Universität Leipzig
Fakultät für Mathematik und Informatik
Institut für Informatik

Automatischer Aufbau eines multilingualen Thesaurus durch Extraktion semantischer und lexikalischer Relationen aus der Wikipedia

Diplomarbeit

Leipzig, Mai 2008

Vorgelegt von:
Daniel Kinzler
Studiengang Informatik

Kurzreferat

Die vorliegende Diplomarbeit beschreibt und analysiert Methoden, um aus den Datenbeständen der Wikipedia in verschiedenen Sprachen einen multilingualen Thesaurus zu erstellen. Dabei sollen insbesondere die Beziehungen zwischen Termen (Wörtern, Wortformen, Phrasen) zu sprachunabhängigen Konzepten extrahiert werden sowie die Beziehungen zwischen solchen Konzepten, speziell Beziehungen der Über- bzw. Unterordnung (Subsumtion) sowie der semantischen Verwandtheit und Ähnlichkeit. Zu diesem Zweck werden die Anforderungen sowie die verfügbaren Rohdaten analysiert, ein Prototyp zur Extraktion der gewünschten Daten entwickelt und die mit dem Prototyp gewonnenen Daten in Bezug auf die zuvor formulierten Anforderungen evaluiert.

Schlagwörter

Wikipedia; Thesaurus; Sprachverarbeitung; Information Retrieval; Übersetzung

CR-Klassifikation

H.3.1 Content Analysis and Indexing
H.3.3 Information Search and Retrieval
I.2.7 Natural Language Processing

Zitation: Daniel Kinzler, *Automatischer Aufbau eines multilingualen Thesaurus durch Extraktion semantischer und lexikalischer Relationen aus der Wikipedia*, Diplomarbeit an der Abteilung für Automatische Sprachverarbeitung, Institut für Informatik, Universität Leipzig, 2008.

```
@mastersthesis{ kinzler2008th,
  author = "Daniel Kinzler",
  title  = "Automatischer Aufbau eines multilingualen Thesaurus durch Extraktion
            semantischer und lexikalischer Relationen aus der Wikipedia",
  school = "Universit\"{a}t Leipzig",
  year   = "2008",
  url = "http://brightbyte.de/papers/2008/DA/WikiWord.pdf"
}
```

URI: <http://brightbyte.de/papers/2008/DA/WikiWord.pdf>

Inhaltsverzeichnis

Menon:
Und auf welche Weise willst du denn dasjenige suchen, Sokrates, wovon du überhaupt gar nicht weißt, was es ist?
Denn als welches Besondere von allem, was du nicht weißt, willst du es dir denn vorlegen und so suchen?
Oder wenn du es auch noch so gut träfest, wie willst du denn erkennen, daß es dieses ist, was du nicht wußtest?

. . .

Sokrates:
Da die ganze Natur unter sich verwandt ist
und die Seele alles innegehabt hat:
so hindert nichts, daß, wer nur an ein einziges erinnert wird,
was bei den Menschen Lernen heißt,
alles übrige selbst auffinde, wenn er nur tapfer ist
und nicht ermüdet im Suchen.
Denn das Suchen und Lernen ist demnach ganz und gar Erinnerung.

— Platon[1]

[1] *Platon*: *Menon*; Übersetzung von *Friedrich Schleiermacher*

Teil I.

Einleitung

1. Grundlagen

Die automatische Verarbeitung von natürlichsprachlichen Texten hat in den letzten Jahren an Bedeutung gewonnen: Aus der immer größer werdenden Flut von Texten erwächst die Notwendigkeit, bessere Methoden zu entwickeln, um Texte automatisch zu erfassen, zu filtern, zu suchen, zu indizieren und zusammenzufassen. Dabei hat sich gezeigt, dass neben geeigneten Verfahren und ausreichender Speicher- und Verarbeitungskapazität auch maschinell verarbeitbares *Wissen* von entscheidender Bedeutung für die Verarbeitung natürlicher Sprache ist [Gabrilovich (2006), Ide und Veronis (1998)]. Solches Wissen reicht von formalen Ontologien, wie sie in Expertensystemen und zur Wissensrepräsentation für Künstliche Intelligenz eingesetzt werden, zu „weichem“, „flachem“ Sprachwissen, z. B. darüber, was ein Name ist, wo ein Satz endet oder welche Wörter in welcher Sprache für welches Konzept stehen.

Der Nutzen, den so genanntes *Weltwissen* für Softwaresysteme hat, wird in der *Knowledge as Power Hypothesis* von *B. G. Buchanan* zusammengefasst [Buchanan und Feigenbaum (1982)]:

> *The power of an intelligent program to perform its task well depends primarily on the quantity and quality of knowledge it has about that task.*

Und *D.B. Lenat* sagt über die Notwendigkeit zu generalisieren, also den Nutzen seiner Begriffshierarchie [Lenat und Feigenbaum (1991)]:

> *To behave intelligently in unexpected situations, an agent must be capable of falling back on increasingly general knowledge.*

Weiter sagt *Lenat*, dass Programme ohne Wissen sehr anfällig (*brittle*) seien, und nur Aufgaben bewältigen können, die vom Programmierer vollständig vorhergesehen wurden [Lenat u. a. (1990)].

Ziel der vorliegenden Diplomarbeit ist es nun, eine Sammlung solchen Wissens, nämlich einen *Thesaurus*, automatisch aus dem Datenbestand der Wikipedia zu erzeugen. Die Wikipedia bietet sich für diesen Zweck an, da sie einen großen, aktuellen und wohlstrukturierten Korpus darstellt, der zudem frei und in einer Vielzahl von Sprachen verfügbar ist.

