,48
EK-4	Faktor 2	Faktor 1,98	Faktor 1,34	Faktor 1,12
EK-5	Faktor 2	Faktor 2,15	Faktor 1,73	Faktor 1,85
EK-6	Faktor 2	Faktor 1,9	Faktor 1,53	Faktor 1,48

Tab. 75: Steigerungsfaktoren für optimierte Abfrage 4

Die summierten Abfragezeiten der Map- und Reduce-Schritte bleiben zumeist – und oft deutlich – unter denen der Originalabfragen; vgl. Abbildung 59 sowie Tabelle 74. Eine Ausnahme bilden wie schon bei Abfrage 3 die beiden kleinsten Textkorpora; dafür reduziert sich die mittlere Laufzeit für EK-6 auf dem Referenzsystem um ca. fünf Minuten auf nun knapp acht Minuten. Auch die Skalierung verläuft weiterhin positiv: Die in Tabelle 75 ausgewiesenen Steigerungsfaktoren der optimierten Laufzeiten liegen durchgehend unter denen der Korpustokenanzahl bzw. der gefundenen Belegsätze.

5.3.3 Neuevaluation Abfrage 5

Abfrage 5 wartet in dieser Messreihe mit der kleinsten Anzahl an Suchkriterien auf, von denen eines (Wortklasse „VRB“) als Ausschlussbedingung formuliert und in Form einer NOT-Verknüpfung umgesetzt ist. Gesucht wird nach W-Fragen, bei denen auf ein satzeinleitendes adverbiales Interrogativpronomen kein Verb folgen darf. Folgende vier Einzelphänomene fließen in die Abfrage ein:

1) TreeTagger-Wortklasse „PWAV“
2) Satzanfang
3) TreeTagger-Wortklasse „VRB“
4) Token „?“ (Fragezeichen)

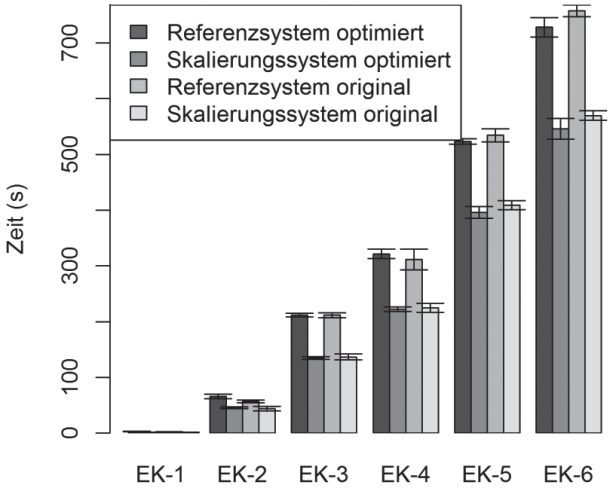


Abb. 60: Vergleich der Abfragezeiten für Abfrage 5 (optimiert, original)

Im Hinblick auf eine optimale Pärchenbildung gilt es zu berücksichtigen, dass die Abstandsforderung eines NOT-Kriteriums notwendigerweise bereits während der Map-Phase verifiziert werden muss. Als Ausschlussfaktoren sollen in diesen Fällen schließlich lediglich diejenigen Realisierungen zählen, die im angegebenen relativen Abstand zu einem bestimmten Vorgänger oder Nachfolger liegen. Übertragen auf Abfrage 5 bedeutet dies, dass nur Verben, die nach dem adverbialen Interrogativpronomen stehen, ausgeschlossen werden sollen – andere Verbpositionen bleiben im Sinne der Abfrageformulierung erlaubt.[153]

Frequenzbasiert wäre eine Verteilung der vier Suchkriterien in die Pärchen ([1:3] [2:4]) optimal, die vergleichsweise kleine „PWAV"-Gruppe – mit immerhin noch knapp sieben Millionen Realisierungen in EK-6 – träfe dabei auf die hochfrequente „VRB"-Wortklasse. Allerdings erlaubt diese Variante keine unmittelbare Verifizierung des Abstands zwischen Kriterium 2 und Ausschlusskriterium 3. Um hier regelkonform zu bleiben, wählen wir die Variante ([2:3] [1:4]), die die erforderliche Abstandsberechnung in die Map-Phase integriert:

```
insert into TB_REDUCE2 (CO_SENTENCEID)
with
      MAP1 as (select T1.CO_SENTENCEID, T1.CO_ID CO_VORNE, T1.CO_ID
      CO_HINTEN from TB_TT_SATZANFANG T1 where not exists (select
      null from TB_TT_MORPHO T2 where T2.CO_MORPHO in ('VMFIN','VAF
      IN','VVFIN','VAIMP','VVIMP','VVINF','VAINF','VMINF','VVIZU','
      VVPP','VMPP','VAPP') and T1.CO_SENTENCEID = T2.CO_SENTENCEID
      and T1.CO_ID < T2.CO_ID)),
      MAP2 as (select T1.CO_SENTENCEID, T1.CO_ID CO_VORNE, T2.CO_ID
      CO_HINTEN from TB_TT_MORPHO T1, TB_TT T2 where T1.CO_MORPHO
      = 'PWAV' and T2.CO_TOKEN = '?' and T1.CO_SENTENCEID = T2.
      CO_SENTENCEID)
select unique T1.CO_SENTENCEID
from MAP1 T1, MAP2 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_VORNE = T2.CO_
VORNE and T1.CO_HINTEN < T2.CO_HINTEN;
```

Tabelle 77 dokumentiert, dass die alternative Abfragevariante ähnlich positiv wie unsere Ausgangsrecherche skaliert. Die Steigerungsfaktoren für Abfragezeiten bleiben unvermindert unter denen der Token- bzw. Beleganzahl. Allerdings lassen sich diesmal durch die Modifikation des Abfragealgorithmus kaum nennenswerte Beschleunigungen der Abfragezeit beobachten. Von der Modifikation des Suchalgorithmus profitieren allein die beiden größten Evaluationskorpora auf beiden Systemen (vgl. Tab. 76 und Abb. 60), und selbst in diesen Fällen deuten die überlappenden Konfidenzintervalle auf fehlende Signifikanz hin. Für die kleineren Korpora konstatieren wir wechselweise leichte Verlängerungen sowie leichte Verringerungen der Laufzeiten, ebenfalls ohne Signifikanzen.

[153] Im konkreten Fall fällt diese Einschränkung nicht ins Gewicht, weil das Interrogativpronomen am Satzanfang stehen soll und folglich keine ungewollten Ausschlüsse zu erwarten sind. Aus diesem Grund wäre auch eine Vertauschung der beiden ersten Suchkriterien denkbar. An der grundsätzlichen Pflicht zur korrekten Pärchenbildung mit NOT-Kriterien, um die Streichung unproblematischer Vorkommen zu vermeiden, ändert sich dadurch nichts.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	3	3,31 (+0,96)	1,68 (+0,38)
EK-2	125	66,41 (+9,63)	45,98 (+1,66)
EK-3	738	211,85 (+0,15)	134,86 (-2,05)
EK-4	1.277	321,85 (+9,86)	222,61 (-2,66)
EK-5	2.595	523,62 (-10,86)	395,91 (-13,37)
EK-6	4.811	728,12 (-29,63)	546 (-23,79)

Tab. 76: Mittelwerte und Änderungen der optimierten Abfragezeiten für Abfrage 5 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 41,7	Faktor 20,06	Faktor 27,37
EK-3	Faktor 10	Faktor 5,9	Faktor 3,19	Faktor 2,93
EK-4	Faktor 2	Faktor 1,73	Faktor 1,52	Faktor 1,65
EK-5	Faktor 2	Faktor 2,03	Faktor 1,63	Faktor 1,78
EK-6	Faktor 2	Faktor 1,85	Faktor 1,39	Faktor 1,38

Tab. 77: Steigerungsfaktoren für optimierte Abfrage 5

5.3.4 Neuevaluation Abfrage 6

Als erste Abfrage der Optimierungsreihe referiert Abfrage 6 nicht allein auf lineare Wortverkettungen, sondern zusätzlich auf textspezifische Metadaten. Passende Belege enthalten movierte Anredeformen (ohne Eigennamen) zur expliziten Kennzeichnung des weiblichen Geschlechts, beschränkt auf die thematische Domäne „Politik/Wirtschaft/Gesellschaft“ sowie Texte des laufenden 21. Jahrhunderts. Insgesamt sollen sechs Suchkriterien abgearbeitet werden:

1) Token „Frau“
2) Token „*in“
3) Connexor-Wortklasse „N“
4) Connexor-Subkategorie „Prop“

5) Domäne 4
6) Datum >= 2000

Im Vorfeld der Testläufe wurde für die Abfrage eine geeignete Strategie hinsichtlich der Einbeziehung hierarchisch übergeordneter Metadaten ermittelt; vgl. Abschnitt 5.2.3. Dabei wurde evaluiert, ob mehrere Textmetadaten-Spezifizierungen in einem Mapping zusammengefasst oder aber separat behandelt und an wortbezogene Blätter gekoppelt werden sollten. Im vorliegenden Fall erbrachte die Gegenüberstellung beider Varianten ein eindeutiges Ergebnis: Die Kombination von Textmetadaten in einem gemeinsamen Mapping generierte durchgängig niedrigere Laufzeiten als jede andere Pärchenbildung, sie soll deshalb für unsere Testreihe als gesetzt gelten.

Dadurch reduziert sich die Anzahl der noch frei anordenbaren Suchkriterien auf vier. Wie bereits bei Abfrage 5 gilt es, bereits in der Map-Phase den korrekten Abstand (hier: null Wörter) des NOT-Kriteriums zu einem Vorgänger-/Nachfolgekriterium zu verifizieren. Aufgrund der weiterhin geforderten Positionsidentität der Kriterien zwei und drei (Tokenendung auf „*in“, Wortklasse „Nomen“) ist dies auf mehreren Wegen möglich. Unser Algorithmus entscheidet sich für die Variante ([1:3] [2:4] [5:6]), da auf diese Weise das Kriterium mit den höchsten sowie das Kriterium mit den niedrigsten Trefferfrequenzen kombiniert werden.

Die Abfrage erfordert zwei aufeinander folgende Reduce-Schritte:

```
insert into TB_REDUCE1 (CO_SENTENCEID, CO_ID_FIRST, CO_ID_LAST)
with
      MAP1 as (select T1.CO_SENTENCEID, T1.CO_ID CO_VORNE, T2.CO_ID
      CO_HINTEN from TB_CONNEXOR T1, TB_CONNEXOR_MORPHO_N T2 where
      T1.CO_TOKEN='Frau' and T1.CO_SENTENCEID = T2.CO_SENTENCEID),
      MAP2 as (select T1.CO_SENTENCEID, T1.CO_ID CO_VORNE, T1.CO_ID
      CO_HINTEN from TB_CONNEXOR T1 where reverse(T1.CO_TOKEN) like
      reverse('%in') and not exists (select null from TB_CONNEX-
      OR_MORPHO_N_PROP T2 where T1.CO_SENTENCEID = T2.CO_SENTENCEID
      and T1.CO_ID = T2.CO_ID))
select T1.CO_SENTENCEID
from MAP1 T1, MAP2 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_VORNE=T2.CO_
VORNE-1 and T2.CO_VORNE = T1.CO_HINTEN;

insert into TB_REDUCE2 (CO_SENTENCEID)
with
      MAP3 as (select T5.CO_SENTENCEID from TB_CONNEXOR_SENTENCE T5
      where T5.CO_TEXTID in (select CO_TEXTID from TB_TEXT T6 where
      T6.CO_DOMAIN=4 and T6.CO_YEAR >= 2000))
select unique T1.CO_SENTENCEID
from MAP3 T1, TB_REDUCE1 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID;
```

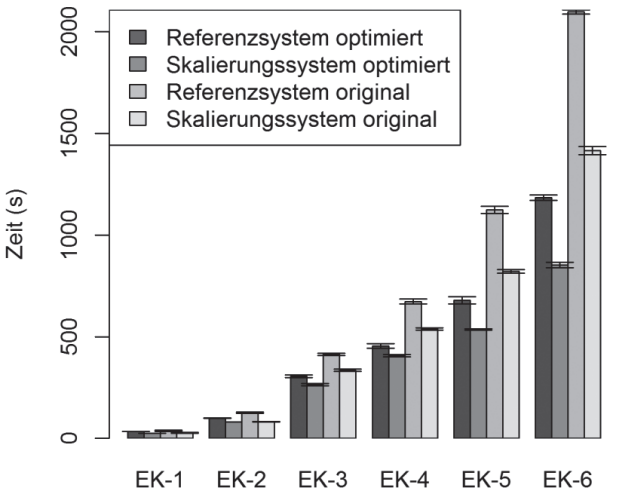


Abb. 61: Vergleich der Abfragezeiten für Abfrage 6 (optimiert, original)

Die in Abbildung 61 bzw. Tabelle 78 nebeneinander gestellten Messwerte von originaler und modifizierter Abfrage dokumentieren zum ersten Mal einen durchgängigen Vorteil für die problemorientierte Map-Reduce-Variante in allen Evaluationskorpora. Sogar bei den kleinsten Korpusvolumen ist eine Beschleunigung der Abfragezeit messbar; bei den größeren Korpora liegt diese Verbesserung bei über einem Drittel des Ausgangswerts. Die in Tabelle 79 errechneten Steigerungsfaktoren der Abfragezeiten bewegen sich in allen Fällen unter denen der Token- bzw. Belegsteigerung.

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	7	34,39 (-3,84)	25,6 (-1,8)
EK-2	2.291	100,21 (-26,56)	80,49 (-1,88)
EK-3	38.783	305,35 (-108,83)	265,34 (-70,22)
EK-4	81.005	454,56 (-218,30)	406,81 (-131,56)
EK-5	145.733	678,79 (-444,48)	536,76 (-283,94)
EK-6	281.129	1.183,89 (-912,88)	852,26 (-563,59)

Tab. 78: Mittelwerte und Änderungen der optimierten Abfragezeiten für Abfrage 6 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 327,29	Faktor 2,91	Faktor 3,14
EK-3	Faktor 10	Faktor 16,93	Faktor 3,05	Faktor 3,3
EK-4	Faktor 2	Faktor 2,09	Faktor 1,49	Faktor 1,53
EK-5	Faktor 2	Faktor 1,8	Faktor 1,49	Faktor 1,32
EK-6	Faktor 2	Faktor 1,93	Faktor 1,74	Faktor 1,59

Tab. 79: Steigerungsfaktoren für optimierte Abfrage 6

5.3.5 Neuevaluation Abfrage 8

Als finaler Evaluationsgegenstand unserer problemorientierten Segmentierung kombiniert Abfrage 8 wortspezifische Suchkriterien mit deren Einbettung in eine übergeordnete Wortgruppe sowie mit Frequenzwerten. Passende Belegsätze bestehen aus einer Partizipialphrase mit einem aus dem Verb *sehen* gebildeten Adjektiv (Partizip I) oder einer als Adjektiv gebrauchten Verbform (Partizip II; beides Xerox-Wortklasse „ADJ“). Diese Konstruktion soll Teil einer Adjektivphrase sein und unmittelbar von einem niederfrequenten Nomen gefolgt werden.

Insgesamt gilt es folgende fünf Suchkriterien zu kombinieren:

1) Xerox-Wortklasse „ADJ“
2) Lemma „sehen“
3) Xerox- Adjektivphrase „AP“ in Xerox-Knotentabelle
4) Xerox-Wortklasse „NOUN“
5) Frequenzklasse > 9 in Xerox-Lemmaliste

Unter dem Gesichtspunkt der Häufigkeitsverteilungen wäre Mapvariante 11 mit den Pärchen ([1:5][2:4][3:3]) eine ideale Segmentierung, da sie die mit Abstand umfangreichste Wortklassenkategorie „NOUN“ mit der hochselektiven Lemmaausprägung „sehen“ kombiniert. Da diese Lösung aber nicht sämtliche Abstandsvorgaben verifiziert (vgl. Abschnitt 5.2.2), wählt unser Algorithmus die zweitbeste Mapvariante 3 mit den Pärchen ([1:2] [3] [4:5]) aus. Auf diese Weise lassen sich gleichermaßen die Abfrage der Xerox-Knotentabelle sowie die Einschränkung aus der Xerox-Lemmaliste an die maßgeblichen Wortpositionen koppeln:

```
insert into TB_REDUCE1
with
      MAP1 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T2.
      CO_ID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-WORTTABELLE> T2
      where T1.CO_POS='ADJ' and T2.CO_LEMMA = 'sehen' and T1.CO_SEN-
      TENCEID = T2.CO_SENTENCEID and T1.CO_ID = T2.CO_ID-1),
      MAP2 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T1.
      CO_ID CO_ID_LAST from <EK-KNOTENTABELLE> T1 where T1.CO_PARENT
      = 'AP' and T1.CO_SENTENCEID = T2.CO_SENTENCEID)
select T1.CO_SENTENCEID, T1.CO_ID_FIRST, T2.CO_ID_LAST
from MAP1 T1, MAP2 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID_LAST =
T2.CO_ID_FIRST;

insert into TB_REDUCE2 (CO_SENTENCEID)
with
      MAP3 as (select T1.CO_SENTENCEID, T1.CO_ID CO_ID_FIRST, T1.
      CO_ID CO_ID_LAST from <EK-WORTTABELLE> T1, <EK-LEMMALISTE> T2
      where T1.CO_POS = 'NOUN' and T1.CO_LEMMA = T2.CO_LEMMA and
      T2.CO_FREQCLASS > 9)
select unique T1.CO_SENTENCEID from MAP3 T1, TB_REDUCE1 T2
where T1.CO_SENTENCEID = T2.CO_SENTENCEID and T1.CO_ID_FIRST-1 =
T2.CO_ID_LAST;
```

Korpus	Belege	Abfragezeit Referenzsystem	Abfragezeit Skalierungssystem
EK-1	2	66,5 (+33,2)	44,25 (+32,98)
EK-2	131	167,06 (+51,03)	108,96 (+38,7)
EK-3	1.294	462,63 (-63,67)	355,67 (-42,37)
EK-4	2.541	749,17 (-92,68)	542,49 (-82,06)
EK-5	5.621	1.169,88 (-171,33)	917,57 (-143,78)
EK-6	11.242	2.225,36 (-335,90)	1.692,32 (-234,99)

Tab. 80: Mittelwerte und Änderungen der optimierten Abfragezeiten für Abfrage 8 in Sekunden

Korpus	Steigerung Tokenanzahl	Steigerung Beleganzahl	Steigerung Abfragezeit Referenzsystem	Steigerung Abfragezeit Skalierungssystem
EK-2	Faktor 100	Faktor 65,5	Faktor 2,51	Faktor 2,46
EK-3	Faktor 10	Faktor 9,88	Faktor 2,77	Faktor 3,26
EK-4	Faktor 2	Faktor 1,96	Faktor 1,62	Faktor 1,53
EK-5	Faktor 2	Faktor 2,21	Faktor 1,56	Faktor 1,69
EK-6	Faktor 2	Faktor 2	Faktor 1,9	Faktor 1,84

Tab. 81: Steigerungsfaktoren für optimierte Abfrage 8

Nennenswerte Auswirkungen auf die gemessenen Abfragezeiten (vgl. Tab. 80 und Abb. 62) lassen sich für Korpusgrößen von mehreren Milliarden Wortformen erkennen; allerdings erreichen diese bei weitem nicht die bei Abfrage 3, 4 oder 6 beobachteten Ausmaße. Kleinere Korpora reagieren sogar mit Laufzeitverlängerungen auf die Überführung der „all in one"-Abfrage in zwei aufeinander aufbauende Reduce-Schritte. Positiv bleibt die durchgängig unterproportionale Entwicklung der Steigerungsfaktoren in Tabelle 81.