Die Idee ist, ein Softwaresystem (im Folgenden: *WikiWord*) zu entwerfen, das in der Lage ist, Wissen über Bezeichnungen für und die Beziehungen zwischen Konzepten aus der Wikipedia zu extrahieren. Dieses Wissen ist in der Struktur und den Inhalten der Wikipedia mehr oder weniger explizit enthalten: Beziehungen werden durch spezielle syntaktische Konstrukte (*Markup*) sowie Konventionen über den Aufbau und die Formatierung von Inhalten repräsentiert. Ziel dieser Diplomarbeit ist es, Methoden zur Auswertung dieser Strukturen zu entwerfen und zu evaluieren, insbesondere solche Methoden, die ohne eine Betrachtung von natürlicher Sprache, also der eigentlichen textuellen Inhalte, auskommen. Ein Entwurf für ein Softwaresystem, das solche Methoden umsetzt, ist in ▸II beschrieben, detaillierte Informationen über die konkrete Implementation von WikiWord finden sich unter ▸III. Die Evaluation wird in ▸IV erläutert.

Bevor der Entwurf und die Implementation von WikiWord dargelegt werden können, werden in den verbleibenden Abschnitten dieses Kapitels Grundbegriffe geklärt (▹1.1), ein Überblick über wichtige Eigenschaften der Wikipedia geboten (▹1.2) sowie verschiedene Arten und Aspekte von Thesauri betrachtet (▹1.3). Das folgende Kapitel gibt dann eine Übersicht über den Stand der Forschung in Hinblick auf die automatische Erstellung von Thesauri im Allgemeinen und die Extraktion von Wissen aus der Wikipedia im Speziellen (▹2).

1.1. Begriffe

Zunächst eine kurze Übersicht über Begriffe, die in dieser Arbeit spezielle Bedeutung haben:

Wiki: Im Allgemeinen eine Webseite, die von jedem frei bearbeitet werden kann [Leuf und Cunningham (2001)]; im Speziellen ein bestimmtes Wikipedia-Projekt (siehe unten).

Wikipedia: Ein Projekt zur kooperativen Erstellung einer Enzyklopädie unter Verwendung der Wiki-Technologie (▹1.2). Je nach Kontext bezeichnet *Wikipedia* ein bestimmtes Wikipedia-Projekt in einer bestimmten Sprache (siehe unten).

MediaWiki: MediaWiki ist die Software, auf der die Wikipedia-Wikis basieren und die für diesen Zweck von der Wikimedia Foundation entwickelt wird [MW:ABOUT]. MediaWiki ist freie Software, lizenziert unter der GPL [GPL].

Wikimedia: Die Wikimedia Foundation ist eine gemeinnützige Stiftung mit Sitz in St. Petersburg, Florida, USA, deren Zweck die Förderung von freiem Wissen ist [WM:ABOUT, OKD, FCW]. Sie betreibt Anfang 2008 über 750 Wiki-Projekte [M:SITES, WM:PROJECTS] zu unterschiedlichen Themen in unterschiedlichen Sprachen, unter anderem auch die Wikipedia-Projekte.

WikiWord: Das im Rahmen dieser Diplomarbeit entwickelte Softwaresystem zur automatischen Extraktion eines multilingualen Thesaurus aus der Wikipedia.

Projekt: Ein (Wiki-)*Projekt* ist eine konkrete Website, auf der ein Wiki-System verwendet wird und eine Gemeinschaft von Benutzern gemeinsam Inhalte zu einem bestimmten Thema erstellt — zum Beispiel die deutschsprachige Wikipedia (`de.wikipedia.org`). Im Falle der Wikipedia korrespondiert ein konkretes Projekt jeweils mit einer Sprache, in der enzyklopädische Inhalte erstellt werden. Diese Inhalte dienen als Korpus im Sinne dieser Arbeit (siehe unten).

Sprache: Ein Wikipedia-Projekt ist immer mit der Sprache assoziiert, in der seine Inhalte verfasst sind. Jeder Sprache ist ein Code zugeordnet, der als Präfix für Language-Links und als Subdomain dient (▹1.2). Diese Codes folgen im Wesentlichen der ISO 639-1 Norm [ISO:639-1 (2002), M:SITES]. Zum Beispiel verwendet die deutschsprachige Wikipedia den Code `de` (und damit die Domain `de.wikipedia.org`). Ein Sprach-Code identifiziert also auch ein Wiki-Projekt und damit einen Korpus, aus dem ein (lokaler) Datensatz für den Thesaurus extrahiert werden kann.

Korpus: Ein Korpus ist im Kontext dieser Arbeit eine Sammlung von Wiki-Seiten, die Gegenstand der Analyse zwecks Erzeugung eines (lokalen) Datensatzes für den Thesaurus sein kann. Ein Korpus ist immer einer Sprache und einem Wiki-Projekt zugeordnet; mit dem einer Sprache zugeordneten Korpus ist daher die Menge von Wiki-Seiten gemeint, die in dem zu der Sprache gehörigen Wikipedia-Projekt enthalten sind.

Seite: Eine (Wiki-)Seite ist eine (Wiki-)Text-*Ressource* mit einem eindeutigen Namen innerhalb eines Wiki-Projektes und damit einer eindeutigen URL. Sie kann analysiert werden, um aus ihr Informationen zu gewinnen, die dann zum Aufbau eines Thesaurus verwendet werden können (siehe ▹8).

Im Folgenden werden die Namen von Wiki-Seiten serifenlos und kursiv gesetzt, zum Beispiel: *Sprachwissenschaft*.

Artikel: Ein Artikel ist eine Wiki-Seite, die ein Konzept definiert. Nicht alle Seiten sind Artikel, prominente Gegenbeispiele sind Weiterleitungen (Redirects), Begriffsklärungen (Disambiguations) und Kategorien (siehe ▹1.2, ▹8.5); weitere Gegenbeispiele sind Wiki-Seiten, die im Zuge dieser Arbeit nicht betrachtet werden, wie zum Beispiel Benutzerseiten.

Konzept: Ein Konzept ist im Kontext dieser Arbeit ein (abstrakter oder konkreter) Betrachtungsgegenstand, in der Wikipedia insbesondere Gegenstand von Artikeln. Konzepte haben eine Menge von Termen als *Bezeichnungen*, sie sind somit die *Bedeutung* der ihnen zugeordneten Terme, also das *Bezeichnete* (auch *Signifikat*, fr. *Signifié*) im Sinne der Semiotik *de Saussures* [de Saussure (2001), s. 76ff]. Ein wesentlicher Aspekt dieser Arbeit ist es, diese Beziehung zwischen Konzept und Term zu explizieren, also eine *Bedeutungsrelation* aufzubauen. Gemeinsam mit den Beziehungen zwischen Konzepten bildet die Bedeutungsrelation einen konzeptorientierten Thesaurus [Matthews (2003)] (siehe ▹4.2).