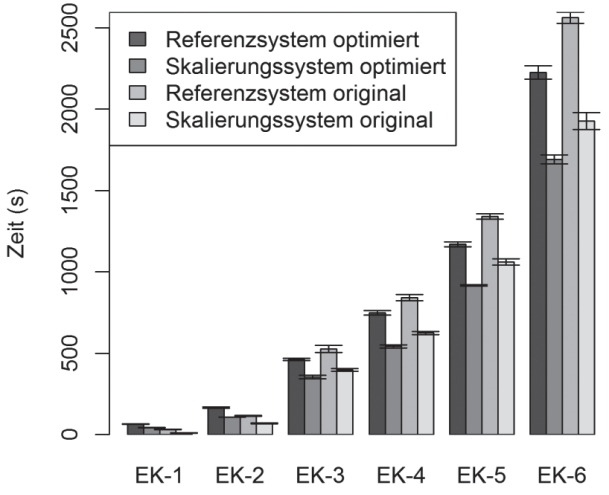


Abb. 62: Vergleich der Abfragezeiten für Abfrage 8 (optimiert, original)

6. Integration in ein Online-Framework

Die evaluierten Typen von Korpusrecherchen lassen sich – unabhängig davon, ob es sich um „all in one"-Statements mit mehreren Joins oder um MR-Varianten mit segmentierten Suchkriterien handelt – prototypisch durch datenbankintern gespeicherte Prozeduren (*stored procedures*) realisieren. Angereichert um webtypische Navigations-, Interaktions- und Präsentationselemente entsteht auf diese Weise eine browsergestützte, plattformunabhängige Abfrageoberfläche.[154] Die Generierung sämtlicher Recherche- und Ergebnisseiten erfolgt dynamisch. Funktional entspricht dieser Ansatz der in der Software-Entwicklung klassischen Drei-Schichten-Architektur (*three tier architecture*) mit der Datenbank als Backend (*data tier*) für das persistente Speichern der Korpusinhalte, einem Datenbank-Web-Gateway als zwischengeschalteter Schicht für die Anwendungslogik (*middle tier*) und dem Web-Browser als Frontend für Benutzereingaben und Ergebnispräsentation (*client tier*). Aufgrund der Nutzung von Stored Procedures existiert eine marginale technische Kopplung zwischen Backend und Logikschicht. Durch eine strikte Trennung der Software- und Daten-Schemata kann diese Kopplung allerdings aufgefangen werden, so dass keinerlei Abhängigkeitsprobleme hinsichtlich der Pflege, Weiterentwicklung oder Skalierbarkeit einzelner Schichten zu erwarten sind.

6.1 Suchformulare

Dynamische Web-Formulare bedienen in unserem prototypischen Online-Framework die musterbasierte Suche nach Belegsätzen unter potenzieller Verwendung sämtlicher verfügbarer Annotationsebenen bzw. Metadaten. Sie erlauben das sukzessive Hinzufügen von Suchkriterien vom Typ Token, Lemma, Wortklasse und Position (z.B. Satzanfang, Satzende). Die syntaktisch korrekte Eingabe von Suchspezifikationen wird unterstützt, ohne dass vorab eine spezielle Abfragesprache erlernt werden müsste. Sobald die Recherche gestartet wird, übernimmt ein internes Datenbankskript die Segmentierung der Suchkriterien sowie das dynamische Formulieren und Ausführen der passenden SQL-Statements. Auf diesem Wege lässt sich ein Großteil der in der Forschungspraxis anfallenden Aufgaben mit einem intuitiven Frontend abdecken; für komplexere Kombinationen von Suchkriterien bieten sich direkte SQL-Recherchen auf dem Datenbestand an.

[154] Die hier vorgestellte Online-Oberfläche ist als Machbarkeitsstudie zu verstehen. Denkbar sind darüber hinaus insbesondere programmierte Schnittstellen zu existierenden Forschungsinfrastrukturen, etwa im Kontext computerlinguistischer Initiativen wie CLARIN oder DARIAH.

In einem Formular für die wortbezogene Korpusrecherche auf Satzebene lassen sich Werte für Token und Lemma unter Beachtung der korrekten Groß-/Kleinschreibung frei eingeben. Die Platzhalteroperatoren * und ? werden automatisch in die SQL-Äquivalente % bzw. _ überführt. Bei der Auswahl von Wortklassenbezeichnern und Positionsangaben helfen inkrementelle Drop-Down-Listen. Als Verknüpfungsoperatoren stehen die beiden Ausdrücke „und gefolgt von" sowie „nicht gefolgt von" zur Verfügung, letzterer für die Realisierung von Ausschlussbedingungen (NOT-Kriterien).[155]

Minimale und maximale Wortabstände zwischen Suchkriterien können explizit angegeben werden. Bleiben die entsprechenden Felder leer, so wird als Maximalabstand die Spanne bis zum Ende des jeweiligen Suchsegments (= Satzes) angenommen. Möglich sind an dieser Stelle auch negative Werte: Ein Minimalabstand von „-5" legt fest, dass das folgende Suchkriterium in der linearen Wortfolge eines Satzes mindestens fünf Positionen weiter vorne lokalisiert werden soll. Das Eintragen einer Null als Minimal- und Maximalwert impliziert Positionsidentität – auf diese Weise lassen sich Suchmuster wie „Lemma das am Satzanfang" oder „Nomina auf *in" ausdrücken.

Abbildung 63 zeigt eine für Katalogabfrage 3 passend ausgefüllte Suchmaske. Da in diesem Fall keine Einschränkungen hinsichtlich der textspezifischen Metadaten gefordert sind, bleiben die entsprechenden Optionsknöpfe im Formular maximal selektiert. Nach Terminierung der durch den MR-Algorithmus generierten SQL-Statements erhält der Benutzer eine Rückmeldung zur Anzahl der gefundenen Belegsätze, die er wahlweise inspizieren oder – in Form einer Satznummernliste – in eine wiederverwendbare Ergebnistabelle abspeichern kann. Anhand der Satznummern lassen sich einzelne Belege jederzeit rekonstruieren und zusammen mit der passenden Textsigle, die ihrerseits als Zeiger zu textspezifischen Metadaten dient, präsentieren (vgl. Abb. 64).

[155] Analog dazu lassen sich auch *oder*-Verknüpfungen implementieren, was jedoch für die Umsetzung der Abfragen aus unserem Referenzkatalog nicht erforderlich war. Alternativ können gespeicherte Anfragen mit variierenden Kriteriumsausprägungen zusammengefasst werden (siehe nachfolgender Abschnitt).

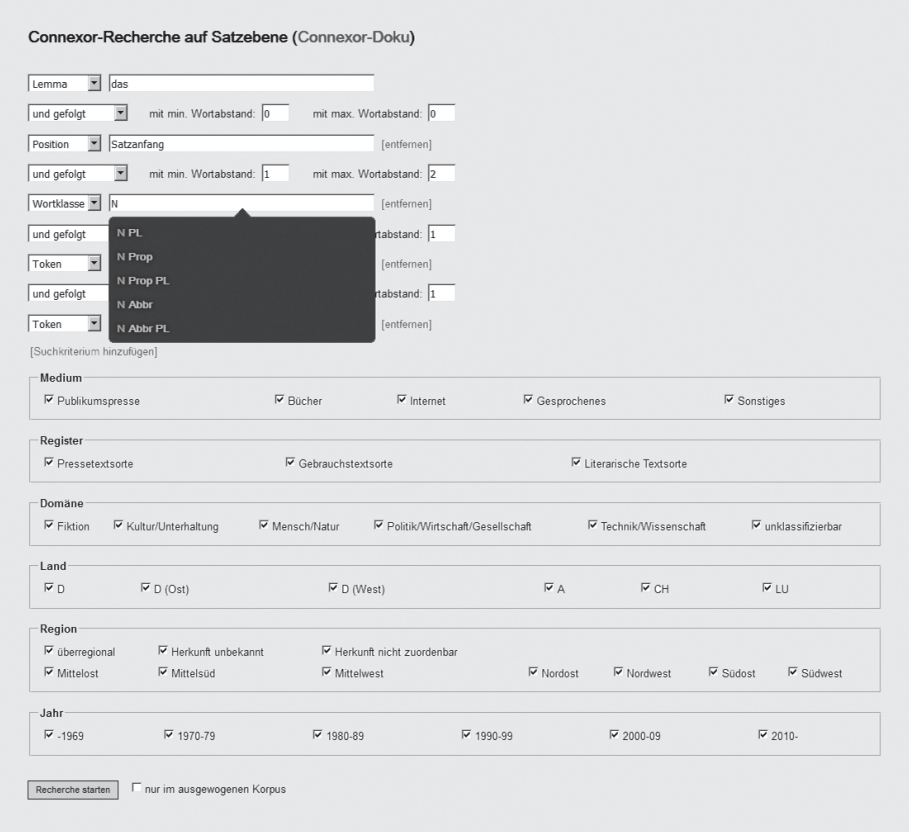


Abb. 63: Ausgefülltes Online-Suchformular für Abfrage 3

Text-ID	Beleg
E96/SEP.23366	Das Radio Schwalbe , was Agatashya in Kinyarwanda bedeutet , war i m Sommer 1994 von der Schweizer Sektion der Reporters sans frontières gegründet worden und spielte eine wichtige Rolle als mehr oder weniger einzige Informationsquelle der 1,2 Millionen Menschen , die aus Ruanda nach Zaire geflüchtet waren .
F99/911.68722	Das Geld , was der drohenden Zinssteuer zuvorkommen will , kann über die Schweiz durchgeleitet und in alle Welt dirigiert werden .
DPA10/SEP.11761	Das Nächste , was ich weiß , ist , dass ich getroffen wurde .
NUN06/AUG.03054	Das Geld , was jetzt schon " vermasselt ist , hätte wohl für den vorzeitigen Ruhestand fast gereicht , und wer weiß , was noch folgt .
NON07/SEP.00169	Das Nachdenken , was jetzt kommt , hat dann i m zweiten Monat eingesetzt .
F01/105.32671	Das einzige Zeichen , was auf sie schließen läßt , sind die Zeitungen der Partei , die auf den Tischen umherliegen und von dem Wirth für seine Gäste gehalten werden .
HAZ08/APR.04963	Das Glück , was den Bayern fehlte , hatten die Gäste .

Abb. 64: Belegsätze einer Korpusrecherche mit Textsiglen

Für die gezielte Inspektion von Belegsätzen extrahiert ein Datenbankskript auf Mausklick die relevanten Segmentinformationen (Token, Lemma, Wortklasse) aus den Korpustabellen und setzt diese zu einer spaltenorientierten Übersicht zusammen (vgl. Abb. 65).[156]

TOKEN	Das	Glück	,	was	den	Bayern	fehlte	,	hatten	die	Gäste	.
LEMMA	das	glück	,	was	die	Bayer	fehlen	,	haben	die	gast	.
WORTART	DET	N	P	PRON	DET	N	V	P	V	DET	N	P

Abb. 65: Anzeige eines Belegsatzes mit Segmentgrenzen und Annotationsdaten

6.2 Speicherung von Beleglisten

Eine nachhaltige Bereitstellung von Rechercheergebnissen umfasst nicht allein die persistente Speicherung von Trefferzahlen, sondern die Dokumentation sämtlicher verwendeter Suchparameter sowie gegebenenfalls von Suchraum-Einschränkungen. Weiterhin relevant sind eindeutige Belegverweise und quantitative Daten zur Verteilung der Ergebnisse unter Berücksichtigung der diversen Metadaten-Ausprägungen. Um eine effektive Anbindung an nachgeordnete Auswertungen zu ermöglichen, speichert unser prototypisches Recherche-Frontend all diese Angaben in benutzerspezifischen Ergebnistabellen. Sichten auf Einzelbelege lassen sich anschließend wie oben illustriert „on the fly“ generieren.

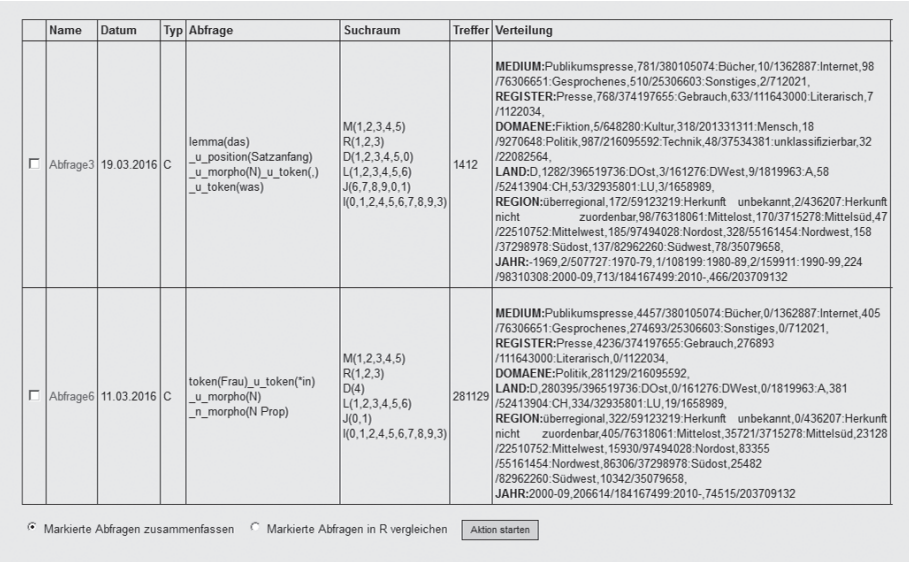
Abbildung 66 dokumentiert exemplarisch die gespeicherten Rechercheergebnisse der Katalogabfragen 3 und 6. Im Einzelnen werden folgende Informationen archiviert:

- Name: frei vergebbare Benennung der Korpusabfrage
- Datum: Zeitpunkt der Abfrage
- Typ: für die Abfrage verwendetes Annotationswerkzeug (C=Connexor, T=TreeTagger, X=Xerox)
- Abfrage: Suchkriterien und deren Verknüpfungsoperatoren in einer properitären Kodierung
- Suchraum: Festlegung des Suchraums durch textspezifische Metadaten, properitäre Kodierung der erlaubten Ausprägungen von Medium (M), Register (R), Domäne (D), Land (L), Region (I), Jahr (J), sowie ggf. Angabe des

[156] Für die effektive Durchführung dieser Operationen werden in der Datenbank zusätzliche Indizes auf den Satz-ID-Spalten benötigt, die nicht Gegenstand unserer bisherigen Evaluierungen waren.

durchsuchten Korpus (z.B. UK = Untersuchungskorpus, AK = ausgewogenes Korpus)

– Treffer: Anzahl der ermittelten Belegsätze
– Verteilung: Verteilung der Belege unter Berücksichtigung der verschiedenen Metadatenausprägungen (Anzahl der Belege pro Ausprägung/Gesamtvolumen pro Ausprägung)