Im Folgenden werden die Namen von Konzepten in Kapitälchen gesetzt, zum Beispiel: Sprachwissenschaft.

Term: Ein Term ist im Kontext dieser Arbeit ein *Wort* oder eine *Wortgruppe* bzw. *Phrase*, die benutzt wird, um auf ein Konzept Bezug zu nehmen — das heißt, er ist eine *Bezeichnung* für ein Konzept, also das *Bezeichnende* (auch *Signifikant*, fr. *Signifiant*) im Sinne der Semiotik *de Saussures* [de Saussure (2001), s. 76ff]. Unterschiedliche Wortformen und Schreibweisen werden dabei als unterschiedliche Terme betrachtet, selbst dann, wenn sie sich nur in der Großschreibung unterscheiden. Die *Bedeutungsrelation* zwischen Termen und Konzepten ist ein wesentlicher Aspekt dieser Arbeit (siehe ▹4.2 und ▹8.9).

Im Folgenden werden Terme kursiv und in Anführungszeichen gesetzt, zum Beispiel: „*Sprachwissenschaft*“.

Dump: Ein (Wikipedia-)Dump ist eine Datei, die eine Menge von Wikipedia-Seiten enthält, das heißt, sie beinhaltet den Wiki-Text der betreffenden Seiten sowie die dazugehörige Meta-Information, eingebettet in eine dafür entworfene XML-Struktur [MW:XML]. Dumps der Wikipedia-Inhalte werden periodisch alle paar Monate erstellt und zur freien Weiternutzung angeboten [WM:DOWNLOAD]. Es gibt für jede Wikipedia unterschiedliche Dumps, die verschiedene Mengen von Seiten enthalten. Für diese

Arbeit sind diejenigen Dumps relevant, die alle Seiten im Hauptnamensraum (die „*Artikel*“) sowie alle Kategorieseiten beinhalten, wobei nur die aktuelle Revision jeder Seite benötigt wird. Die für diese Anwendung zugeschnittenen Dumps folgen dem Namensschema *wikiname*-*datum*-pages-articles.xml, also zum Beispiel dewiki-20080320-pages-articles.xml für den Dump der deutschsprachigen Wikipedia vom 20. März 2008, der die aktuelle Version aller Seiten im Hauptnamensraum sowie Kategorien und Vorlagen enthält.

Datensatz: Im Zusammenhang dieser Diplomarbeit ist ein *Datensatz* eine Sammlung von Informationen über Konzepte und Terme und ihre Beziehungen zueinander. Datensätze gibt es in zwei Ausprägungen: *Lokale* (sprachspezifische) Datensätze enthalten die Daten, die aus einem Korpus in einer bestimmten Sprache extrahiert wurden, sie repräsentieren damit einen monolingualen Thesaurus (siehe ▸4.2). *Globale* Datensätze kombinieren mehrere lokale Datensätze zu einem multilingualen Thesaurus, indem sie äquivalente Konzepte aus den einzelnen Sprachen zu jeweils einem sprachunabhängigen Konzept zusammenfassen (siehe ▸4.3).

Datensätze werden typischerweise nach dem Wiki benannt, auf dessen Basis sie erstellt wurden.

Kollektion: Im Zusammenhang dieser Diplomarbeit ist eine *Kollektion* eine Menge von Datensätzen, die zusammen einen multilingualen Thesaurus bilden. Eine Kollektion enthält einen lokalen Datensatz pro betrachteter Sprache sowie einen globalen Datensatz, der die Konzepte aus den lokalen Datensätzen miteinander verbindet.

Wie in ▸10.1 beschrieben, wird der Name der Kollektion verwendet, um einen universell eindeutigen Bezeichner (URI) für den Datensatz zu erzeugen. Es ist daher zweckmäßig, aussagekräftige und eindeutige Namen für Kollektionen zu verwenden, insbesondere solche, die Informationen über die verwendeten Wikis, deren Version und eventuelle thematische Ausschnitte enthalten.

1.2. Wikipedia

Wikipedia ist eine Online-Enzyklopädie, die von Tausenden von Benutzern in Kollaboration erstellt wird [Fiebig (2005)]. Sie beruht auf dem „Wiki-Prinzip“, also der Idee, dass jeder, auch ohne Anmeldung, die Inhalte sofort bearbeiten und erweitern kann [Leuf und Cunningham (2001)]. Wikipedia verwendet zu diesem Zweck die *MediaWiki*-Software [MW:ABOUT], die von der Wikimedia Foundation [WM:ABOUT] für diesen Zweck entwickelt wird.

Wikipedia zählt Anfang 2008, nur sieben Jahre nach ihrer Gründung im Jahre 2001, mit bis zu fünfundsechzigtausend Seitenaufrufen pro Sekunde zu den zehn meistgenutzten Webseiten weltweit [WM:SURVEY (2008), WM:10M (2008)]. Wikipedia-Projekte gibt es in über 250 Sprachen mit insgesamt über zehn Millionen Artikeln[1] [WM:10M (2008), M:SITES, WM:PROJECTS], davon über

[1] Nach der internen Zählweise von MediaWiki: Alle Artikel im Hauptnamensraum, die keine Weiterleitung sind und mindestens einen Wiki-Link enthalten; Begriffsklärungsseiten werden also mitgezählt.

zwei Millionen in der englischsprachigen und über siebenhunderttausend in der deutschsprachigen Wikipedia [WM:2M (2007), WP:STATS]. Alle Inhalte sind unter der GFDL lizenziert [GFDL] und damit frei verwendbar [OKD, FCW]. Der gesamte Korpus der Wikipedia wird periodisch als XML [MW:XML] in so genannte *Dumps* exportiert und zur Weiternutzung angeboten [WM:DOWNLOAD].