	Name	Datum	Typ	Abfrage	Suchraum	Treffer	Verteilung
☐	Abfrage3	19.03.2016	C	lemma(das) _u_position(Satzanfang) _u_morpho(N)_u_token(,) _u_token(was)	M(1,2,3,4,5) R(1,2,3) D(1,2,3,4,5,0) L(1,2,3,4,5,6) J(6,7,8,9,0,1) I(0,1,2,4,5,6,7,8,9,3)	1412	**MEDIUM:**Publikumspresse,781/380105074:Bücher,10/1362887:Internet,98/76306651:Gesprochenes,510/25306603:Sonstiges,2/712021, **REGISTER:**Presse,768/374197655:Gebrauch,633/111643000:Literarisch,7/1122034, **DOMAENE:**Fiktion,5/648280:Kultur,318/201331311:Mensch,18/9270648:Politik,987/216095592:Technik,48/37534381:unklassifizierbar,32/22082564, **LAND:**D,1282/396519736:DOst,3/161276:DWest,9/1819963:A,58/52413904:CH,53/32935801:LU,3/1658989, **REGION:**überregional,172/59123219:Herkunft unbekannt,2/436207:Herkunft nicht zuordenbar,98/76318061:Mittelost,170/3715278:Mittelsüd,47/22510752:Mittelwest,185/97494028:Nordost,328/55161454:Nordwest,158/37298978:Südost,137/82962260:Südwest,78/35079658, **JAHR:**-1969,2/507727:1970-79,1/108199:1980-89,2/159911:1990-99,224/98310308:2000-09,713/184167499:2010-,466/203709132
☐	Abfrage6	11.03.2016	C	token(Frau)_u_token(*in) _u_morpho(N) _n_morpho(N Prop)	M(1,2,3,4,5) R(1,2,3) D(4) L(1,2,3,4,5,6) J(0,1) I(0,1,2,4,5,6,7,8,9,3)	281129	**MEDIUM:**Publikumspresse,4457/380105074:Bücher,0/1362887:Internet,405/76306651:Gesprochenes,274693/25306603:Sonstiges,0/712021, **REGISTER:**Presse,4236/374197655:Gebrauch,276893/111643000:Literarisch,0/1122034, **DOMAENE:**Politik,281129/216095592, **LAND:**D,280395/396519736:DOst,0/161276:DWest,0/1819963:A,381/52413904:CH,334/32935801:LU,19/1658989, **REGION:**überregional,322/59123219:Herkunft unbekannt,0/436207:Herkunft nicht zuordenbar,405/76318061:Mittelost,35721/3715278:Mittelsüd,23128/22510752:Mittelwest,15930/97494028:Nordost,83355/55161454:Nordwest,86306/37298978:Südost,25482/82962260:Südwest,10342/35079658, **JAHR:**2000-09,206614/184167499:2010-,74515/203709132

◉ Markierte Abfragen zusammenfassen ○ Markierte Abfragen in R vergleichen Aktion starten

Abb. 66: Archivierung von Rechercheergebnissen

Über die Option „Markierte Abfragen zusammenfassen“ lassen sich zwei oder mehrere Beleglisten vereinen. Voraussetzung hierfür ist deren Übereinstimmung hinsichtlich Abfragetyp und Suchraum, um eine Aufrechterhaltung der statistischen Aussagekraft zu gewährleisten. Auf diese Weise können Recherchen, ausgehend von einem Basiskriterium, sukzessive ausgebaut werden. Weiterhin begegnen wir auf diese Weise dem Umstand, dass unser Suchformular-Prototyp kein Boolesches ODER (z.B. Suche nach Token *frug* oder *fragte*) und keine Optionalitätsspezifikation (z.B. Suche nach Artikeln und Substantiven, die entweder unmittelbar aufeinander folgen oder optional durch ein Adjektiv getrennt sind) anbietet. Die entsprechenden Recherchen können ersatzweise nacheinander ausgeführt und die gefundenen Treffer abschließend zusammengefasst werden.

6.3 Schnittstellen zu Statistikwerkzeugen

Empirisch arbeitende Linguisten beurteilen die Ergebnisse von Korpusrecherchen häufig unter Zuhilfenahme statistischer Methoden und Werkzeuge. Als Gegenentwurf zur intuitiven Inspektion sollen auf diese Weise ungerechtfertigte Schlussfolgerungen vermieden werden; vgl. Abschnitt 2.1. Typische Einsatzbereiche sind Fälle, in denen es abzuklären gilt, ob gemessene Frequenzunterschiede zwischen sprachlichen Varietäten tatsächlich signifikant oder lediglich einem Stichprobenzufall geschuldet sind (vgl. Bubenhofer et al. 2014, S. 125ff.), ob sich Entwicklungstendenzen herausarbeiten lassen oder keine eindeutige Richtung erkennbar ist etc. Ähnliches gilt bei der Beurteilung von Belegstreuungen: Betrachtet man ausschließlich absolute Trefferzahlen, ist die Aussagekraft einer Recherche begrenzt. Die gefundenen Belege könnten sich mehr oder weniger gleichmäßig über unterschiedliche Regionen, Textsorten und Zeiträume verteilen. Ebenso gut könnten sie aber auch nur einem einzelnen Autor zuordenbar sein, im Extremfall nur einem einzigen Text. Erst durch genaueres Hinschauen, also durch Inspektion der textspezifischen Metadatenverteilung, lassen sich solche Unsicherheiten auflösen. Insbesondere die Variationslinguistik bedient sich darüber hinaus sogenannter multivarianter Analyseverfahren (Szmrecsanyi 2013, S. 269) zur Bestimmung derjenigen (inner- und außersprachlichen) Faktoren, die einen statistisch messbaren Einfluss auf die sprachliche Realität ausüben.

Allgemeine Information zu den Abfragen | MEDIUM | REGISTER | LAND | REGION | DOMAENE | JAHR | aggregierte Daten

MEDIUM

R-Code zur Tabellenerzeugung

```
TABLE <- rbind(
   c(10),
   c(510),
   c(98),
   c(781),
   c(2)
)

rownames(TABLE) <- c('Buecher', 'Gesprochenes', 'Internet', 'Publikumspresse', 'Sonstiges')
colnames(TABLE) <- c('Abfrage3')



corpus.TABLE <- rbind(
   c(1362887),
   c(25306603),
   c(76306651),
   c(380105074),
   c(712021)
)

rownames(corpus.TABLE) <- c('Buecher', 'Gesprochenes', 'Internet', 'Publikumspresse', 'Sonstiges')
colnames(corpus.TABLE) <- c('Abfrage3-Korpus')
```

Abb. 67: Automatisiert erstellter R-Code zur Analyse der Medienverteilung von Abfrage 3

Unser exemplarisches Online-Framework stellt für weiterführende empirische Auswertungen eine webbasierte Schnittstelle zu KoGra-R (vgl. Hansen/Wolfer 2016 sowie www.ids-mannheim.de/kogra-r/) bereit. KoGra-R ist ein öffentlich nutzbares Werkzeug zur Durchführung standardisierter statistische Verfahren für korpusbasierte Häufigkeiten auf Basis des statistischen Softwarepakets R (R Core Team (Hg.) 2016). Implementiert sind Analysen für Einzelabfragen ebenso wie vergleichende Auswertungen zweier oder mehrerer Korpusrecherchen. Zum Inventar der abgedeckten Funktionalitäten gehören Diagramme mit Konfidenzintervallen für absolute und relative Häufigkeiten, ein Chi-Quadrat-Test zur Überprüfung der Datenverteilung, Phi Assoziationsstärke/Cramérs V Assoziationsstärke für die Berechnung des Einflusses verschiedener Variablen, Assoziations- und Mosaikplots sowie Dispersionsmaße wie Gries' DP Norm (Gries 2008, 2010) für die Beschreibung der Streuung von Belegdaten.

Das statistische Werkzeug lässt sich für gespeicherte Beleglisten aktivieren und übergibt die in der Datenbank hinterlegten Metadatenverteilungen (vgl. Abb. 66) an die programmierte R-Schnittstelle. Für jeden Metadatentyp liefert KoGra-R den R-Code zur Erzeugung von Tabellen für Rohdaten (vgl. Abb. 67), normierte und relative Werte. Daran anschließend folgen die Ergebnisse einzelner Testverfahren (vgl. Abb. 68) sowie verschiedene Diagrammtypen.

Chi Quadrat Test

```
        Chi-squared test for given probabilities

data:  treffer.tab
X-squared = 1738.789, df = 4, p-value < 2.2e-16
```

Chi Quadrat Test, erwartete Häufigkeiten

```
[1] 280.2 280.2 280.2 280.2 280.2
```

Chi Quadrat Test, Residuen

```
[1] -16.14177  13.72827 -10.88465  29.91784 -16.61970
```

Konfidenzintervalle

```
=== Abfrage3 ===
              Anteil in Prozent Konf.-Intervall unten Konf.-Intervall obe
Buecher               0.7137759             0.2729628            1.154588
Gesprochenes         36.4025696            33.8830669           38.922072
Internet              6.9950036             5.6594044            8.330602
Publikumspresse      55.7458958            53.1450641           58.346727
```

Abb. 68: Statistische Online-Tests für Abfrage 3

Das Konfidenzintervall-Diagramm in Abbildung 69 visualisiert die prozentuale Verteilung der Treffer (relative Werte) in Form von Säulenelementen. Wurden mehrere Abfragen zusammengefasst, generiert das Skript eine Säulengruppe. Diese ist definiert über die Ausprägungen des jeweiligen Metadatums; innerhalb jeder Säulengruppe befindet sich dann pro Abfrage eine Säule. Die eingezeichneten Konfidenzintervalle helfen bei der Interpretation von Aussagen über die Stichprobenergebnisse hinsichtlich der Grundgesamtheit. Die Intervallgrenzen geben mit einer vorab festlegbaren Sicherheit (im illustrierten Fall 95%) an, in welchem Bereich sich die Populationswerte befinden. Nur wenn sich die Intervallgrenzen einer Säule bzw. Säulengruppe nicht mit den Grenzen einer anderen Diagrammsäule überschneiden, deutet dies auf einen signifikanten Unterschied zwischen den Metadaten-Ausprägungen hin.

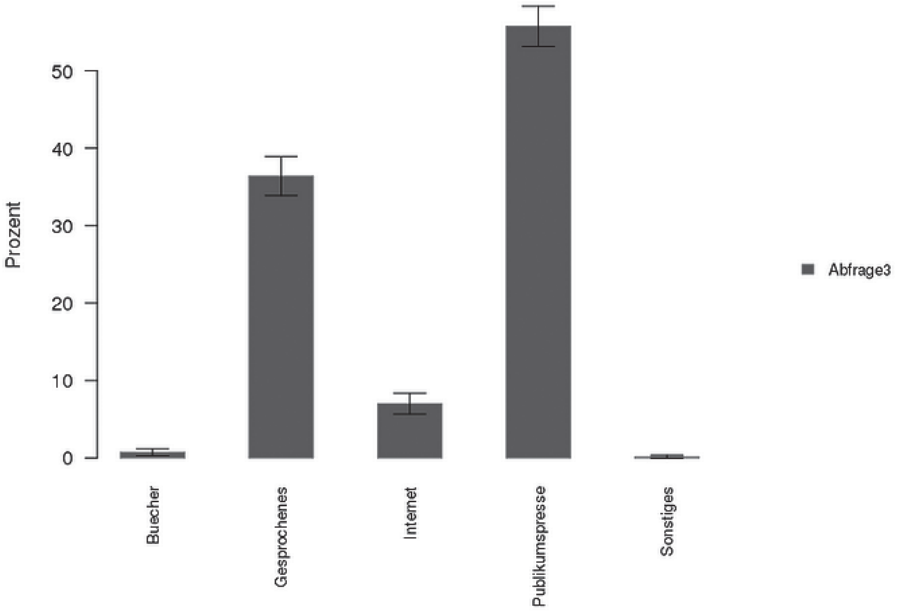


Abb. 69: Konfidenzintervalle-Diagramm für Metadatenverteilung von Abfrage 3

6.4 Übersichtslisten

Eine vergleichsweise unspektakuläre, aber für die praktische Arbeit sinnvolle Funktionalität des Online-Frameworks besteht in der Bereitstellung von Hintergrundinformationen zu Korpusspezifika, insbesondere zur Stratifikation oder zu Eigenschaften (Häufigkeiten, Längen etc.) einzelner Segmente. Hierzu zählen Übersichtslisten für Token, Lemmata, Wortklassen und -gruppen,

separiert nach Tagging-Werkzeug. Manche dieser Informationen lassen sich ad hoc generieren, andere liegen in Form von Lookup-Tabellen fest kodiert in der Datenbank. Abbildung 70 demonstriert die Präsentation der in Kapitel 3 vorgestellten Lemmatabelle TB_CONNEXOR_LEMMALIST. Benutzer des Frontends können an dieser Stelle quantitative Lemmadaten abrufen und diese für die Planung zukünftiger Korpusrecherchen einsetzen. Andererseits dienen die in den Lookup-Tabellen hinterlegten Frequenzen dem Suchalgorithmus als Entscheidungsgrundlage für die Segmentierung komplexer Abfragen. Ähnliches gilt für die Charakteristika virtueller Subkorpora auf Basis textspezifischer Metadaten: Auch diese einmalig gezählten Häufigkeiten lassen sich gleichermaßen durch Übersichtslisten einsehbar machen und bei Bedarf an online verknüpfte Werkzeuge für die Durchführung statistischer Tests und Visualisierungen übermitteln.

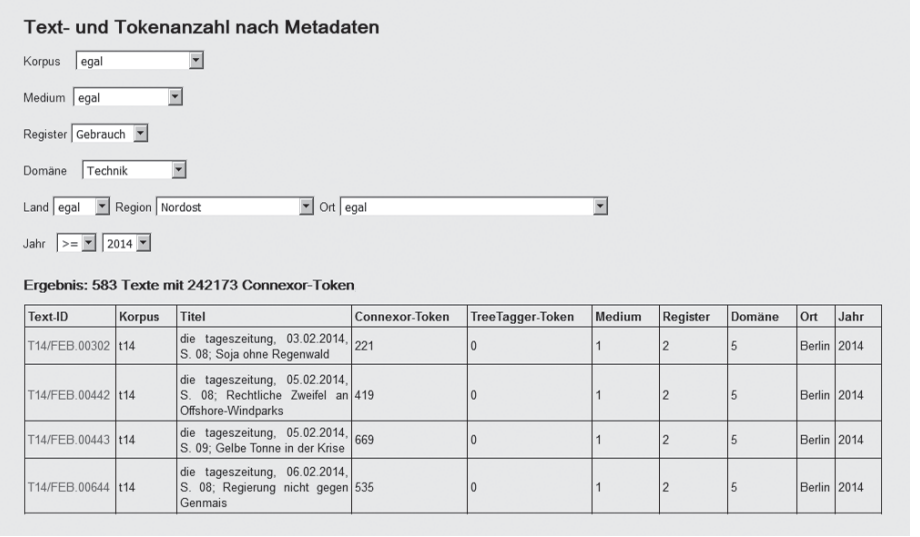
Connexor Lemmaliste

Liste: Connexor Lemmaliste Einschränkung (Wildcards erlaubt): Sortiert nach: Frequenz

[<<] 0-999/31371441 [>>]

Frequenz	Länge	Connexor Lemma
406007796	3	die (HK 1)
385661278	1	. (HK 1)
383095591	1	, (HK 1)
270512069	3	der (HK 1)
181416813	2	in (HK 2)
151763185	3	und (HK 2)
143609263	3	das (HK 2)
142196885	4	sein (HK 2)
113310246	6	" (HK 2)
86189712	2	zu (HK 3)
81474382	3	ein (HK 3)
74111687	6	werden (HK 3)
72670310	3	von (HK 3)
61688955	1	: (HK 3)
61502737	5	haben (HK 3)
60574217	3	mit (HK 3)
59682179	2	an (HK 3)
54324289	1	) (HK 3)
54233674	1	((HK 3)
52918158	1	- (HK 3)
51338837	4	eine (HK 3)
50210931	3	für (HK 4)

Abb. 70: Recherchierbare Lemma-Übersichtsliste



Text- und Tokenanzahl nach Metadaten

Korpus egal

Medium egal

Register Gebrauch

Domäne Technik

Land egal Region Nordost Ort egal

Jahr >= 2014

Ergebnis: 583 Texte mit 242173 Connexor-Token

Text-ID	Korpus	Titel	Connexor-Token	TreeTagger-Token	Medium	Register	Domäne	Ort	Jahr
T14/FEB.00302	t14	die tageszeitung, 03.02.2014, S. 08; Soja ohne Regenwald	221	0	1	2	5	Berlin	2014
T14/FEB.00442	t14	die tageszeitung, 05.02.2014, S. 08; Rechtliche Zweifel an Offshore-Windparks	419	0	1	2	5	Berlin	2014
T14/FEB.00443	t14	die tageszeitung, 05.02.2014, S. 09; Gelbe Tonne in der Krise	669	0	1	2	5	Berlin	2014
T14/FEB.00644	t14	die tageszeitung, 06.02.2014, S. 08; Regierung nicht gegen Genmais	535	0	1	2	5	Berlin	2014

Abb. 71: Eingrenzung und Zählung von Korpusinhalten auf Basis von Text-Metadaten

Andere Typen von Übersichtslisten werden vorzugsweise dynamisch erstellt. Hierzu zählen in erster Linie die in Abbildung 71 dokumentierte Berechnung von Text- und Worthäufigkeiten für variabel spezifizierbare Ausprägungen textspezifischer Metadaten. Bereits die ausnehmend hohe Anzahl potenzieller Kombinationen von Metadatentypen und -werte – im dargestellten Beispiel für Korpusname, Medium, Register, thematische Domäne, regionale Zuordnung durch drei abgestufte Attribute sowie für die Jahresangabe – macht eine fest kodierte Speicherung der zugehörigen Häufigkeiten unpraktikabel. Stattdessen bedient sich das die Übersicht generierende Skript der in Kapitel 3 eingeführten Indizes für die Texttabelle TB_TEXT sowie für die in der Korpushierarchie darunter liegenden Segmenttabellen.

7. Zusammenfassung und Fazit

Die Nutzung umfangreicher Korpussammlungen ist für Linguisten – insbesondere wenn sie Sprache mit empirischen Mitteln erforschen und beschreiben wollen – zunehmend unabdingbar. Korpora stellen authentisches Datenmaterial zur Verfügung, ohne das ein zuverlässiger Blick auf Sprachrealitäten kaum möglich erscheint. Dies gilt gleichermaßen für zweckoffene, multifunktional angelegte Sammlungen wie für fach- oder themenbezogene Spezialkorpora. Anwendungsgebiete finden sich in der theoriebildenden Wissenschaft, bei der Explikation der Beschaffenheit bestimmter Realitätsausschnitte oder im Zusammenhang mit sprachstatistischen Erhebungen zur Deskription einzelner Sprachphänomene.