Die kollaborative Natur von Wikipedia erlaubt eine hohe Aktualität der Inhalte sowie ein rapides Wachstum der thematischen Abdeckung. Die Anzahl der Seiten in einer Sprache folgt in der Regel zunächst einer exponentiellen Wachstumskurve [Voss (2005B)] und scheint sich mittlerweile zu linearem Wachstum abzuflachen [WP:STATS]. Für die Zukunft ist damit zu rechnen, dass sich das Wachstum noch weiter verlangsamt, so dass sich insgesamt eine logistische Funktion(„S-Kurve“) für das Wachstum ergibt. Die Qualität einzelner Artikel in der Wikipedia ist recht unterschiedlich, insgesamt jedoch mit der Qualität traditioneller Enzyklopädien vergleichbar [Giles (2005)].

Einträge in der Wikipedia sind recht uniform und folgen gewissen Konventionen: insbesondere beschreibt jede Seite genau ein *Konzept* [WP:ARTIKEL] (Ausnahmen sind *Weiterleitungen* und *Begriffsklärungsseiten*, siehe unten) und beginnt mit einer Begriffsdefinition *(Glosse)* [WP:WSIGA].

Da es sich bei der Wikipedia um eine Enzyklopädie handelt, enthält sie vor allem Substantive, Nominalphrasen und Eigennamen; Verben und Adjektive kommen dagegen kaum vor. Für eine Verwendung im Bereich des Information Retrieval ist das kaum problematisch (und eventuell sogar von Vorteil), da zur Klassifikation und Indexierung von Dokumenten ohnehin hauptsächlich Substantive verwendet werden, weil diese gut geeignet sind, den thematischen Inhalt eines Dokumentes wiederzugeben, siehe [Syed u. a. (2008), Hepp u. a. (2006), Eiron und Mccurley (2003)] sowie [WP:V]. Insbesondere enthält die Wikipedia im Gegensatz zu manuell erstellten Thesauri und Wörterbüchern eine große Zahl von Einträgen über Orte, Personen, Werke und Organisationen — sie eignet sich daher besonders, um Informationen zu *Eigennamen* (*proper nouns* bzw. *Named Entities*) zu finden. Auch enthält sie eine Vielzahl von Fachausdrücken aus den verschiedensten Fachbereichen, die in der Regel in allgemeinen Thesauri und Wörterbüchern nicht enthalten sind.

Eine weitere Eigenheit der Wikipedia betrifft die *Granularität* ihrer Einträge: zwar deckt die Wikipedia einen weiten Bereich menschlichen Wissens ab, da sie aber eine Enzyklopädie ist, bleiben die Einträge aus lexikographischer Sicht eher grob: zum Beispiel werden die Begriffe „*physikalisch*“ und „*Physiker*“ mit unter den Eintrag für „*Physik*“ gefasst. Diese Art von Ungenauigkeit ist aber in Hinblick auf die Indexierung und Klassifikation von Dokumenten, also für eine „oberflächliche“ Analyse von Texten, eher hilfreich als schädlich (vergleiche [Ide und Veronis (1998), s. 22] und [Wilks u. a. (1996), s. 59]). Die *fachliche*, *taxonomische* Granularität aber (ob z. B. die „*Atomphysik*“ einen eigenen Artikel hat oder mit unter „*Physik*“ behandelt wird) hängt vor allem von der Gesamtgröße der Wikipedia ab und variiert damit stark zwischen den verschiedenen Sprachen — ein Umstand, den es bei der Erstellung des globalen Datensatzes zu berücksichtigen gilt (siehe ▸4.3 und ▸9.2).

Weitere Funktionen, Eigenschaften und Konventionen, die Wikipedia zu einer exzellenten Ressource für lexikalisch-semantische Informationen machen und im Zusammenhang mit dieser Arbeit wichtig sind, seien im Folgenden genannt:

Seiten: Ein Wiki enthält ein Netzwerk von (Hypertext-)Seiten, die untereinander eng über Hyperlinks verknüpft sind [Bellomi und Bonato (2005B)]. Jede Seite hat einen eindeutigen

Namen, über den sie innerhalb des Wikis über Wiki-Links angesprochen werden kann [Leuf und Cunningham (2001)] und der als URL-Suffix dient, so dass eine eindeutige URI [RFC:3986 (2005)] für diese Seite entsteht.

Der Name der Seite ist in der Wikipedia das enzyklopädische *Lemma* des Artikels. Im Zuge der vorliegenden Arbeit wird der Seitenname als eindeutiger Bezeichner sowohl für die Seite (also Ressource) als auch für das Konzept, das auf der Seite beschrieben wird, verwendet [W3C (2005), Berners-Lee (1998)]. Welches von beiden gemeint ist, ist im Kontext eindeutig, für eine externe Darstellung muss aber klar zwischen diesen unterschieden werden; mehr dazu in Kapitel ▸10.

Für den Fall, dass die „natürliche“ Bezeichnung zweier Konzepte derselbe Term ist, wird durch einen sogenannten *Klammerzusatz* als Qualifikator ein eindeutiges Lemma geschaffen (ein *Klammer-Lemma*) [WP:NK]. So wird zum Beispiel zwischen *Bock_(Gestell)*, *Bock_(Turngerät)*, *Bock_(Insel)* und *Bock_(Dudelsack)* sowie *Bock_(Familienname)* unterschieden (das Tier dagegen wird unter *Huftiere* behandelt). In einem solchen Fall gibt es zusätzlich eine Begriffsklärungsseite (siehe unten), die diese Bedeutungen aufzählt — sie würde entweder den Namen ohne Klammerzusatz tragen (*Bock*), oder den Zusatz „Begriffsklärung“ (also *Bock_(Begriffsklärung)*) [WP:BKL].

Wiki-Text: Wikis verwenden eine Markup-Sprache, genannt *Wiki-Text*, die sich im Funktionsumfang an HTML orientiert, aber leichter für Menschen zu lesen und zu bearbeiten sein soll [Leuf und Cunningham (2001)]. „Wiki-Text“ ist als allgemeine Bezeichnung zu verstehen, die konkreten Markup-Sprachen verschiedener Wiki-Systeme unterscheiden sich zum Teil erheblich voneinander. Im Kontext dieser Arbeit ist mit Wiki-Text die von MediaWiki verwendete Markup-Sprache gemeint (▸A).

Beispiele für Wiki-Text werden im Folgenden kursiv in einer nichtproportionalen Schrift gesetzt, zum Beispiel: *[[Sprachwissenschaft]]*.

Die Art und Weise, wie Wiki-Markup in der Wikipedia verwendet wird [WP:WSIGA], ist im Vergleich zu Webseiten im Allgemeinen relativ reich an Semantik. Insbesondere die Verwendung von Vorlagen sowie die Zuweisung von Kategorien sind im Kontext dieser Arbeit sehr nützlich.

Wiki-Links: Ein wichtiger Aspekt von Wikis ist die leichte Verknüpfbarkeit von Seiten untereinander [WP:V]. MediaWiki verwendet für diesen Zweck Wiki-Links, die als Name des Zielartikels in doppelten eckigen Klammern geschrieben werden, z. B. *[[Sprache]]*. Die Syntax ist in ▸A.3 genauer beschrieben. Wiki-Links, die auf eine Seite zeigen, die nicht existiert, werden von MediaWiki rot dargestellt. Solche „roten Links“ sind nicht als Fehler zu werten, sondern vielmehr als ein Hinweis auf fehlende Inhalte [WP:RL].

Der Link-Text ist eine wichtige Quelle für die Zuordnung von Termen (dem Link-Text) zu Konzepten (dem Link-Ziel), das heißt, für den Aufbau der Bedeutungsrelation (siehe ▸8.7): Der Link-Text verhält sich zum Link-Ziel ähnlich wie der Titel der Seite (also das Lemma) zu ihrem Inhalt (dem Konzept) [Mihalcea (2007)] sowie [Craswell u. a. (2001), Eiron und Mccurley (2003), Kraft und Zien (2004)].

Namensräume: MediaWiki unterstützt (und Wikipedia nutzt) Namensräume für Wiki-Seiten, um Seiten mit unterschiedlicher Funktion voneinander zu trennen und Namenskonflikte zu

vermeiden [WP:NS]. MediaWiki gibt eine Anzahl von Namensräumen fest vor, darunter solche für Benutzerseiten, Projektseiten, Hilfeseiten, Kategorien und einige mehr [MW:NS]. Jedem Namensraum ist zudem ein entsprechender Diskussionsnamensraum zugeordnet (und jeder Seite entsprechend potentiell eine Diskussionsseite, die dazu dient, die Arbeit an der Seite zu koordinieren). Namensräume werden über einen Präfix am Seitennamen adressiert: Der Link `[[Hilfe:Formatieren]]` verweist auf eine Seite im Namensraum *Hilfe*, die sich mit Textformatierung befasst. Die Namen (Präfixe) der Namensräume sind sprachabhängig, eine Liste der Namen der verwendeten Namensräume ist in jedem XML-Dump enthalten.

Der *eigentliche* Inhalt des Wikis, im Falle der Wikipedia die enzyklopädische Information, befindet sich auf Seiten im *Hauptnamensraum*, dem kein Präfix (oder genauer, der leere String als Präfix) zugeordnet ist. Seiten im Hauptnamensraum werden auch als *Artikel* bezeichnet, insbesondere dann, wenn sie keine Weiterleitungen und keine Begriffsklärungen sind, wenn sie also ein Konzept beschreiben. Solche Artikel beinhalten das Sachwissen der Wikipedia.

Interwiki-Links: Auch Verknüpfungen auf andere Wikis sind mittels sogenannter *Interwiki-Links* bequem möglich: für jedes Wiki, das als Ziel von Interwiki-Links dienen soll, muss dafür ein Präfix zugewiesen werden, der bei der Definition des Wiki-Links verwendet wird (siehe ▹A.3).

Language-Links: Interwiki-Links mit bestimmten Präfixen, die als Sprachcodes definiert sind, werden besonders behandelt: Sie werden als Verknüpfungen auf eine *äquivalente* Seite in einer anderen Sprache in einem anderen Wiki interpretiert (siehe ▹A.3).

Kategorien: Kategorien erlauben es, Wiki-Seiten zu Gruppen zusammenzufassen. Eine Seite kann zu beliebig vielen Kategorien gehören und jeder Kategorie ist eine Kategorieseite zugeordnet, über die sie beschrieben und selbst wieder Kategorien zugeordnet werden kann [WP:KAT]. Um einer Seite eine Kategorie zuzuweisen, wird auf der Seite einfach ein Wiki-Link auf diese Kategorie eingefügt, siehe ▹A.3. *Sort-Keys* (*Sortierschlüssel*), die bei der Zuweisung von Kategorien angegeben wurden, sind eine wichtige Quelle für die Terme, die einem Konzept zuzuordnen sind (siehe auch ▹9.1).

Per Konvention bilden Kategorien in der Wikipedia eine thematische Polyhierarchie [WP:KAT], sie lassen sich damit als semantische Relation der Über- bzw. Unterordnung (*Subsumtion*) zwischen den von den Seiten beschriebenen Konzepten interpretieren. Zum Beispiel wäre das speziellere Konzept Unsinn dem allgemeineren Konzept Semantik untergeordnet. Dabei ist die Art der Unterordnung unspezifiziert, die Relation bildet also keine Taxonomie und lässt sich auch nur sehr eingeschränkt in Ontologien verwenden (vergleiche [Ponzetto und Strube (2007b)]). Diese Relation entspricht der Hyponym-Relation in einem klassischen, Term-basierten Thesaurus, bzw. der NT/BT-Relation nach ISO 2788 [ISO:2788 (1986)]. Sie bildet zusammen mit der Bedeutungsrelation den Kern des Thesaurus (siehe ▹4.2). Zu beachten ist dabei, dass die MediaWiki-Software Zyklen in der Kategoriestruktur zulässt — diese sind zwar per Konvention unerwünscht, können aber vorkommen und müssen bei der Erstellung des Thesaurus entfernt werden (siehe ▹9.1.3).