Durch die „korpuslinguistische Brille" schaut man dabei selten ausschließlich auf Primärdaten, also auf digitalisierte Rohtexte oder transkribiertes Audiomaterial. Mindestens ebenso wichtig ist die Einbeziehung von Sekundärdaten in Form (morpho-)syntaktischer, semantischer oder phonetischer Annotationen sowie textspezifischer Metadaten. Unser in Kapitel 2 vorgestellter Anforderungskatalog für linguistisch motivierte Recherchen deckt vor diesem Hintergrund bereits ein exemplarisches Spektrum von Kategorien und Merkmalen ab, die für die Erforschung komplexer Fragestellungen zum Einsatz kommen können. Unter Verwendung multidimensionaler Suchkriterien werden sprachliche Phänomene auffindbar gemacht, die sich anschließend mit statistischen Methoden – etwa hinsichtlich des Einflusses von ebenfalls in der Datenbasis kodierten inner- oder außersprachlichen Faktoren – gezielt weiter analysieren lassen.

Der intensivierte Einsatz korpuslinguistischer Methoden in lexikografisch-semantisch, grammatisch oder pragmatisch orientierten Forschungskontexten verläuft eingebettet in weitreichende methodische und technologische Fortschritte auf anderen Gebieten: Computerlinguistische Projekte beschäftigen sich mit der maschinell unterstützten Analyse digitaler Sprachdaten, auch und gerade für nicht-standardnahe Produktionskontexte – etwa in sozialen Medien – sowie für ein breites Spektrum von Einzelsprachen. Aus diesen Bemühungen resultiert ein sukzessive verbessertes Angebot an Werkzeugen für die automatisierte Annotation unterschiedlicher Beschreibungsebenen (Sätze, Wortgruppen/Phrasen, Wörter etc.) natürlichsprachlicher Texte. Aus technischer Sicht profitieren Korpushaltung und -bearbeitung von den in jüngerer Zeit überproportional angestiegenen Datenvolumina, die sich auf handelsüblicher Hardware verwalten lassen. Verfügbarkeit und Kosten einschlägi-

ger Lösungen stellen somit keine nennenswerten Hürden für die Archivierung datenintensiver Sprachressourcen mehr dar.

Zunehmend anspruchsvoller gestaltet sich hingegen die gezielte Korpusrecherche. Dabei steht nicht allein die Ausdrucksmächtigkeit einzelner Abfragesprachen als vielmehr eine alltagstaugliche Performanz – im Sinne akzeptabler Laufzeiten – im Blickpunkt. Angesichts kontinuierlich ansteigender Archivgrößen weltweit stehen die Betreiber moderner Korpusabfragesysteme vor dem Dilemma, dass vormals ausreichende Lösungen, etwa unter Zuhilfenahme von Volltext-Retrievalsystemen, häufig nicht die Anforderungen komplexer Forschungsproblematiken hinsichtlich der Einbeziehung diverser heterogener Sekundärdaten ausreichend bedienen. Erweitert man die Möglichkeiten horizontal bzw. linear orientierten Textretrievals um vertikale Funktionalitäten, so stellt sich unweigerlich die Frage nach problemadäquaten Datenmodellierungen und Indizierungsstrategien sowie deren performanter Integration in existierende Lösungen. In diesem Sinne bieten unsere Untersuchungen Einblicke in ein hochaktuelles Forschungsumfeld und dokumentieren Archivierungsvarianten, Zugriffswege und Abfrageformulierungen für die effektive Korpusrecherche.

Das in Kapitel 3 vorgestellte Referenzsystem einer Korpusdatenbank nimmt sich dieser Problematiken an. Am Beispiel mehrerer paralleler Annotationsebenen wurde aufgezeigt, wo praktische Herausforderungen im Einzelnen liegen und wie eine Lösung unter Zuhilfenahme relationaler Techniken aussehen kann. Insbesondere wurden für eine Datenmenge von zunächst einer Milliarde fortlaufender Wortformen grundlegende Designentscheidungen – beispielsweise n-Gramm-Tabellen vs. streng wortorientierte Relationierung sowie die Eignung einzelner Indizierungsvarianten für korpuslinguistisch interessante Phänomene – evaluiert und als Nebeneffekt die Gültigkeit des Zipf'schen Gesetzes für Wortkombinationen validiert. Davon unberührt bleiben mögliche Verfeinerungen des Datenmodells, etwa zur Normalisierung der – in unseren Abfragen nicht verwendeten – Xerox-Featurelist (CO_FTS) in atomare Wertebereiche. Die Implementierung auf einer Mid-Range-Hardwareplattform liefert zum einen den Nachweis, dass eine relationale Abbildung verschiedener Segmentierungsebenen von zentraler Bedeutung ist, um detaillierte Recherchen überhaupt erst zu ermöglichen. Je vielfältiger die für eine Abfrage relevanten Metadatentypen ausfallen, umso unabdingbarer erscheinen eine feingranulare Datenhaltung und -abfrage. Zum anderen belegen die Messergebnisse empirisch den Umstand, dass datenbankgestützte Korpusretrievalzeiten maßgeblich von den absoluten Frequenzen der abzufragenden Phänomene abhängen.

Unser präferiertes Datenmodell mit tokenorientierter Relationierung benötigt im Vergleich zu n-Gramm-Tabellen deutlich weniger Volumenkapazitäten und bietet gleichzeitig maximale Flexibilität hinsichtlich der kombinierten Abfrage beliebig vieler Suchkriterien mit frei festlegbaren Wortabständen. Trotz der strengen Relationierung erreichen wir für abgestufte Häufigkeitsklassen durchgehend akzeptable Retrievalzeiten. Im Einzelfall – etwa bei der Recherche nach hochfrequenten Phänomenen – bietet sich darüber hinaus ein partielles Aufbrechen der Relationierung unter Rückgriff auf die Zipf'sche Formel oder die 80-20-Regel (Pareto-Prinzip) an, um potenzielle „Flaschenhälse" in ausgelagerten Tabellen isoliert zu handhaben. Insgesamt stellen die detailliert dokumentierten Performanzwerte eine projektübergreifend wertvolle empirische Ressource zur Entscheidung konkreter Designfragen für authentisches Sprachmaterial dar.

Unter weitestgehender Übernahme der evaluierten Modellierung belegen die Testläufe in Kapitel 4, dass unser Datenbankdesign grundsätzlich auch für große Korpora mit mehreren Milliarden fortlaufenden Wortformen geeignet ist. Ausgangspunkt ist die Aufteilung des Datenbestands in Evaluationskorpora mit einem Umfang von einer Million (EK-1) bis acht Milliarden Wortformen (EK-6). Gemessen wurden die zehn Referenzrecherchen des Abfragekatalogs mit differierenden Anzahlen und Typen von Suchkriterien. Die Auswertungen der Retrievalzeiten auf unterschiedlich leistungsstarken Rechnerplattformen – zusätzlich zum Referenzsystem kam zur Überprüfung der vertikalen Skalierung eine zweite Hardwareumgebung mit der vierfachen Menge an CPU-Kernen und einer verdoppelten Hauptspeicherkapazität zum Einsatz – zeichnen ein vorwiegend positives Bild.

Dabei liegt unser Hauptaugenmerk nicht auf den gemessenen absoluten Abfragezeiten. Das Ziel der durchgeführten Untersuchungen besteht nicht im unmittelbaren Vergleich mit existierenden Korpusabfragesystemen oder in der Etablierung eines einschlägigen Benchmark-Tests, nicht zuletzt weil ein solcher aufgrund potenziell mannigfaltiger korpus- und technologiebezogener Rahmenparameter kaum gehaltvolle Aussagewerte liefern würde. Weitaus interessanter erscheinen die grundlegenden Tendenzen: Die Steigerungsfaktoren der Laufzeiten skalieren in allen Fällen unterproportional und liegen signifikant unter den Steigerungsfaktoren für Korpusgröße und Beleganzahl. Daraus lässt sich der Schluss ziehen, dass bereits als „all in one" konstruierte Join-Abfragen auf angemessen relationierten Sprachdaten für den Einsatz in expandierenden Korpusrecherchesystemen eine praktikable Strategie darstellen. Dies betrifft fünf der insgesamt zehn Katalogabfragen, die entweder vergleichsweise wenige Suchkriterien beinhalten oder – sofern reguläre Aus-

drücke abgearbeitet werden sollen – mit vorgeschalteten Filterabfragen operieren. Auch hier bietet der ermittelte Datenfundus eine Quelle für weiterführende Untersuchungen.

Neben diesem grundsätzlich positiven Befund steht die Erkenntnis, dass speziell Recherchen unter Einbeziehung mehrerer inner- und außersprachlicher Selektionskriterien rasch optimierungsbedürftige Laufzeiten generieren. In Anbetracht der jüngeren technologischen Entwicklung, die weniger auf eine Maximierung von CPU-Taktfrequenzen als auf die Erhöhung der Anzahl parallel arbeitender CPU-Kerne setzt, erscheint deshalb für die performante Korpusrecherche aus informatischer Sicht folgende Vorgehensweise als sinnvoll:

– Segmentierung komplexer Korpusabfragen in unabhängige Teilabfragen
– Weitestgehend parallele Durchführung dieser Teilabfragen und abschließende Zusammenführung der Zwischenergebnisse

Der in Kapitel 5 thematisierte Ansatz einer „problemorientierten Algorithmisierung“ versucht, diese Anforderungen auf authentischem Sprachmaterial umzusetzen. Als Inspiration dient das etablierte Map-Reduce-Programmiermodell, das anhand einer originär sprachwissenschaftlichen Aufgabenstellung vorgestellt wird. Allerdings teilen wir in der praktischen Anwendung nicht den Suchraum in separat durchsuchbare Teilbereiche ein, sondern konzentrieren uns auf die Zerlegung von Abfrageformulierungen. Eine maßgebliche Zielsetzung besteht also in der Identifikation derjenigen Suchkriterien, die sich optimal in einer Teilabfrage kombinieren lassen. Darüber hinaus gilt es, diese „Mappings“ so zu gestalten, dass bei der späteren Zusammenführung in „Reduce“-Schritten die Verifizierung sämtlicher Verknüpfungsoperatoren – also beispielsweise der geforderten Abstände zwischen einzelnen Textwörtern – gewährleistet bleibt. Neben der korrekten linearen Anordnung sind auch Suchkriterien einzubeziehen, die sich auf übergeordnete syntaktische Strukturen oder textspezifische Metadaten beziehen. Diese Aufgabe ist keinesfalls trivial und wird für unsere Testläufe von einem Subalgorithmus übernommen, der für eine gegebene Anzahl von Suchkriterien passende Map- und Reducevarianten in Form sogenannter MR-Bäume vorgibt. Die letztliche Bestimmung der „Map“-Kombinationen erfolgt auf Basis von in der Datenbank vorgehaltenen einzelphänomenspezifischen Frequenzangaben.

Zu klären ist in jedem Fall, wie umfangreich die Menge der in einem „Map“-Schritt zusammengefassten Suchkriterien sein soll. Derartige Entscheidungen lassen sich vermutlich nicht allgemeingültig treffen, sondern hängen u.a. von der Leistungsfähigkeit der verwendeten Hardware, der Beschaffenheit des Datenmaterials sowie dem Datenmodell ab. Für die im Referenzsystem im-

plementierten Indexvarianten legen die Testläufe eine Kombination von maximal zwei Suchkriterien pro Teilabfrage nahe; diese Festlegung wurde auch für die in Kapitel 6 vorgestellte Implementierung im Rahmen eines Online-Frameworks übernommen.

Betrachten wir die dokumentierten Auswirkungen der alternativen MR-Algorithmisierung auf komplexe Korpusrecherchen, so erscheint das Ziel einer Verkürzung von Abfragezeiten prinzipiell erreicht. Zwar erfüllt auch unsere alternative Suchstrategie nicht die Hoffnung auf eine proportional lineare Verbesserung der Abfragezeiten bei ansteigender CPU-Zahl. Trotzdem lassen sich positive Auswirkungen auf die meisten Optimierungskandidaten nachweisen – von der problemorientierten Segmentierung profitieren zuvorderst die Katalogabfragen 3, 4 und 6. In diesen Fällen bewirken Isolation und parallele Abarbeitung von „Map"-Pärchen, die vorzugsweise aus je einem hoch- und einem niederfrequenten Suchkriterium bestehen, eine signifikante Beschleunigung der Gesamtlaufzeiten. Abfrage 8, die wortspezifische Suchkriterien mit Referenzen auf übergeordnete Wortgruppen sowie mit Einschränkungen hinsichtlich der Häufigkeitsklassen kombiniert, liefert im Großen und Ganzen unveränderte Werte. Am wenigsten profitiert Abfrage 5 mit nur vier Suchkriterien, die sämtlich auf vergleichsweise hochfrequente Phänomene referieren, von der Umstellung.

Dieser Befund leitet über zu einer pragmatischen Eingrenzung des Einsatzbereichs von MR-Bäumen für die Korpusrecherche: Mit Sicherheit lassen sich Anwendungsfälle konstruieren, in denen die problemorientierte Segmentierung nicht nur nicht weiterhilft, sondern sogar kontraproduktive Auswirkungen auf die Laufzeit mit sich bringt. Dies betrifft naheliegenderweise Abfragen, bei denen durch die Pärchenbildung keine Entlastung erreichbar ist, weil z.B. alle Einzelsuchkriterien umfangreiche Ergebnislisten generieren und „Map"-Schritte zu entsprechend rechenintensiven Zwischenergebnissen führen. Unser Versuch einer Optimierung des Abfragealgorithmus in Kapitel 5 ist folglich als optionale, abfrage- und systemabhängige Alternative zu verstehen. Die in moderne Datenbankmanagementsysteme üblicherweise integrierten Anfrageoptimierer (z.B. Cost-Based-Optimizer) formulieren bereits für „all in one"-Abfragen unter Heranziehung von Datenverteilungs-Statistiken, Speichereigenschaften etc. leistungsfähige Ausführungspläne. Die explizite Vorab-Segmentierung komplexer Abfragen in Teilprobleme bietet sich dann an, wenn durch die damit einher gehende Hinzufügung (linguistischen) Weltwissens eine Reduzierung der Abfragekomplexität, d.h. der zu berechnenden Joins, realisierbar ist. Nicht zuletzt profitieren davon Recherchen, bei denen ein oder mehrere Suchkriterien ergebnislos bleiben: Sobald bereits eine

einzige Teilabfrage ohne Treffer terminiert, kann die Gesamtsuche abgebrochen werden.

Wesentlicher Kern des modifizierten Ansatzes bleibt die Segmentierung linguistisch motivierter Suchkriterien als Gegenentwurf zur physischen Segmentierung des Datenbestands. Gleichwohl stellen Lösungen für eine horizontale Skalierung, also die Verteilung auf unterschiedliche Serverknoten, eine mögliche und gegebenenfalls sogar wünschenswerte Erweiterung zur Erzielung zusätzlicher Laufzeitverkürzungen dar. Die konkrete Implementierung des Algorithmus kann im Einzelnen variabel durchgeführt werden und sollte auch auf andere Rechercheplattformen – also insbesondere auf Systeme ohne eigene kostenbasierte Optimierung – übertragbar sein.

Der Wert unseres problemorientierten Retrievalmodells ist damit nicht auf relationale Datenbanksysteme beschränkt. Ziel ist insgesamt nicht ein Ersatz, sondern eine Erleichterung der Aufgaben etablierter Technologien im Kontext der Korpusrecherche gemäß der in (Stonebraker et al. 2007, S. 1159) formulierten Empfehlung: „The necessity of rethinking both data models and query languages for the specialized engines, which we expect to be dominant in the various vertical markets“.

Literatur

Abel, Andrea/Zanin, Renata (Hg.) (2011): Korpora in Lehre und Forschung. Bozen: Bozen University Press.

Adamzik, Kirsten (Hg.) (2000): Textsorten: Reflexionen und Analysen. Tübingen: Stauffenburg.

Altmann, Vivien/Altmann, Gabriel (2008): Anleitung zu quantitativen Textanalysen. Methoden und Anwendungen. (= Studies in Quantitative Lingzuistics 2). Lüdenscheid: RAM-Verlag.

Amdahl, Gene M. (1967): Validity of the single processor approach to achieving large scale computing capabilities. In: American Federation of Information Processing Societies (Hg.): Proceedings of the AFIPS Spring Joint Computer Conference, Atlantic City, NJ. Washington, DC: Thomson Book Company, S. 483-485.

Andrews, Gregory R. (2000): Foundations of multithreaded, parallel, and distributed programming. Reading, MA u.a.: Addison-Wesley.

Aston, Guy/Burnard, Lou (1998): The BNC handbook. Exploring the British National Corpus with SARA. Edinburgh: Edinburgh University Press.

Baayen, R. Harald (2008): Analyzing linguistic data. A practical introduction to statistics using R. Cambridge: Cambridge University Press.

Banko, Michele/Brill, Eric (2001): Scaling to very very large corpora for natural language disambiguation. In: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics. Stroudsburg, PA: Association for Computational Linguistics, S. 26-33.

Bański, Piotr et al. (2013): KorAP: the new corpus analysis platform at IDS Mannheim. In: Vetulani, Zygmunt/Uszkoreit, Hans (Hg.): Human language technologies as a challenge for computer science and linguistics. 6th Language and Technology Conference. Poznań: Uniwersytet im. Adama Mickiewicza w Poznaniu, S. 586-587.

Bański, Piotr et al. (2014): Access control by query rewriting: the case of KorAP. In: Calzolari (Hg.), S. 3817-3822. www.lrec-conf.org/proceedings/lrec2014/pdf/743_Paper.pdf (Stand: 31.8.2018).

Bański, Piotr et al. (Hg.) (2015): Proceedings of the LREC 2015 workshop „Challenges in the Management of Large Corpora (CMLC-3)“. Mannheim: Institut für Deutsche Sprache.

Bański, Piotr et al. (Hg.) (2018): Proceedings of the LREC 2018 workshop „Challenges in the Management of Large Corpora (CMLC-6)“, Miyazaki. Paris: European Language Resources Association (ELRA).

Bardoel, Thomas (2012): Comparing n-gram frequency distributions. Explorative research on the discriminative power of n-gram frequencies in newswire corpora. Master Thesis. Tilburg. Department of Communication and Information Sciences, Tilburg University. http://ilk.uvt.nl/downloads/pub/papers/hait/bardoel2012.pdf (Stand: 31.8.2018).

Baroni, Marco et al. (2009): The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. In: Language Resources and Evaluation 43, S. 209-226.

Bartz, Thomas/Beißwenger, Michael/Storrer, Angelika (2014): Optimierung des Stuttgart-Tübingen-Tagset für die linguistische Annotation von Korpora zur internetbasierten Kommunikation: Phänomene, Herausforderungen, Erweiterungsvorschläge. In: Journal for Language Technology and Computational Linguistics (JLCL) 28, 1, S. 157-198.

Baumann, Steffen et al. (2004): Multi-dimensional annotation of linguistic corpora for investigating information structure. In: Proceedings Frontiers in Corpus Annotation Workshop at HLT/NAACL. Stroudsberg, PA: The Association for Computer Linguistics, S. 39-46.

Beal, Joan C./Corrigan, Karen P./Moisl, Hermann L. (Hg.) (2007): Creating and digitizing language corpora. Bd. 1: Synchronic Databases. London: Palgrave Macmillan.

Becher, Margit (2009): XML. DTD, XML-Schema, XPath, XQuery, XSLT, XSL-FO, SAX, DOM. Herdecke/Witten: W3L-Verlag.

Beißwenger, Michael (2018): Internetbasierte Kommunikation und Korpuslinguistik. Repräsentation basaler Interaktionsformate in TEI. In: Lobin/Schneider/Witt (Hg.), S. 307-349.

Beißwenger, Michael/Storrer, Angelika (2008): Corpora of computer-mediated communication. In: Lüdeling/Kytö (Hg.), S. 292-308.

Beißwenger, Michael/Storrer, Angelika (2011): Digitale Sprachressourcen in Lehramtsstudiengängen. Kompetenzen – Erfahrungen – Desiderate. In: Journal for Language Technology and Computational Linguistics (JLCL) 26, 1, S. 119-139.

Beißwenger, Michael et al. (Hg.) (2014): Building and annotating corpora of computer-mediated communication. Issues and challenges at the interface of corpus and computational linguistics. Journal of Language Technology and Computational Linguistics 29, 2.

Belica, Cyril (2011): Semantische Nähe als Ähnlichkeit von Kookkurrenzprofilen. In: Abel/Zanin (Hg.), S. 155-178.

Belica, Cyril/Steyer, Kathrin (2008): Korpusanalytische Zugänge zu sprachlichem Usus. In: Vachková, Marie (Hg.): Beiträge zur bilingualen Lexikographie. Prag: Univerzita Karlova, Filozifická Fakulta, S. 7-24.

Belica, Cyril et al. (2011): The morphosyntactic annotation of DeReKo. Interpretation, opportunities, and pitfalls. In: Konopka et al. (Hg.), S. 451-470.

Ben-Ari, Moti/Lutz, Michael (1985): Grundlagen der Parallelprogrammierung. München: Hanser.

Benor, Sarah Bunin/Levy, Roger (2006): The chicken or the egg? A probabilistic analysis of English binomials. In: Language Resources and Evaluation 82, S. 233-278.

Berberich, Klaus/Bedathur, Srikanta (2013): Computing n-gram statistics in MapReduce. In: Paton, Norman (Hg.): Advances in database technology (EDBT 2013). 16th International Conference on Extending Database Technology, Genoa. New York: Association for Computing Machinery, S. 101-112.

Best, Karl-Heinz (2006): Quantitative Linguistik. Eine Annäherung. 3., stark überarb. u. erg. Aufl. (= Göttinger Linguistische Abhandlungen 3). Göttingen: Peust und Gutschmidt.

Biber, Douglas (1993a): Representativeness in corpus design. In: Literary and Linguistic Computing 8, 4, S. 243-257.

Biber, Douglas (1993b): The Multi-Dimensional Approach to Linguistic Analyses of Genre Variation: An Overview of Methodology and Findings. In: Computers and the Humanities 26, S. 331-345.

Biber, Douglas/Conrad, Susan/Reppen, Randi (1998): Corpus linguistics. Investigating language structure and use. Cambridge: Cambridge University Press.

Bickel, Hans et al. (2009): Schweizer Text Korpus. Theoretische Grundlagen, Korpusdesign und Abfragemöglichkeiten. In: Linguistik online 39. https://bop.unibe.ch/linguistik-online/article/view/474 (Stand: 1.2.2019).

Biemann, Christian (2007): A random text model for the generation of statistical language invariants. In: Proceedings of HLT-NAACL-07. Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics. Stroudsburg, PA: Association for Computational Linguistics. www.aclweb.org/anthology/N/N07/N07-1.pdf (Stand: 1.2.2019).

Biemann, Christian et al. (2007): The Leipzig Corpora Collection. Monolingual Corpora of Standard Size. In: Matthew Davies et al. (Hg.): Proceedings of the Corpus Linguistics Conference CL2007, Birmingham. Birmingham: University of Birmingham. http://ucrel.lancs.ac.uk/publications/CL2007/paper/190_Paper.pdf (Stand: 3.9.2018).

Biemann, Christian et al. (2013): Scalable construction of high-quality web corpora. In: Journal for Language Technology and Computational Linguistics (JLCL) 28, 2, S. 23-60.

Bindernagel, Matthias (2007): XML versus RDBMS. Performanz und Implementierung linguistischer Anfragen. Studienarbeit. Humboldt-Universität. Berlin: Institut für Informatik.

Bird, Steven et al. (2005): Extending XPath to support linguistic queries. In: Workshop on Programming Language Technologies for XML (Plan-X). San Francisco: ACM, S. 35-46.

Bird, Steven/Liberman, Mark (1999): A formal framework for linguistic annotation. Hrsg. v. Department of Computer &. Information Science, University of Pennsylvania (Technical Reports (CIS)). https://repository.upenn.edu/cis_reports/110/ (Stand: 1.2.2019).

Bodmer, Franck (2005): Recherchieren in den Korpora des IDS. In: Sprachreport 3, S. 2-5.

Bourret, Ronald (2005): XML and Databases. www.rpbourret.com/xml/XMLAndDatabases.htm (Stand: 3.9.2018).

Bos, Johan et al. (2017): The Groningen Meaning Bank. In: Ide, Nancy/Pustejovsky, James (Hg.): Handbook of linguistic annotation. Bd. 2. Berlin: Springer, S. 463-496.

Brants, Sabine et al. (2002): The TIGER Treebank. In: Proceedings of the Workshop on Treebanks and Linguistic Theories. Sozopol. www.coli.uni-saarland.de/publikationen/softcopies/Brants:2002:TT.pdf (Stand: 3.9.2018).

Brants, Thorsten/Franz, Alex (2006): Web 1T 5-gram corpus version 1.1. https://catalog.ldc.upenn.edu/LDC2006T13 (Stand: 1.2.2019).

Brants, Thorsten/Skut, Wojciech/Uszkoreit, Hans (1999): Syntactic annotation of a German newspaper corpus. In: Proceedings of the ATALA Treebank Workshop, Paris, S. 69-76. www.coli.uni-saarland.de/publikationen/softcopies/Brants:1999:SAG.pdf (Stand: 3.9.2018).

Bresnan, Joan et al. (2007): Predicting the dative alternation. In: Bouma, Gerlof/Kraemer, Irene/Zwarts, Joost (Hg.): Cognitive foundations of interpretation. Amsterdam: Royal Netherlands Academy of Arts and Sciences, S. 69-94.

Brocardo, Marcelo Luiz et al. (2013): Authorship verification for short messages using stylometry. In: IEEE *Xplore* (Hg.): International Conference on Computer, Information and Telecommunication Systems CITS.2013, Athens. Athen: IEEE. DOI: 10.1109/CITS.2013.6705711.

Bryla, Bob/Loney, Kevin (2013): Oracle database 12c. The complete reference. New York: McGraw-Hill (Oracle Press).

Bubenhofer, Noah (2009): Sprachgebrauchsmuster. Korpuslinguistik als Methode der Diskurs- und Kulturanalyse. (= Sprache und Wissen 4). Berlin: De Gruyter.

Bubenhofer, Noah (2011): Korpuslinguistik in der linguistischen Lehre. Erfolg und Misserfolge. In: Journal for Language Technology and Computational Linguistics (JLCL) 26, 1, S. 41-156.

Bubenhofer, Noah/Kupietz, Marc (Hg.) (2018): Visualisierung sprachlicher Daten. Visual linguistics – praxis – tools. Heidelberg: Heidelberg University Publishing.

Bubenhofer, Noah/Scharloth, Joachim (2015): Maschinelle Textanalyse im Zeichen von Big Data und Data-driven Turn – Überblick und Desiderate. In: Zeitschrift für germanistische Linguistik 43, 1, S. 1-26.

Bubenhofer, Noah/Konopka, Marek/Schneider, Roman (2014): Präliminarien einer Korpusgrammatik. (= Korpuslinguistik und interdisziplinäre Perspektiven auf Sprache (CLIP) 4). Tübingen: Narr.

Burchardt, Aljoscha et al. (2009): Using FrameNet for the semantic analysis of German: annotation, representation, and automation. In: Boas, Hans C. (Hg.): Multilingual FrameNets in computational lexicography: methods and applications. Berlin/Boston: De Guyter: 209-244.

Burghardt, Manuel/Wolff, Christian (2009): Stand off-Annotation für Textdokumente. Vom Konzept zur Implementierung (zur Standardisierung?). In: Chiarcos, Christian/de Castilho, Richard Eckart/Stede, Manfred (Hg.): Von der Form zur Bedeutung. Texte automatisch verarbeiten = From form to meaning. Processing texts automatically. Proceedings of the Biennial GSCL Conference 2009. Tübingen: Narr, S. 53-59.

Burleson, Donald K. (2014): Oracle tuning. The definitive reference. 3. Aufl. Kittrell, NC: Rampant TechPress.

Burnard, Lou (2005): Metadata for corpus work. In: Wynne (Hg.), S. 30-46.

Burnard, Lou/Bauman, Syd (Hg.) (2013): TEI P5: guidelines for electronic text encoding and Iiterchange. Version 2.3.0. Originally edited by C.M. Sperberg-McQueen and Lou Burnard for the ACH-ALLC-ACL Text Encoding InitiativeNote: Now entirely revised and expanded under the supervision of the Technical Council of the TEI Consortium. TEI Consortium. Charlottesville, VA. www.tei-c.org/Guidelines/P5/ (Stand: 3.9.2018).

Calzolari, Nicoletta (Hg.) (2010): Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC 2010). Valletta: European Language Resources Association (ELRA).

Calzolari, Nicoletta (Hg.) (2012): Proceedings of the 8th Conference on International Language Resources and Evaluation (LREC 2012), Istanbul, Turkey. Istanbul: European Language Resources Association (ELRA).

Calzolari, Nicoletta (Hg.) (2014): Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC 2014). Reykjavik: European Language Resources Association (ELRA).

Capriolo, Edward/Wampler, Dean/Rutherglen, Jason (2012): Programming hive. Sebastopol, CA: O'Reilly.

Carletta, Jean et al. (2005): The NITE XML toolkit. Data model and query language. In: Language Resources and Evaluation 39, 4, S. 313-334.

Carlson, Lynn (2002): RST discourse treebank. Philadelphia, PA: Linguistic Data Consortium.

Carstensen, Kai-Uwe et al. (2010): Computerlinguistik und Sprachtechnologie. Eine Einführung. 3., überarb. u. erw. Aufl. Berlin/Heidelberg: Spektrum Akademischer Verlag.

Chen, Peter Pin-Shan (1976): The Entity-Relationship Model. Toward a unified view of data. In: ACM Transactions on Database Systems 1, S. 9-36.

Chen, Peter Pin-Shan (2002): Entity-Relationship Modeling. Historical events, future trends, and lessons learned. In: Broy, Manfred/Denert, Ernst (Hg.): Software pioneers. Contributions to software engineering. Berlin/New York: Springer, S. 296-310.

Cheng, Winnie (2012): Exploring corpus linguistics. Language in action. London/New York: Routledge.

Chiarcos, Christian/Ritz, Julia/Stede, Manfred (2009): 'By all these lovely tokens…': merging conflicting tokenizations. In: Stede, Manfred et al (Hg.): Proceedings of the Third Linguistic Annotation Workshop (LAW-III) at ACL-IJCNLP 2009. Singapur: Association for Computational Linguistics, S. 35-43. www.atala.org/revuetal (Stand: 1.2.2019).

Chiarcos, Christian et al. (2008): A flexible framework for integrating annotations from different tools and tag sets. In: Traitement Automatique des Langues 49, 2, S. 217-246.

Christ, Oliver (1994): A modular and flexible architecture for an integrated corpus query system. In: Proceedings of COMPLEX 1994. Budapest, S. 23-32.

Chubak, Pirooz/Rafiei, Davood (2012): Efficient indexing and querying over syntactically annotated trees. In: Proceedings of the VLDB Endowment 5, 11, S. 1316-1327.

Church, Kenneth W./Gale, William A. (1995): Poisson Mixtures. In: Natural Language Engineering 1, 2, S. 163-190.

Church, Kenneth W./Mercer, Robert L. (1993): Introduction to the special issue on computational linguistics using large corpora. In: Computational Linguistics 19, 1, S. 1-24.

CLARIN-D AP 5 (2012): CLARIN-D User Guide. http://media.dwds.de/clarin/userguide/userguide-1.0.1.pdf (Stand: 1.2.2019).

Cunningham, Hamish (2000): Software architecture for language engineering. Sheffield: University of Sheffield. http://citeseerx.ist.psu.edu/viewdoc/download?-doi=10.1.1.12.325&rep=rep1&type=pdf (Stand: 1.2.2019).

Cunningham, Hamish/Bontcheva, Kalina (2006): Computational language systems, architectures. In: Brown, E. Keith/Anderson, Anne (Hg.): Encyclopedia of language & linguistics. 2. Aufl. Amsterdam: Elsevier.

Das, Debopam/Stede, Manfred (2018): Developing the Bangla RST Discourse Treebank. In: Calzolari, Nicoletta et al. (Hg.): Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA). www.lrec-conf.org/proceedings/lrec2018/index.html (Stand: 1.2.2019).

Date, Christopher J. (2004): An introduction to database systems. 8. Aufl. Boston: Pearson/Addison Wesley.

Date, Christopher J./Darwen, Hugh (2007): Databases, types and the relational model. The third manifesto. 2. Aufl. Reading, MA u.a.: Addison-Wesley.

Davies, Mark (2005): The advantage of using relational databases for large corpora. In: International Journal of Corpus Linguistics 10, 3, S. 307-334.

Dean, Jeffrey/Ghemawat, Sanjay (2004): MapReduce: simplified data processing on large clusters. In: Usenix Association (Hg.): Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI04). San Francisco, S. 137-149. www.usenix.org/legacy/event/osdi04/tech/full_papers/dean/dean.pdf (Stand: 1.2.2019).

Dijcks, Jean-Pierre (2009): Oracle white paper. In-Database MapReduce. Redwood Shores. www.oracle.com/technetwork/database/bi-datawarehousing/twp-indatabase-mapreduce-128831.pdf (Stand: 3.9.2018).

Dipper, Stefanie et al. (2004): ANNIS. A linguistic database for exploring information structure. In: Interdisciplinary Studies on Information Structure. (= ISIS Working Papers of the SFB 632 1). Potsdam: Universitätsverlag Potsdam, S. 245-279.

Dipper, Stefanie et al. (2007): Representing and querying standoff XML. In: Rehm, Georg/Witt, Andreas/Lemnitzer, Lothar (Hg.): Datenstrukturen für linguistische Ressourcen und ihre Anwendungen. Data structures for linguistic resources and applications. Tübingen: Narr, S. 337-346.

Duffner, Rolf/Näf, Anton (2006): Digitale Textdatenbanken im Vergleich. In: Linguistik online 28. https://bop.unibe.ch/linguistik-online/article/view/608/1044 (Stand: 1.2.2019).

Durusau, Patrick/O'Donnell, Matthew Brook (2002): Concurrent markup for XML documents. In: Proceedings XML Europe. Barcelona: IDEAlliance.

Dürscheid, Christa/Elspaß, Stephan/Ziegler, Arne (2011): Grammatische Variabilität im Gebrauchsstandard – das Projekt „Variantengrammatik des Standarddeutschen". In: Konopka et al. (Hg.), S. 123-140.

Efer, Thomas (2015): Text mining with graph databases. Traversal of persisted token-level representations for flexible on-demand processing. In: Unger, Herwig/Halang, Wolfgang (Hg.): Autonomous systems. Proceedings of the 8th GI Conference. Düsseldorf: VDI Verlag, S. 157-167.

Ekoniak, Nathaniel (2006): A query-oriented approach to corpus analysis with SQL and modern relational database management systems. Athens, OH: Ohio University, Honors Tutorial College, Department Honors Paper.

Elmasri, Ramez/Navathe, Sham (2009): Grundlagen von Datenbanksystemen. 3., auktual. Aufl. München: Pearson Studium.