Neben den inhaltlich relevanten Kategorien existieren in der Wikipedia eine Vielzahl von *Wartungskategorien* wie zum Beispiel *Kategorie:Wikipedia:Lückenhaft*, die sich auf die *Seite* beziehen, nicht auf das beschriebene Konzept. Solche Kategorien sollten zur Erstellung eines Thesaurus nicht verwendet werden und werden daher beim Import nach Möglichkeit ignoriert (siehe ▹9.1).

Weiterleitungen: *Weiterleitungen* (*Redirects*) definieren einen Alias für einen Seitentitel (siehe ▹A.6 für eine Beschreibung der Syntax). Eine Wiki-Seite kann über diesen Mechanismus unter mehreren Namen erreichbar sein. Zum Beispiel könnte *USA* eine Weiterleitung auf *Vereinigte Staaten von Amerika* sein [WP:WL]. Weiterleitungen sind eine wichtige Quelle für Informationen darüber, welche alternativen Bezeichnungen (Terme) für welche Konzepte verwendet werden (▹A.6).

Vorlagen: MediaWiki unterstützt *Vorlagen* (*Templates*) als eine Methode, den Inhalt einer Wiki-Seite in eine andere einzubinden (siehe ▹A.5). Beim Einbinden von Vorlagen können sogenannte *Vorlagen-Parameter* verwendet werden: Sie erlauben es, beim Einbinden konkrete Werte anzugeben, die dann für Platzhalter eingesetzt werden, die im Wiki-Text der Vorlage verwendet wurden [WP:VOR].

Vorlagen werden in der Wikipedia für verschiedene Zwecke benutzt: zur einheitlichen Anzeige von strukturierten Daten (in sogenannten *Infoboxen* oder *Taxoboxen* [WP:IB]), als Navigationselement zu verwandten Themen (sogenannten *Navileisten*, [WP:NAVI]), als Markierung für Probleme mit einer Seite, als Formatierungshilfe im laufenden Text sowie für diverse andere Zwecke [WP:TB]. Unter anderem werden Vorlagen auch verwendet, um besondere Arten von Seiten zu kennzeichnen, insbesondere zum Beispiel *Begriffsklärungsseiten*. Sie sind daher besonders nützlich, um solche besonderen Seiten zu erkennen.

Vorlagen sind Gegenstand diverser Arbeiten über die Wikipedia, insbesondere können Infoboxen leicht für eine *Wissensextraktion* verwendet werden, die Eigenschaft-Wert-Paare für die Eigenschaften des Artikelgegenstandes liefert (z. B. bei Auer u. a. (2007), siehe auch ▹2.5). In der vorliegenden Arbeit werden Vorlagen jedoch fast ausschließlich zur Identifikation von Ressourcen- und Konzepttypen verwendet (siehe ▹8.5 bzw. ▹8.6 sowie ▹8.4).

Magic Words: MediaWiki verwendet zwei Arten von *Magic Words*: „Variablen“, die Zugriff auf kontext- oder projektspezifische Werte bieten und „Optionen“, die die Verarbeitung der Seite beeinflussen (siehe ▹A.6).

Die meisten *Magic Words* werden für den Zweck dieser Arbeit ignoriert; manche Variablen aber, insbesondere eben der Seitenname, werden vor der Bearbeitung durch den betreffenden konkreten Wert ersetzt (siehe ▹8.3). Einige weitere werden ähnlich wie Vorlagen interpretiert, um zusätzliche Informationen zu gewinnen — Beispiele hierfür sind `{{DISPLAYTITLE:...}}` und `{{DEFAULTSORT:...}}` (vergleiche ▹8.4). Ein weiteres Beispiel für *Magic Words*, die von WikiWord verwendet werden, sind die Markierungen für Weiterleitungsseiten, siehe ▹A.6.

***Extension-Tags* und *Parser-Functions*:** MediaWiki bietet zwei Möglichkeiten für die Erweiterung des Wiki-Text-Markups an: *Extension-Tags* verwenden Tags der Form

`<mytag>...</mytag>`, *Parser-Functions* verwenden vorlagenartige Strukturen der Form `{{#mytag:argument}}` (siehe auch ▹A.7). Beide Formen werden bei der Analyse von Wiki-Text im Rahmen dieser Arbeit vollständig ignoriert (siehe ▹8.3).

Begriffsklärungen: *Begriffsklärungsseiten* (*Disambiguierungen*) sind eine wichtige Konvention in Wikipedia-Projekten: Gibt es mehrere unterschiedliche Bedeutungen zu einem Term, so wird für diesen Term eine Begriffsklärungsseite angelegt, die alle Bedeutungen des Terms aufzählt [WP:BKL]. Dazu wird in der Regel eine Listenstruktur verwendet, die in jeder Zeile eine Glosse für eine der Bedeutungen hat und auf den entsprechenden Artikel verweist. Wie schon oben erwähnt, wird die Begriffsklärungsseite entweder direkt mit dem fraglichen Term als Name angelegt oder mit dem Zusatz *(Begriffsklärung)* — in letzterem Fall enthält per Konvention die Seite ohne Namenszusatz (die Seite zur *Hauptbedeutung* des Terms) einen Verweis auf die Begriffsklärungsseite.

Wie auch Weiterleitungen sind Begriffsklärungen im Kontext dieser Arbeit eine wichtige Quelle für die *Bedeutungsrelation*, also um alle Bedeutungen eines Terms und umgekehrt alle Bezeichnungen für ein Konzept zu finden (siehe ▹8.10).

1.3. Thesauri

Ein Thesaurus ist traditionell eine Sammlung von Termen (*Wörtern*), ihrer Beziehungen und Definitionen. Die verschiedenen Beziehungen dienen insbesondere dazu, den passendsten Term für einen Gegenstand, das heißt, die beste Bezeichnung für ein Konzept zu finden. Ein Thesaurus ist daher besonders als Struktur für ein *kontrolliertes Vokabular* geeignet, das zur Indexierung von Dokumenten verwendet werden soll.