Engelberg, Stefan/Lemnitzer, Lothar (2009): Lexikographie und Wörterbuchbenutzung. 4., überarb. u. erw. Aufl. (=Stauffenburg-Einführungen 14). Tübingen: Stauffenburg.

Evert, Stefan (2006): How random is a corpus? The library metaphor. In: Zeitschrift für Anglistik und Amerikanistik 54, S. 177-190.

Evert, Stefan (2010): GoogleWeb 1T 5-Grams made easy (but not for the computer). In: Proceedings of the NAACL HLT 6th Web as Corpus Workshop. Los Angeles: Association for Computational Linguistics, S. 32-40.

Evert, Stefan/Fitschen, Arne (2010): Textkorpora. In: Carstensen et al. (Hg.), S. 369-376.

Evert, Stefan/Hardie, Andrew (2011): Twenty-first century corpus workbench. Updating a query architecture for the new millennium. In: Proceedings of the Corpus Linguistics 2011 Conference, Birmingham. Birmingham: University of Birmingham. http://purl.org/stefan.evert/PUB/EvertHardie2011.pdf (Stand: 6.9.2018).

Evert, Stefan/Hardie, Andrew (2015): Ziggurat. A new data model and indexing format for large annotated text corpora. In: Bański et al. (Hg.), S, 21-27.

Farrar, Scott (2006): A universal data model for linguistic annotation tools. In: Proceedings of 2006 E-MELD Workshop. East Lansing, MI. http://emeld.org/workshop/2006/papers/farrar.html (Stand: 6.9.2018).

Farzindar, Atefeh/Inkpen, Diana (2015): Natural language processing for social media. In: Synthesis Lectures on Human Language Technologies 8, 2, S. 1-166.

Felder, Ekkehard/Gardt, Andreas (2014): Handbuch Sprache und Wissen. Berlin: De Gruyter.

Fialho, Olivia/Zyngier, Sonia (2014): Quantitative methodological approaches to stylistics. In: Burke, Michael (Hg.): The Routledge handbook of stylistics. London u.a.: Routledge, S. 329-345.

Fillmore, Charles J. (1992): „Corpus linguistics" or „Computer-aided armchair linguistics". In: Svartvik, Jan (Hg.): Directions in corpus linguistics. Proceedings of Nobel Symposium 82. Stockholm: De Gruyter, S. 35-60.

Freiknecht, Jonas (2014): Big Data in der Praxis. Lösungen mit Hadoop, HBase und Hive. Daten speichern, aufbereiten, visualisieren. München: Hanser.

Friedl, Jeffrey (2006): Mastering regular expressions. 3. Aufl. Beijing u.a.: O'Reilly.

Fries, Tobias/Kalkreuth, Roman/Hein, Markus (2014): Künstlich Neuronale Netze als Lösungsansatz zur Ermittlung komplexer Korrelationen in der Produktion zum

Einsatz in Cyber Physical Systems. In: Hoffmann, Frank/Hüllermeier, Eyke (Hg.): Proceedings 24. Workshop Computational Intelligence. (= Schriftenreihe des Instituts für Angewandte Informatik/Automatisierungstechnik am Karlsruher Institut für Technologie 50). Karlsruhe: KIT Scientific Publishing, S. 349-361.

Fuß, Eric et al. (Hg.) (2018): Grammar and Corpora 2016. Heidelberg: Heidelberg University Publishing.

Gates, Alan (2011): Programming Pig. Sebastopol: O'Reilly and Associates.

Gatto, Maristella (2014): Web as corpus. Theory and practice. London/New York: Bloomsbury Academic.

Geiselberger, Heinrich/Moorstedt, Tobias (Hg.) (2013): Big Data. Das neue Versprechen der Allwissenheit. Berlin: Suhrkamp (Edition Unseld).

George, Lars (2015): Hbase. The definitive guide. 2. Aufl. Sebastopol, CA: O'Reilly.

Geyken, Alexander (2004): Korpora als Korrektiv für einsprachige Wörterbücher. In: Zeitschrift für Literaturwissenschaft und Linguistik 136, S. 72-100.

Geyken, Alexander (2007): The DWDS corpus. A reference corpus for the German language of the 20th century. In: Christiane Fellbaum (Hg.): Idioms and collocations. Corpus-based linguistic and lexicograhic studies. London: Continuum Press, S. 23-40.

Geyken, Alexander (2011): Statistische Wortprofile zur schnellen Analyse der Syntagmatik in Textkorpora. In: Abel/Zanin (Hg.), S. 115-137.

Geyken, Alexander/Hanneforth, Thomas (2005): TAGH. A complete morphology for German based on weighted finite state automata. In: Proceedings of FSMNLP 2005. Heidelberg: Springer, S. 55-66.

Geyken, Alexander/Klein, Wolfgang (2010): Das Digitale Wörterbuch der Deutschen Sprache (DWDS). In: Heid, Ulrich et al. (Hg.): Lexikographica. Berlin/New York: Akademie der Wissenschaften, S. 79-93.

Ghodke, Sumukh/Bird, Steven (2008): Querying linguistic annotations. In: MacArthur, R. et al. (Hg.): Proceedings of the 13th Australasian Document Computing Symposium. Hobart, S. 69-72. www.researchgate.net/publication/208033288_Querying_Linguistic_Annotations (Stand: 1.2.2019).

Ghodke, Sumukh/Bird, Steven (2010): Fast query for large treebanks. In: Human Language Technologies. The 2010 Annual Conference of the North American Chapter of the ACL. Los Angeles: Association for Computational Linguistics, S. 267-275.

Giles, Dominic (2005): Oracle bitmap indexes and their use in pattern matching. http://dominicgiles.com/PatternMatchingAndBitmappIndexes.pdf (Stand: 6.9.2018).

Gippert, Jost/Gehrke, Ralf (Hg.) (2015): Historical corpora. Challenges and perspectives. (= Korpuslinguistik und interdisziplinäre Perspektiven auf Sprache (CLIP) 5). Tübingen: Narr.

Gleim, Rüdiger/Mehler, Alexander/Eikmeyer, Hans-Jürgen (2007): Representing and maintaining large corpora. In: Davies, Matthew et al. (Hg.): Proceedings of the Corpus Linguistics Conference CL2007, Birmingham. Birmingham: University of Birmingham.

Godfrey, John F./Zampolli, Antonio (1997): Language resources: overview. In: Cole, Ron (Hg.): Survey of the state of the art in human language technology. New York: Cambridge University Press, S. 381-384.

Goldhahn, Dirk/Eckart, Thomas/Quasthoff, Uwe (2012): Building large monolingual dictionaries at the Leipzig Corpora Collection. From 100 to 200 languages. In: Calzolari (Hg.), S. 759-765.

Good, Jeff/Hendryx-Parker, Calvin (2006): Modeling contested categorization in linguistic databases. In: Proceedings of EMELD '06 workshop on digital language documentation. Tool and standards. The state of the art. Lansing, MI: Emeld. http://emeld.org/workshop/2006/papers/GoodHendryxParker-Modelling.pdf (Stand: 13.9.2018).

Graff, David/Cieri, Christopher (2003): English gigaword. Philadelphia: Linguistic Data Consortium. www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2003T05 (Stand: 13.9.2018).

Gray, Jim (2008): Distributed computing economics. In: ACM Queue – Object-Relational Mapping 6,3, S. 63-68.

Gries, Stefan Th. (2003): Towards a corpus-based identification of prototypical instances of constructions. In: Annual Review of Cognitive Linguistics 1, S. 1-27.

Gries, Stefan Th. (2008a): Dispersions and adjusted frequencies in corpora. In: International Journal of Corpus Linguistics 13, 4, S. 403-437.

Gries, Stefan Th. (2008b): Statistik für Sprachwissenschaftler. Göttingen: Vandenhoeck und Ruprecht.

Gries, Stefan Th. (2010a): Corpus linguistics and theoretical linguistics: a love-hate relationship? Not necessarily… In: International Journal of Corpus Linguistics 15, 3, S. 327-343.

Gries, Stefan Th. (2010b): Dispersions and adjusted frequencies in corpora: further explorations. In: Gries, Stefan Th./Wulff, Stefanie/Davies, Mark (Hg.): Corpus linguistic applications: current studies, new directions. Amsterdam: Rodopi, S. 197-212.

Gries, Stefan Th. (2016): Quantitative corpus linguistics with R. London/New York: Routledge.

Gurevych, Iryna/Zesch, Torsten (Hg.) (2013): Language resources and evaluation journal. Special issue on collaboratively constructed language resources. Dordrecht: Springer.

Gustafson, John L. (1988): Reevaluating Amdahl's Law. In: Communications of the ACM 31, 5, S. 532-533.

Gut, Ulrike; Milde et al. (2004): Querying annotated speech corpora. In: Bel, Bernhard (Hg.): Speech Prosody 2004. Tokio: SProSIG.

Hansen, Sandra/Wolfer, Sascha (2017): Standardisierte statistische Auswertung von Korpusdaten im Projekt „Korpusgrammatik" (KoGra-R). In: Konopka, Marek/Wöllstein, Angelika (Hg.): Grammatische Variation. Empirische Zugänge und theoretische Modellierung. (= Jahrbuch des Instituts für Deutsche Sprache 2016). Berlin: De Gruyter, S. 345-356.

Hansen-Morath, Sandra et al. (2018): KoGra-R. Standardisierte statistische Auswertungen von Korpusrecherchen. In: Fuß, Eric/Konopka, Marek/Wöllstein, Angelika (Hg.): Grammatik im Korpus. (= Studien zur Deutschen Sprache 80). Tübingen: Narr, S. 305-363.

Hardt, Manfred/Theis, Fabian J. (2004): Suchmaschinen entwickeln mit Apache Lucene. Frankfurt a.M.: Software & Support.

Henrich, Andreas/Gradl, Tobias (2013): DARIAH(-DE). Digital research infrastructure for the arts and humanities – concepts and perspectives. In: International Journal of Humanities and Arts Computing 7, S. 47-58.

Herring, Susan C. (2010): Computer-mediated conversation. Bd. 1: Introduction and overview. In: Language@Internet 7. www.languageatinternet.org/articles/2010/2801 (Stand: 1.2.2019).

Herring, Susan C. (2011): Computer-mediated conversation. Bd. 2: Introduction and overview. In: Language@Internet 8. www.languageatinternet.org/articles/2011/Herring (Stand: 1.2.2019).

Heyer, Gerhard/Eckart, Thomas/Goldhahn, Dirk (2015): Was sind IT-basierte Forschungsinfrastrukturen für die Geistes- und Sozialwissenschaften und wie können sie genutzt werden? In: Information – Wissenschaft und Praxis 66, S. 295-303.

Heyer, Gerhard/Quasthoff, Uwe/Wittig, Thomas (2008): Text mining. Wissensrohstoff Text. Konzepte, Algorithmen, Ergebnisse. Herdecke: W3L-Verlag.

Hiemstra, Djoerd/Hauff, Claudia (2010): MapReduce for experimental search. In: Proceedings of the Nineteenth Text REtrieval Conference 2010, Gaithersburg. Gaithersburg: NIST. http://trec.nist.gov/pubs/trec19/papers/univ.twente.web.rev.pdf (Stand: 13.9.2018).

Humboldt, Wilhelm von (1988): Wilhelm von Humboldts gesammelte Werke. Bd. 6. Berlin: De Gruyter.

Hunston, Susan (2008): Collection strategies and design decisions. In: Lüdeling/Kytö (Hg.), S. 154-168.

Ide, Nancy/Suderman, Keith (2007): GrAF. A graph-based format for linguistic annotations. In: LAW '07 Proceedings of the Linguistic Annotation Workshop. Prag: ACM, S. 1-8.

Ide, Nancy/Pustejovsky, James (Hg.) (2017): Handbook of linguistic annotation. Berlin/New York: Springer.

Ide, Nancy/Bonhomme, Patrice/Romary, Laurent (2000): XCES. An XML-based encoding standard for linguistic corpora. In: Proceedings of the Second international language resources and evaluation conference. Paris: European Language Resources Association, S. 831-835.

Ihaka, Ross/Gentleman, Robert (1996): R. A language for data analysis and graphics. In: Journal of Computational and Graphical Statistics 5, 3, S. 299-314.

Jakubíček, Miloš et al. (2010): Fast syntactic searching in very large corpora for many languages. In: Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation (PACLIC 2010). Tokio, S. 741-747.

Janus, Daniel/Przepiórkowski, Adam (2007): Poliqarp. An open source corpus indexer and search engine with syntactic extensions. In: Proceedings of the 45th Annual Meeting of the ACL. Prag: Association for Computational Linguistics, S. 85-88.

Jouis, Christophe (2012): Next generation search engines. Advanced models for information retrieval. Hershey, PA: Information Science Reference.

Kahn, Robert/Wilensky, Robert (2006): A framework for distributed digital object services. In: International Journal on Digital Libraries 6, 2, S. 115-123.

Kallmeyer, Werner/Zifonun, Gisela (Hg.) (2007): Sprachkorpora. Datenmengen und Erkenntnisfortschritt. (= Jahrbuch des Instituts für Deutsche Sprache 2006). New York/Berlin: De Gruyter.

Keibel, Holger/Belica, Cyril (2007): CCDB. A corpus-linguistic research and development workbench. In: Proceedings of the 4th Corpus Linguistics Conference. Birmingham. http://corpora.ids-mannheim.de/cl2007-134.pdf (Stand: 1.2.2019).

Keibel, Holger/Kupietz, Marc/Belica, Cyril (2008): Approaching grammar. Inferring operational constituents of language use from large corpora. In: Štícha, František/Fried, Mirjam (Hg.): Grammar & Corpora 2007. Selected contributions from the conference Grammar and Corpora, Sept. 25-27, 2007, Liblice. Prag: Academia, S. 235-242.

Keibel, Holger et al. (2011): Approaching grammar. Detecting, conceptualizing and generalizing paradigmatic variation. In: Konopka et al. (Hg.), S. 329-355.

Kemper, Alfons/Keibel, André (2015): Datenbanksysteme. Eine Einführung. 10., aktual. u. erw. Aufl. Berlin/Boston: Oldenbourg.

Kennedy, Graeme (2008): An introduction to corpus linguistics. 7. Aufl. London: Longman.

Kepser, Stephan (2003): Finite structure query. A tool for querying syntactically annotated corpora. In: Copestake, Ann/Haji, Jan (Hg.): Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (ACL). Budapest: ACL, S. 179-186.

Kepser, Stephan/Mönich, Uwe/Morawietz, Frank (2010): Regular query techniques for XML-Documents. In: Witt, Andreas (Hg.): Linguistic modeling of information and markup languages. Contributions to language technology. Dordrecht: Springer, S. 249-266.

Ketzan, Erik/Schuster, Ingmar (2012): CLARIN-D user guide. Chapter 4: Access to resources and tools – technical and legal issues. Utrecht: CLARIN. http://media.dwds.de/clarin/userguide/userguide-1.0.1.pdf (Stand: 1.2.2019).

Kilgarriff, Adam (2001): Comparing corpora. In: International Journal of Corpus Linguistics 6, 1, S. 97-133.

Kilgarriff, Adam (2007): Googleology is bad science. In: Computational Linguistics 33, 1, S. 147-151.

Kilgarriff, Adam/Grefenstette, Gregory (2003): Introduction to the special issue on web as corpus. In: Computational Linguistics 29, 3, S. 333-347.

Kilgarriff, Adam et al. (2014): The sketch engine. Ten years on. In: Lexicography 1, 1, S. 7-36.

Klein, Dominik/Tran-Gia, Phuoc/Hartmann, Matthias (2013): Aktuelles Schlagwort: Big Data. In: Informatik-Spektrum 36, 3, S. 319-323.

Klosa, Annette (2007): Korpusgestützte Lexikographie: besser, schneller, umfangreicher? In: Kallmeyer/Zifonun (Hg.), S. 105-122.

Klosa, Annette/Kupietz, Marc/Lüngen, Harald (2012): Zum Nutzen von Korpusauszeichnungen für die Lexikographie. In: Lexicographica 28, S. 71-97.

Koehn, Philipp (2005): Europarl: a parallel corpus for statistical machine translation. Europarl MT Summit. Phuket: Europarl. http://homepages.inf.ed.ac.uk/pkoehn/publications/europarl-mtsummit05.pdf (Stand: 17.9.2018).

Köhler, Reinhard (1986): Zur linguistischen Synergetik. Struktur und Dynamik der Lexik. (= Quantitative Linguistics 31). Bochum: Studienverlag Brockmeyer.

Köhler, Reinhard (2005): Korpuslinguistik. Zu wissenschaftstheoretischen Grundlagen und methodologischen Perspektiven. In: LDV-Forum Themenschwerpunkt Korpuslinguistik 20, 2, S. 1-16.

Köhler, Reinhard/Altmann, Gabriel/Piotrovskii, Rajmund G. (Hg.) (2005): Quantitative Linguistik. Ein internationales Handbuch / Quantitative Linguistics. An international Handbook. (= Handbücher zur Sprach- und Kommunikationswissenschaft 27). Berlin/New York: De Gruyter.

Konecny, Christine (2010): Lexikalische Kollokationen und der Beitrag der Internet-Suchmaschine Google zu ihrer Erschließung und Beschreibung. In: Ptashnyk, Stefaniya/Hallsteinsdóttir, Erla/Bubenhofer Noah (Hg.): Korpora, Web und Datenbanken. Computergestützte Methoden in der modernen Phraseologie und Lexikographie. (= Phraseologie und Parömiologie 25). Baltmannsweiler: Schneider-Verlag, S. 77-94.

Konopka, Marek/Fuß, Eric (2016): Genitiv im Korpus. Pilotuntersuchungen zur starken Flexion des Nomens im Deutschen. (= Studien zur Deutschen Sprache 70). Tübingen: Narr.

Konopka, Marek et al. (Hg.) (2011): Grammatik und Korpora 2009. Dritte Internationale Konferenz/Grammar and Corpora 2009. Third International Conference. Mannheim, 22.-24.09.2009. (= Korpuslinguistik und interdisziplinäre Perspektiven auf Sprache (CLIP) 1). Tübingen: Narr.

Krause, Thomas/Zeldes, Amir (2014): ANNIS3: A new architecture for generic corpus query and visualization. In: Digital Scholarship in the Humanities 31, 1, S. 118-139.

Kübler, Sandra (2010): Baumbanken. In: Carstensen et al.(Hg.), S. 492-503.

Kucera, Henry/Francis, Winthrop Nelson (1964): Brown corpus manual. Manual of information to accompany a standard corpus of present-day edited American English, for use with digital computers. Providence, RI: Department of Linguistics, Brown University. http://icame.uib.no/brown/bcm.html (Stand: 17.9.2018).

Kunze, Claudia/Lemnitzer, Lothar (2007): Computerlexikographie. Eine Einführung. Tübingen: Narr.

Künneth, Thomas (2001): Datenbankgestützte Speicherung von Korpora. (= CLUE-Arbeitsberichte/CLUE Technical Reports 4). Erlangen/Nürnberg: Friedrich-Alexander-Universität.

Kupietz, Marc/Keibel, Holger (2009): The Mannheim German Reference Corpus (DeReKo) as a basis for empirical linguistic research. In: Minegishi, Makoto/Kawaguchi, Yuji (Hg.): Working papers in corpus-based linguistics and language education. Bd. 3. Tokio: Tokyo University of Foreign Studies, S. 53-59.

Kupietz, Marc/Lüngen, Harald (2014): Recent developments in DeReKo. In: Calzolari (Hg.), S. 2378-2385.

Kupietz, Marc/Schmidt, Thomas (2015): Schriftliche und mündliche Korpora am IDS als Grundlage für die empirische Forschung. In: Eichinger, Ludwig M. (Hg.): Sprachwissenschaft im Fokus. Positionsbestimmungen und Perspektiven. (= Jahrbuch des Instituts für Deutsche Sprache 2014). Berlin/Boston: De Gruyter, S. 297-322.

Kupietz, Marc/Schmidt, Thomas (Hg.) (2018): Korpuslinguistik. (= Germanistische Sprachwissenschaft um 2020 5). Berlin/New York: De Gruyter.

Kupietz, Marc/Diewald, Nils/Fankhauser, Peter (2018): How to get the computation near the data. Improving data accessibility to, and reusability of analysis functions in corpus query platforms. In: Bański et al. (Hg.), S. 20-25.

Kupietz, Marc et al. (2010): The German Reference Corpus DeReKo. A primordial sample for linguistic research. In: Calzolari (Hg.), S. 1848-1854.

Kupietz, Marc et al. (2014): Maximizing the potential of very large corpora. 50 years of big language data at IDS Mannheim. In: Kupietz et al. (Hg.), S. 1-6.

Kupietz, Marc et al. (Hg.) (2014): Proceedings of the 9th Conference on International Language Resources and Evaluation (LREC 2014) workshop „Challenges in the Management of Large Corpora (CMLC-2)". Reykjavik: European Language Resources Association (ELRA).

Kupietz, Marc et al. (2017): Möglichkeiten der Erforschung grammatischer Variation mithilfe von KorAP. In: Konopka, Marek/Wöllstein, Angelika (Hg.): Grammatische Variation. Empirische Zugänge und theoretische Modellierung. Jahrbuch des Instituts für Deutsche Sprache 2016. Berlin: De Gruyter, S. 319-329.

Kuras, Christoph et al. (2018): Automation, management and improvement of text corpus production. In: Bański et al. (Hg.), S. 1-5.

Kyte, Thomas (2010): Expert Oracle database architecture. Oracle database 9i, 10g, and 11g programming techniques and solutions. 2. Aufl. New York: Apress.

Laffal, Julius (1997): Union and separation in Edgar Allan Poe. In: Literary and Linguistic Computing 12, S. 1-13.

Lai, Catherine/Bird, Steven (2005): LPath+. A first-order complete language for linguistic tree query. In: Proceedings of the 19th Pacific Asia conference on language, information and computation. Taipei: Academia Sinica, S. 1-12.

Laney, Doug (2001): 3D data management. Controlling data volume, velocity, and variety. In: Applications Delivery Strategies 949. Stamford: Meta Group. https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf (Stand: 1.2.2019).

Le Maitre, Jacques/Muriasco, Elisabeth/Rolbert, Monique (1998): From annotated corpora to databases. The SgmlQL language. In: Nerbonne, John (Hg.): Linguistic databases. Stanford: CSLI Publications, S. 37-58.

Leech, Geoffrey (2005): Adding linguistic annotation. In: Wynne (Hg.), S. 17-29.

Leech, Geoffrey (2006): New resources, or just better old ones? The Holy Grail of representativeness. In: Language and Computers 59, S. 133-149.

Lemnitzer, Lothar/Zinsmeister, Heike (2015): Korpuslinguistik. Eine Einführung. 3. Aufl. Tübingen: Narr.

Lemnitzer, Lothar/Romary, Laurent/Witt, Andreas (2013): Representing human and machine dictionaries in markup languages (SGML, XML). In: Gouws, Rufus H. et al.

(Hg.): Dictionaries. An international encyclopedia of lexicography. Supplementary volume. Recent developments with special focus on computational lexicography. Berlin u.a.: De Gruyter, S. 1195-1209.

Lezius, Wolfgang (2002): Ein Suchwerkzeug für syntaktisch annotierte Textkorpora. Diss. Stuttgart: Institut für Maschinelle Sprachverarbeitung.

Lin, Jimmy/Dyer, Chris (2010): Data-intensive text processing with MapReduce. San Rafael, CA: Morgan and Claypool Publishers.

Lin, Jimmy et al. (2009): Of ivory and smurfs. Loxodontan MapReduce experiments for web search. In: Proceedings of the eighteenth text REtrieval conference, Gaithersburg, MA. Gaithersburg: NIST. http://lintool.github.io/NSF-projects/IIS-0916043/publications/Lin_etal_TREC2009.pdf (Stand: 20.9.2018).

Lobin, Henning (2000): Informationsmodellierung in XML und SGML. Heidelberg/Berlin: Springer.

Lobin, Henning (2009): Computerlinguistik und Texttechnologie. München: Wilhelm Fink.

Lobin, Henning/Lemnitzer, Lothar (2004): Texttechnologie. Perspektiven und Anwendungen. Tübingen: Stauffenburg.

Lobin, Henning/Schneider, Roman/Witt, Andreas (Hg.) (2018): Digitale Infrastrukturen für die germanistische Forschung. (= Germanistische Sprachwissenschaft um 2020 6). Berlin/New York: De Gruyter.

Lüdeling, Anke/Kytö, Merja (Hg.) (2008): Corpus linguistics. An international handbook. 2 Bde. (= Handbücher zur Sprach- und Kommunikationswissenschaft 29). Berlin: De Gruyter.

Lüdeling, Anke/Evert, Stefan/Baroni, Marco (2007): Using web data for linguistic purposes. In: Hundt, Marianne/Biewer, Caroline/Nesselhauf, Nadja (Hg.): Corpus linguistics and the web. Amsterdam: Rodopi, S. 7-24.

Lüngen, Harald/Sperberg-McQueen, Cristopher M. (2012): A TEI P5 Document Grammar for the IDS Text Model. In: Journal of the Text Encoding Initiative 3, S. 1-18.

Lüngen, Harald/Witt, Andreas (2008): Multi-dimensional markup. N-way relations as a generalisation over possible relations between annotation layers. Proceedings of Digital Humanities. Oulu: University of Oulu.

Mann, William C./Thompson, Sandra A. (1988): Rhetorical structure theory. Toward a functional theory of text organization. In: Text 3, S. 243-281.

Manning, Christopher D./Schütze, Hinrich (1999): Foundations of statistical natural language processing. Cambridge, MA: MIT Press.

Marcus, Mitchell P./Santorini, Beatrice/Marcinkiewicz, Mary Ann (1993): Building a large annotated corpus of Englisht. The Penn Treebank. In: Computational Linguistics 19, 2, S. 313-330.

Martens, Scott (2013): TüNDRA. A web application for treebank search and visualization. In: Kübler, Sandra/Osenova, Petya/Volk, Martin (Hg.): Proceedings of the Twelfth Workshop on Treebanks and Linguistic Theories. Sofia: Bulgarian Academy of Sciences, S. 133-14.

McCreadie, Richard/Macdonald, Craig/Ounis, Iadh (2012): MapReduce indexing strategies. Studying scalability and efficiency. In: Information Processing and Management 48, 5, S. 873-888.

McEnery, Tony (2003): Corpus linguistics. In: Mitkov, Ruslan (Hg.): The Oxford handbook of computational linguistics. Oxford/New York: Oxford University Press, S. 448-463.

McEnery, Tony/Hardie, Andrew (2013): The history of corpus linguistics. In: Allan, Keith (Hg.): The Oxford handbook of the history of linguistics. Corby: Oxford University Press.

McEnery, Tony/Wilson, Andrew (2001): Corpus linguistics. 2. Aufl. Edinburgh: Edinburgh University Press. www.lancs.ac.uk/fss/courses/ling/corpus/ (Stand: 24.9.2018).

McEnery, Tony/Xiao, Richard/Tono, Yukio (2010): Corpus-based language studies. An advanced resource book. Reprint. London/New York: Routledge.

Meindl, Claudia (2011): Methodik für Linguisten. Eine Einführung in Statistik und Versuchsplanung. Tübingen: Narr Francke Attempto.

Mikheev, Andrei (2002): Periods, capitalized words, etc. In: Computational Linguistics 23, 3, S. 289-318.

Miner, Donald/Shook, Adam (2013): MapReduce design patterns. Sebastopol, CA: O'Reilly.

Mohanty, Hrushikesha/Bhuyan, Prachet/Chenthati, Deepak (2015): Big data. A primer. (= Studies in Big Data 11). Neu-Delhi: Springer.

Moon, Rosamund (Hg.) (2009): Words, grammar, text: revisiting the work of John Sinclair. Amsterdam: Benjamins.

Moore, Gordon Earle (1965): Cramming more components onto integrated circuits. In: Electronics 38, 8, S. 114-117.

Mueller, Martin (2010): Towards a digital carrel. A report about corpus query tools. Mellon Foundation. http://panini.northwestern.edu/mmueller/corpusquerytools.pdf (Stand: 24.9.2018).

Müller, Christoph (2005): A flexible stand-off data model with query language for multilevel annotation. In: Proceedings of the ACL 2005. Stroudsburg, PA: Association for Computational Linguistics, S. 109-112.

Müller, Stefan (2002): Syntax or morphology. German particle verbs revisited. In: Dehé, Nicole et al. (Hg.): Verb-Particle Explorations. (= Interface Explorations 1). Berlin/New York: De Gruyter, S. 119-139.

Müller, Stefan (2007): Qualitative Korpusanalyse für die Grammatiktheorie. Introspektion vs. Korpus. In: Kallmeyer/Zifonun (Hg.), S. 70-90.

Müller, Stefan/Meurers, Walt Detmar (2006): Corpus evidence for syntactic structures and requirements for annotations of tree banks. In: Proceedings of the International Conference on Linguistic Evidence. Tübingen: Universität Tübingen. www.sfs.uni-tuebingen.de/~dm/papers/mueller-meurers-06.pdf (Stand: 24.9.2018).

Naumann, Sven (2003): XML als Beschreibungssprache syntaktisch annotierter Korpora. In: LDV Forum 18, 1, 2, S. 376-390.

Nelson, Mike (2012): Building a written corpus. What are the basics? In: O'Keeffe/McCarthy (Hg.), S. 53-65.

NISO (2004): Understanding metadata. Baltimore: National Information Standards Organization.

O'Keeffe, Anne/McCarthy, Michael (Hg.) (2012): The Routledge handbook of corpus linguistics. Milton Park/Abingdon/New York: Routledge.

Ogura, Kanayo/Nishimoto, Kazushi (2004): Is a face-to-face conversation model applicable to chat conversations? In: Proceedings of the 18th PRICAI2004 Workshop on Language Sense on Computer, S. 26-31. http://ultimavi.arc.net.my/banana/Workshop/PRICAI2004/Final/ogura.pdf (Stand: 1.2.2019).

Pareto, Vilfredo (2007): Ausgewählte Schriften. Wiesbaden: Verlag für Sozialwissenschaften.

Pasupuleti, Pradeep (2014): Pig design patterns. Simplify Hadoop programming to create complex end-to-end enterprise big data solutions with Pig. Birmingham: Packt Publishing.

Paumier, Sébastien (2003): Unitex User Manual. English translation by the local grammar group at the CIS, Ludwig-Maximilians-Universität, Munich. München: LMU. www.cis.uni-muenchen.de/people/lg3/ManuelUnitex.pdf (Stand: 27.9.2018).

Pavlo, Andrew et al. (2009): A comparison of approaches to large-scale data analysis. In: Proceedings of the 2009 ACM SIGMOD international conference on management of data, Providence, RI. New York: ACM, S. 165-178.

Perkuhn, Rainer/Belica, Cyril (2004): Eine kurze Einführung in die Kookkurrenzanalyse und syntagmatische Muster. Mannheim: Institut für Deutsche Sprache. www.ids-mannheim.de/kl/misc/tutorial.html (Stand: 27.9.2018).

Perkuhn, Rainer/Keibel, Holger/Kupietz, Marc (2012): Korpuslinguistik. Paderborn: Fink.

Pezik, Piotr (2014): Graph-based analysis of collocational profiles. In: Jesenšek, Vida/Grzybek, Peter (Hg.): Phraseologie im Wörterbuch und Korpus. Phraseology in dictionaries and corpora. Maribor u.a.: University of Maribor, S. 227-243.

Pitman, Jim/Rizzolo, Douglas (2015): Schröder's problems and scaling limits of random trees. In: Transactions of the American Mathematical Society 367, 10, S. 6943-6969.

Pol, Sagar J./Suryawanshi, Rakesh (2015): Implementation of the map reduce paradigm techniques in big data. In: International Journal of Scientific Engineering and Research 3, 6, S. 108-113.

Pomikálek, Jan/Rychlý, Pavel/Jakubícek, Miloš (2012): Building a 70 billion word corpus of English from ClueWeb. In: Calzolari (Hg.), S. 502-506.

Porta, Jordi (2014): From several hundred million to some billion words. Scaling up a corpus indexer and a search engine with MapReduce. In: Kupietz et al. (Hg.)S. 25-29.

Prün, Claudia (2005): Das Werk von G. K. Zipf (The work of G. K. Zipf). In: Köhler/Altmann/Piotrovskii (Hg.), S. 142-151.

Quasthoff, Uwe (1998): Projekt der Deutsche Wortschatz. In: Heyer, Gerhard/Wolff, Christian (Hg.): Linguistik und neue Medien. Wiesbaden: DUV, S. 93-99.

Quasthoff, Uwe/Wolff, Christian (1999): Korpuslinguistik und große einsprachige Wörterbücher. In: Linguistik online 3. DOI: https://doi.org/10.13092/lo.3.1038. https://bop.unibe.ch/linguistik-online/article/view/1038/1702 (Stand: 13.12.2018).

Quasthoff, Uwe/Goldhahn, Dirk/Eckart, Thomas (2015): Building large resources for text mining. The Leipzig Corpora Collection. In: Biemann, Chris/Mehler, Alexander (Hg.): Text mining. From ontology learning to automated text processing applications. Heidelberg/New York: Springer, S. 3-24.

Quasthoff, Uwe/Richter, Matthias/Biemann, Chris (2006): Corpus portal for search in monolingual corpora. In: Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006). Genua: European Language Resources Association (ELRA), S. 1799-1802.

R Core Team (Hg.) (2016): R: A language and environment for statistical computing. R foundation for statistical computing. Wien. www.R-project.org (Stand: 1.10.2018).

Rahm, Erhard (1993): Hochleistungs-Transaktionssysteme. Konzepte und Entwicklungen moderner Datenbankarchitekturen. Wiesbaden: Vieweg+Teubner.

Rahm, Erhard/Vossen, Gottfried (2003): Web und Datenbanken. Konzepte, Architekturen, Anwendungen. Heidelberg: Dpunkt-Verlag.

Rahm, Erhard/Saake, Gunter/Sattler, Kai-Uwe (2015): Verteiltes und Paralleles Datenmanagement. Von verteilten Datenbanken zu Big Data und Cloud. Berlin/Heidelberg: Springer.

Randall, Beth (2009): CorpusSearch 2. A tool for linguistic research. http://corpussearch.sourceforge.net/CS-manual/CorpusSearch_Guide.pdf (Stand: 1.10.2018).

Ranger, Colby et al. (2007): Evaluating MapReduce for multi-core and multiprocessor systems. In: 2007 IEEE 13th International Symposium on High Performance Computer Architecture. Scottsdale, AZ: IEEE, S. 13-24.

Rauber, Thomas/Rünger, Gudula (2000): Parallele und verteilte Programmierung. Berlin: Springer.

Rehm, Georg et al. (2009): Sustainability of annotated resources in linguistics. A web-platform for preserving, exploring, visualising, and querying linguistic corpora and other resources. In: Literary and Linguistic Computing 24, S. 193-210.

Renouf, Antoinette/Kehoe, Andrew (2013): Using the WebCorp Linguist's Search Engine to supplement existing text resources. In: International Journal of Corpus Linguistics 18, 2, S. 167-198.

Richards, Brian (1987): Type/Token ratios. What do they really tell us? In: Journal of Child Language 14, S. 201-209.