Thesauri definieren in der Regel zumindest eine Äquivalenzbeziehung zwischen synonymen und pseudo-synonymen (also für den Zweck der Indexierung gleichwertigen) Termen, eine (poly-)hierarchische Subsumtionsbeziehung zwischen allgemeinen und speziellen Termen sowie häufig eine Assoziationsbeziehung zwischen semantisch verwandten Termen [Stock (2007), s. 283ff]. Die ISO definiert insbesondere die folgenden Relationen [ISO:2788 (1986)]:

UF bzw. USE: *Used For* bzw. *Use* gibt an, dass ein Term anstelle eines anderen verwendet werden soll — über diese Relation wird der Deskriptor (*Preferred Term*) aus einer Menge von (Pseudo-)Synonymen (einem sogenannten *SynSet*) festgelegt.

BT bzw. NT: *Broader Term* bzw. *Narrower Term* gibt an, dass ein Term allgemeiner bzw. spezieller als ein anderer, das heißt, ein *Hyperonym* bzw. *Hyponym* zu diesem ist.

RT: *Related Term* gibt an, dass ein Term mit einem anderen semantisch verwandt ist. Diese Beziehung besteht häufig unter anderem zwischen Kohyponymen.

Synonyme können so direkt modelliert werden, Homonyme dagegen nicht: häufig werden sie *disambiguiert*, indem ein *Qualifikator* als Zusatz angehängt wird, entweder direkt als Teil des Terms oder über eine spezielle Relation.

Die hierarchische Beziehung (Hyponymiebeziehung) in einem Thesaurus ist häufig als *Taxonomie* angelegt, also als eine strenge thematische Gliederung, über die Terme in Sachgebiete eingeordnet werden. Die Hyponymiebeziehung kann auch als Polyhierarchie angelegt sein, so dass sich insgesamt ein gerichteter, azyklischer Graph als Struktur bildet, anstelle eines einfachen Baums. Die Semantik der Hyponymiebeziehung ist in der Regel nicht weiter spezifiziert, meist handelt es sich um eine Vermengung der *Abstraktionsbeziehung* (`subtype`), der *Instanzbeziehung* (`is-a`), der *Aggregationsbeziehung* (`part-of`) und gelegentlich der *Attributbeziehung* (`property-of`) [Voss (2006), Ponzetto (2007)].

Im Kontext der automatischen Verarbeitung von Wissen, Sprachen und Dokumenten erweist es sich als hilfreich, eine weitere Ebene der Indirektion einzuführen und so zu einem *„konzeptorientierten Thesaurus“* zu gelangen, der auf einem Konzept-Term-Netzwerk basiert statt auf einem reinen Netzwerk von Termen [Johnston u. a. (1998), Matthews (2003), W3C (2005)]. So ergibt sich ein *Wissensorganisationssystem* (*Knowledge Organisation System*, KOS). Dieses Modell scheint insbesondere auch besser auf die Datenstruktur der Wikipedia zu passen: Es werden die semantischen Beziehungen zwischen Konzepten direkt modelliert und zusätzlich angegeben, welche Terme zu welchem Konzept gehören (vergleiche [van Assem u. a. (2006), Gregorowicz und Kramer (2006)]). Die Beziehungen zwischen Termen (Homonym, Synonym) ergeben sich dann implizit (▹4.2). Ein ähnliches Modell wird auch von WordNet verwendet, in dem Konzepte durch SynSets repräsentiert und semantische Beziehungen zwischen SynSets statt zwischen Termen modelliert werden [Miller (1995), Fellbaum (1998), WordNet].

In einem solchen konzeptorientierten Thesaurus gibt es nun vor allem die folgenden Beziehungen:

Bedeutung: Verbindet einen Term mit einer Anzahl von Konzepten (Bedeutungen), bzw. ein Konzept mit einer Anzahl von Termen (Bezeichnungen). Daraus ergibt sich implizit die Synonymie oder Homonymie von Termen.

Subsumtion: Verbindet allgemeinere mit spezielleren Konzepten, analog zu der Hyponymierelation zwischen Termen.

Verwandtheit: Verbindet verwandte Konzepte, z. B. solche, die zu einem gemeinsamen Themengebiet gehören oder häufig im selben Kontext verwendet werden.

Ähnlichkeit: Verbindet ähnliche Konzepte, z. B. solche, die zu einem gewissen Grade austauschbar sind. Ähnliche Konzepte sind häufig Kohyponyme. Für eine Abgrenzung von semantischer Verwandtheit und Ähnlichkeit siehe [Zesch u. a. (2007b)].

Die semantischen Beziehungen in einem solchen Thesaurus sind aber weiterhin relativ „schwach“: Wie schon oben für die Hyponymierelation, ist auch hier im Gegensatz zu den Relationen in einer *Ontologie* nichts über die Art der Subsumtion (Subtyp, Instanz, Teil etc.) oder der Verwandtheit bekannt [Ponzetto und Strube (2007b), Hepp u. a. (2006)]. Auch können Relationen nicht reifiziert oder klassifiziert werden. Solche schwachen Beziehungen sind aber für die Zwecke des Information Retrieval in aller Regel ausreichend.

Ziel dieser Diplomarbeit ist es nun, einen solchen konzeptorientierten Thesaurus auf Grundlage des Datenbestandes der Wikipedia vollautomatisch zu erstellen.

1.4. Nutzen

Der Nutzen von Thesauri (insbesondere der konzeptorientierten Art) für die automatische Sprachverarbeitung und für das Information Retrieval liegt in der Überbrückung der *Semantic Gap* zwischen dem bloßen *Text* als Folge von Zeichen und dem konzeptuellen *Inhalt* von Dokumenten [Davulcu u. a. (2006), Gregorowicz und Kramer (2006)]. Handelt es sich um einen multilingualen Thesaurus, kann auf diese Weise auch die Kluft zwischen verschiedenen Sprachen überwunden werden: Verschiedene Terme (Wörter und Phrasen) aus verschiedenen Sprachen werden auf eine relativ kleine Menge wohldefinierter Konzepte abgebildet, die untereinander in Beziehung stehen. Die in einem Dokument behandelten Konzepte werden so einer automatischen Analyse zugänglich.