Richter, Matthias et al. (2006): Exploiting the Leipzig Corpora Collection. In: Tomaz, Erjavec/Zganec Gros, Jerneja (Hg.): Language technologies. Proceedings of the IS-LTC 2006 (Fifth Slovenian and First International Language Technologies Conference). Ljubljana: Informacijska Družba. http://nl.ijs.si/is-ltc06/proc/13_Richter.pdf (Stand: 1.10.2018).

Ritter, Norbert et al. (Hg.) (2015): Datenbanksysteme für Business, Technologie und Web (BTW 2015) – Workshopband. Bonn: Gesellschaft für Informatik.

Rohde, Douglas L.T. (2005): TGrep2 user manual. Version 1.15. http://tedlab.mit.edu/~dr/Tgrep2/tgrep2.pdf (Stand: 1.10.2018).

Rosenfeld, Viktor (2010): An implementation of the annis 2 query language. Diplomarbeit, Humboldt-Universität zu Berlin. https://pdfs.semanticscholar.org/fe56/4157c6c986ea0d1ebf9f0593f5620e0f26a1.pdf (Stand: 1.2.2019).

Rychlý, Pavel (2007): Manatee/Bonito. A modular corpus manager. In: Proceedings of the first workshop on recent advancesin Slavonic natural language processing. Brno: Masaryk University, S. 65-70.

Schäfer, Roland (2015): Processing and querying large web corpora with the COW14 architecture. In: Bański et al. (Hg.), S. 28-34.

Schäfer, Roland (2016): CommonCOW. Massively huge web corpora from CommonCrawl data and a method to distribute them freely under restrictive EU copyright laws. In: Calzolari, Nicoletta (Hg.): Proceedings of the 10th Conference on International Language Resources and Evaluation (LREC 2016). Mannheim: Institut für Deutsche Sprache, S. 4500-4504. http://rolandschaefer.net/wp-content/uploads/2015/10/Scha%CC%88fer_2016_CommonCOW_LREC.pdf (Stand: 1.2.2019).

Schäfer, Roland/Bildhauer, Felix (2012): Building large corpora from the web using a new efficient tool chain. In: Calzolari (Hg.), S. 486-493.

Schäfer, Roland/Bildhauer, Felix (2013): Web corpus construction. Synthesis lectures on human language technologies. San Francisco: Morgan and Claypool.

Sardinha, Tony Berber/Pinto, Marcia Veirano (Hg.) (2014): Multi-dimensional analysis, 25 years on. A tribute to Douglas Biber. (= Studies in Corpus Linguistics 60). Amsterdam: Benjamins.

Scherer, Carmen (2006): Korpuslinguistik. (= Kurze Einführungen in die germanistische Linguistik 2). Heidelberg: Winter.

Schiller, Anne et al. (1999): Guidelines für das Tagging deutscher Textcorpora mit STTS. Kleines und großes Tagset. Stuttgart: Institut für maschinelle Sprachverarbeitung. www.sfs.uni-tuebingen.de/resources/stts-1999.pdf (Stand: 1.2.2019).

Schneider, Roman (2012): Evaluating DBMS-based access strategies to very large multilayer annotated corpora. In: Calzolari, Nicoletta et al. (Hg.): Proceedings of the 8th Conference on International Language Resources and Evaluation (LREC 2012) workshop „Challenges in the Management of Large Corpora (CMLC-1)". Paris: European Language Resources Association (ELRA), S. 35-48.

Schneider, Roman (2014): GenitivDB. A corpus-generated database for german genitive classification. In: Calzolari (Hg.), S. 988-994.

Schneider, Roman (2018): Example-based querying for specialist corpora. In: Bański et al. (Hg.), S. 26-32.

Schneider, Roman/Storrer, Angelika/Mehler, Alexander (Hg.) (2013): Webkorpora in Computerlinguistik und Sprachforschung. Web corpora for computational linguistics and linguistic research. (= Journal for Language Technology and Computational Linguistics 28, 2). https://jlcl.org/content/2-allissues/8-Heft2-2013/H2013-2.pdf (Stand: 1.2.2019).

Schnober, Carsten (2012): Using information retrieval technology for a corpus analysis platform. In: Proceedings of KONVENS 2012, Vienna. Wien: ÖGAI, S. 199-207. www.oegai.at/konvens2012/proceedings/27_schnober12p/ (Stand: 1.10.2018).

Schreibman, Susan/Siemens, Raymond George/Unsworth, John (Hg.) (2016): A new companion to digital humanities. (= Blackwell Companions to Literature and Culture 93). Malden/Oxford/Carlton: Wiley-Blackwell.

Schröder, Ernst (1870): Vier kombinatorische Probleme. In: Zeitschrift für Mathematik und Physik 15, S. 361-376. http://resolver.sub.uni-goettingen.de/purl?PPN599415665_0015 (Stand: 1.10.2018).

Sharma, Vivek (2005): Bitmap index vs. B-tree index. Which and when? Redwood Shores: Oracle. www.oracle.com/technetwork/articles/sharma-indexes-093638.html (Stand: 1.10.2018).

Shriparv, Shashwat (2014): Learning HBase. Learn the fundamentals of HBase administration and development with the help of real-time scenarios. Birmingham: Packt Publishing.

Sinclair, John (1991): Corpus, concordance, collocation. Oxford: Oxford University Press.

Sinclair, John (2005): Corpus and text. Basic principles. In: Wynne (Hg.), S. 1-16.

Sokirko, Alexey (2003): DDC. A search engine for linguistically annotated corpora. In: Proceedings of Dialogue 2003 (= International Conference on Computational Linguistics 9). Protvino: Russian State University Moscow.

Stadler, Heike (2014): Die Erstellung der Basislemmaliste der neuhochdeutschen Standardsprache aus mehrfach linguistisch annotierten Korpora. (= OPAL – Online publizierte Arbeiten zur Linguistik 5/2014). Mannheim: Institut für Deutsche Sprache. http://pub.ids-mannheim.de/laufend/opal/opal14-5.html (Stand: 1.2.2019).

Stede, Manfred (2004): The Potsdam Commentary Corpus. In: Webber Bonnie/Byron, Donna K. (Hg.): Proceedings of the 2004 ACL Workshop on Discourse Annotation. Stroudsburg, PA: Association for Computational Linguistics, S. 96-102.

Stede, Manfred (2007): Korpusgestützte Textanalyse. Grundzüge der Ebenen-orientierten Textlinguistik. Tübingen: Narr.

Stede, Manfred (Hg.) (2016): Handbuch Textannotation. Potsdamer Kommentarkorpus 2.0. Potsdam: Universitätsverlag.

Steiner, Ilona/Kallmeyer, Laura (2002): VIQTORYA. A visual query tool for syntactically annotated corpora. In: Proceedings of the 3rd International Conference Language Resources and Evaluation (LREC 2002). Las Palmas: ELRA, S. 1704-1711.

Štícha, František (2008): Usage, frequency, and grammaticality. In: Štícha, František/Fried, Mirjam (Hg.): Grammar and Corpora 2007. Selected contributions from the conference Grammar and Corpora. Prag: Academia, S. 285-291.

Stonebraker, Michael et al. (2007): The end of an architectural era: (it's time for a complete rewrite). In: Koch, Christoph et al. (Hg.): Proceedings of 33rd International Conference on Very Large Data Bases (VLDB). Burlington: Morgan Kaufmann Publishers, S. 1150-1160.

Stonebraker, Michael et al. (2010): MapReduce and parallel DBMSs: friends or foes? In: Communications of the ACM 1, S. 64-71.

Storjohann, Petra (2012a): Der Einsatz verschiedener Korpusmethoden und -verfahren zur Qualitäts- und Konsistenzsicherung am Beispiel der Ermittlung und Dokumentation von Synonymen und Antonymen. In: Gouws, Rufus Hjalmar et al. (Hg.): Lexicographica. International annual for lexicography. Berlin: De Gruyter, S. 121-139.

Storjohann, Petra (2012b): Dornseiff. Der deutsche Wortschatz nach Sachgruppen. In: Haß, Ulrike (Hg.): Große Lexika und Wörterbücher Europas: Europäische Enzyklopädien und Wörterbücher in historischen Porträts. Berlin/Boston: De Gruyter, S. 477-490.

Storrer, Angelika (2011): Korpusgestützte Sprachanalyse in Lexikographie und Phraseologie. In: Knapp, Karlfried/Antos, Gerd/Becker-Mrotzek, Michael (Hg.): Angewandte Linguistik. Ein Lehrbuch. 3., vollst. überarb. u. erw. Aufl. Tübingen/Basel: Francke, S. 216-239.

Strecker, Bruno (2011): Korpusgrammatik zwischen reiner Statistik und „intelligenter“ Grammatikografie. In: Konopka et al. (Hg.), S. 23-46.

Stührenberg, Maik (2012): Auszeichnungssprachen für linguistische Korpora. Theoretische Grundlagen, De-facto-Standards, Normen. Diss. Universität Bielefeld. http://pub.uni-bielefeld.de/download/2492772/2492773 (Stand: 1.10.2018).

Suchowolec, Karolina/Lang, Christian/Schneider, Roman (2018): An empirically validated, onomasiologically structured, and linguistically motivated online terminology. Re-designing scientific resources on German grammar. In: International Journal on Digital Libraries. Special Issue on Networked Knowledge Organization Systems. Berlin/Heidelberg: Springer, S. 1-16.

Suri, Pushpa/Sharma, Divyesh (2012): A model mapping approach for storing XML documents in relational databases. In: International Journal of Computer Science Issues 9, 3, S. 495-498.

Szmrecsanyi, Benedikt (2013): Variation und Wandel. In: Auer, Peter (Hg.): Sprachwissenschaft. Grammatik - Interaktion - Kognition. Stuttgart: Metzler, S. 261-284.

Telljohann, Heike et al. (2009): Stylebook for the Tübingen treebank of written German (TBa-D/Z). Universität Tübingen. www.sfs.uni-tuebingen.de/fileadmin/static/ascl/resources/tuebadz-stylebook-0911.pdf (Stand: 1.10.2018).

Telljohann, Heike et al. (2013): STTS als Part-of-Speech-Tagset in Tübinger Baumbanken. In: Journal for Language Technology and Computational Linguistics 28, 1, S. 1-16.

The Inquirer (Hg.) (2005): Gordon Moore says aloha to Moore’s Law. www.theinquirer.net/inquirer/news/1014782/gordon-moore-aloha-moore-law (Stand: 1.10.2018).

Tidwell, Douglas (2008): XSLT. 2. Aufl. Sebastopol, CA: O’Reilly Media.

Tognini-Bonelli, Elena (2001): Corpus linguistics at work. (= Studies in Corpus Linguistics 6). Amsterdam: Benjamins.

Trippel, Thorsten/Hoppermann, Christina/Depoorter, Griet (2012): The component metadata infrastructure (CMDI) in a project on sustainable linguistic resources. In: Calzolari, Nicoletta (Hg.): Proceedings of the 8th Conference on International Language Resources and Evaluation (LREC 2012) workshop „Describing Language Resources with Metadata“. Mannheim: Institut für Deutsche Sprache, S. 29-36.

Truskkina, Julia (2004): Morpho-syntactic annotation and dependency parsing of German. Diss. Tübingen: Eberhard Karls Universität.

Verma, Abhishek et al. (2013): Breaking the MapReduce stage barrier. In: Cluster Computing 16, 1, S. 191-206.

Volk, Martin (2002): Using the web as corpus for linguistic research. In: Pajusalu, Renate/Hennoste, Tiit (Hg.): Tähendusepüüdja. catcher of the meaning. A Festschrift for Professor Haldur Õim. (= Publications of the Department of General Linguistics 3). Tartu: University of Tartu.

Wamsley, Priscilla (2007): XQuery. Beijing: O'Reilly Media.

Wartala, Ramon (2012): Hadoop. Zuverlässige, verteilte und skalierbare Big-Data-Anwendungen. München: Open Source Press.

Wattam, Stephen/Rayson, Paul/Berridge, Damon (2013): Using life-logging to re-imagine representativeness in corpus design. In: Hardie, Andrew/Love, Robbie (Hg.): Corpus Linguistics 2013 abstract book. Lancaster: Lancaster University Centre for Computer Corpus Research on Language (UCREL), S. 290-293. http://ucrel.lancs.ac.uk/cl2013/doc/CL2013-ABSTRACT-BOOK.pdf (Stand: 14.12.2018).

Weiß, Christian (2005): Die thematische Erschließung von Sprachkorpora (OPAL – Online publizierte Arbeiten zur Linguistik 1/2005). Mannheim: Institut für Deutsche Sprache. http://pub.ids-mannheim.de/laufend/opal/pdf/opal2005-1.pdf (Stand: 1.10.2018).

White, Tom (2015): Hadoop. The definitive guide. 4. Aufl. Sebastopol, CA: O'Reilly.

Wickmann, Dieter (1989): Computergestützte Philologie. Bestimmung der Echtheit und Datierung von Texten. In: Bátori, Istvan S./Lenders, Winfried/Putschke, Wolfgang (Hg.): Computational linguistics / Computerlinguistik. An international handbook on computer oriented language research and applications / Ein internationales Handbuch zur computergestützten Sprachforschung und ihrer Anwendungen. (= Handbücher zur Sprach- und Kommunikationswissenschaft 4). Berlin/New York: De Gruyter, S. 528-534.

Witt, Andreas (2002): Multiple Informationsstrukturierung mit Auszeichnungssprachen. XML-basierte Methoden und deren Nutzen für die Sprachtechnologie. Diss. Universität Bielefeld. http://d-nb.info/963909436/34 (Stand: 1.10.2018).

Witten, Ian H./Moffat, Alistair/Bell, Timothy C. (1999): Managing gigabytes. Compressing and indexing documents and images. 2. Aufl. San Francisco: Morgan Kaufmann.

Wöllstein, Angelika et al. (Hg.) (2018): Grammatiktheorie und Empirie in der Germanistischen Linguistik. (= Germanistische Sprachwissenschaft um 2020 1). Berlin/New York: De Gruyter.

Wynne, Martin (Hg.) (2005): Developing linguistic corpora. A guide to good practice. Oxford: Oxbow Books. www.ahds.ac.uk/creating/guides/linguistic-corpora/index.htm (Stand: 1.10.2018).

Yang, Hung-chih et al. (2007): Map-reduce-merge: simplified relational data processing on large clusters. In: Zhou, Lizhu/Ling, Tok Wang/Ooi, Beng Chin (Hg.): Proceedings of the 2007 ACM SIGMOD international conference on Management of data. Beijing: ACM, S. 1029-1040.

Yoshikawa, Masatoshi/Amagasa, Toshiyuki (2001): XRel. A path-based approach to storage and retrieval of XML documents using relational databases. In: ACM Transactions on Internet Technology 1, S. 110-141.

Zeldes, Amir et al. (2009): ANNIS. A search tool for multi-layer annotated corpora. In: Mahlberg, Michaela/González-Díaz, Victorina/Smith, Catherine (Hg.): Proceedings

of the Corpus Linguistics Conference 2009, University of Liverpool, UK, 20-23 July 2009. Liverpool: University of Liverpool.

Zenkov, Andrei V. (2017): Method of text attribution based on the statistics of numerals. In: Journal of Quantitative Linguistics 25, 3, S. 256-270.

Zesch, Torsten/Gurevych, Iryna (2010): The more the better? Assessing the influence of Wikipedia's growth on semantic relatedness measures. In: Calzolari (Hg.), S. 1374-1380.

Zesch, Torsten/Horsmann, Tobias (2016): FlexTag. A highly flexible pos tagging framework. https://pdfs.semanticscholar.org/9c48/fb5b87b2d07af7223ff72cf4743d7d24c22d.pdf (Stand: 1.2.2019).

Zhang, Chun et al. (2001): On supporting containment queries in relational database management systems. In: ACM SIGMOD Record 30, 2, S. 425-436.

Zierl, Marco (1997): Entwicklung und Implementierung eines Datenbanksystems zur Speicherung und Verarbeitung von Textkorpora. Magisterarbeit, Philosophische Fakultät II, Friedrich-Alexander-Universität, Erlangen-Nürnberg.

Ziesche, Peter (2005): Nebenläufige und verteilte Programmierung. Konzepte, UML 2-Modellierung, Realisierung mit Java 1.4 und Java 5. Herdecke. Bochum: W3L-Verlag.

Zinsmeister, Heike (2010): Korpora. In: Carstensen et al. (Hg.), S. 482-491.

Zipf, George Kingsley (1949): Human behaviour and the principle of least effort. Reading, MA: Addison-Wesley.

Corpus Linguistics and Interdisciplinary Perspectives on Language

Volume 8

Digitale Korpora haben die Voraussetzungen, unter denen sich Wissenschaftler mit der Erforschung von Sprachphänomenen beschäftigen, fundamental verändert. Umfangreiche Sammlungen geschriebener und gesprochener Sprache bilden mittlerweile die empirische Basis für mathematisch präzise Generalisierungen über zu beschreibende Wirklichkeitsausschnitte. Das Datenmaterial ist hochkomplex und besteht neben den Rohtexten aus diversen linguistischen Annotationsebenen sowie außersprachlichen Metadaten. Als unmittelbare Folge stellt sich die Konzeption adäquater Recherchelösungen als beträchtliche Herausforderung dar. Im vorliegenden Buch wird deshalb ein datenbankbasierter Ansatz vorgestellt, der sich der Problematiken multidimensionaler Korpusrecherchen annimmt. Ausgehend von einer Charakterisierung der Anforderungsmerkmale linguistisch motivierter Suchen werden Speicherungs- und Abfragestrategien für mehrfach annotierte Korpora entwickelt und anhand eines linguistischen Anforderungskatalogs evaluiert. Ein Schwerpunkt liegt dabei in der Einführung problemorientierter Segmentierung und Parallelisierung.

ISBN 978-3-8233-8286-7

narr\f
ranck
e\atte
mpto