Konkret kann ein maschinenlesbarer, multilingualer, konzeptorientierter Thesaurus insbesondere in den folgenden Bereichen nützlich sein:

1. Für das klassische Information Retrieval ist es vorteilhaft, statt Termen abstrakte Konzepte für die Indexierung von Dokumenten verwenden zu können: so lassen sich Unterschiede in Sprache und Terminologie leicht überbrücken. Häufig werden Konzepte als Mengen von Termen modelliert, die durch Clustering-Verfahren gewonnen wurden, wie beim *Latent Semantic Indexing* (LSI) [Deerwester u. a. (1990), Kumar u. a. (2006)]. Die Verwendung eines konzeptorientierten Thesaurus erlaubt es dagegen, menschliches Wissen über die Zuordnung von Termen zu Konzepten zu nutzen (siehe ▹2.4 und insbesondere [Milne und Nichols (2007)]). Neben der klassischen Dokumentensuche profitieren auch Aufgaben wie *Resource Selection* und *Topic Tracking* von diesem Ansatz.

2. Ein Thesaurus auf Basis der Wikipedia eignet sich besonders für die *Named Entity Recognition* (*NER*), da Wikipedia im Gegensatz zu klassischen Wörterbüchern eine große Zahl von Einträgen zu Orten, Personen und Organisationen hat (siehe ▹2.6).

3. Die Informationen über ähnliche, verwandte und übergeordnete Konzepte, die in einem Thesaurus enthalten sind, können unter anderem für eine gezielte *Query Expansion* verwendet werden [Milne und Nichols (2007)]. Es ist zu erwarten, dass dies besonders dann effektiv ist, wenn die Indexierung und Abfrage anhand von Konzepten statt von Termen erfolgt, da so Mehrdeutigkeiten vermieden werden können [Syed u. a. (2008)].

4. Die Verwandtheitsbeziehungen zwischen Konzepten können auch verwendet werden, um Terme im Kontext zu *disambiguieren*: Für eine Menge von Termen kann die Bedeutung jedes Terms so gewählt werden, dass die Bedeutungen (Konzepte) gut zueinander passen — auf diese Weise lässt sich mit großer Wahrscheinlichkeit die intendierte Bedeutung der Terme finden [Mihalcea (2007), Strube und Ponzetto (2006), Cucerzan (2007)], siehe ▹2.3. Diese Bedeutungszuweisung ist neben der Anwendung auf Suchanfragen auch wichtig bei der *Wissensextraktion*, dem *Topic Tracking* sowie der *automatischen Übersetzung*.

5. Ein multilingualer Thesaurus kann auch als Basis für ein Übersetzungswörterbuch oder gar ein automatisches Übersetzungssystem dienen. Die aus der Wikipedia gewonnenen Informationen bieten hier eine nützliche Ergänzung zu traditionellen Wörterbüchern, da sie

eine große Zahl von Fachbegriffen sowie viele Namen für Orte und Personen abdecken, die in manuell gepflegten Wörterbüchern in der Regel fehlen.

6. Ein aus der Wikipedia gewonnener multilingualer Thesaurus kann auch als Basis für eine (teil-)manuelle Erstellung von Wörterbüchern, Taxonomien und Ontologien dienen. Für Ontologien und Wissensrepräsentationssysteme wie *DBpedia* oder *YAGO* kann es auch möglich sein, Informationen aus der Wikipedia automatisch zu integrieren (siehe ▸2.2 und ▸2.5).

7. Viele der weiter unten in ▸2 beschriebenen Systeme und Verfahren verwenden als Grundlage eine aus der Wikipedia extrahierte Struktur, die deutlich weniger Informationen berücksichtigt als WikiWord. Sie könnten vermutlich davon profitieren, WikiWord als Werkzeug für die Extraktion der Wikipedia-Struktur zu verwenden, um dann die betreffenden Verfahren auf diese reichere Struktur anzuwenden.

Ein wichtiger Aspekt des in der vorliegenden Diplomarbeit vorgeschlagenen Ansatzes ist es, dass der Aufbau des Thesaurus aus den Wikipedia-Daten vollautomatisch und verhältnismäßig schnell[2] erfolgt: Die manuelle Wartung eines Thesaurus ist sehr aufwändig und teuer, um so mehr, je mehr Fachgebiete der Thesaurus abdecken soll. So ist es kaum möglich, einen Thesaurus auf dem neusten Stand zu halten, insbesondere in Hinblick auf neue Technologien oder auch aktuell wichtige Organisationen und Personen. Schon deshalb decken die meisten Thesauri nur einen eng eingegrenzten Fachbereich ab und enthalten nur relativ wenige Terme bzw. Konzepte: *WordNet* zum Beispiel enthält etwa 117 000 Konzepte (*SynSets*) [WordNet], *MeSH* kennt nur 24 767 Konzepte [MeSH] (siehe auch ▸2.2).

Ein Thesaurus, der automatisch aus Wikipedia-Daten erstellt wird, hat diese Probleme nicht: Wikipedia ist sehr aktuell und deckt sehr weite Bereiche des menschlichen Wissens ab [Ruiz-Casado u. a. (2005), Voss (2005b)] (WikiWord liefert etwa 6 Millionen Konzepte und 12 Millionen Terme für die englische Sprache, vergleiche ▸11.2). Insbesondere sind genau solche Themenbereiche abgedeckt, die im Augenblick besonders im öffentlichen Bewusstsein stehen und daher besonders interessant sind für das Information Retrieval (vergleiche [Syed u. a. (2008), Hepp u. a. (2006), Milne und Nichols (2007)]). Zudem stehen die Daten unter einer freien Lizenz und sind in einer Vielzahl von Sprachen verfügbar.

Ein multilingualer, konzeptorientierter Thesaurus auf Wikipedia-Basis besitzt also das Potential, für wichtige Aufgaben des Information Retrieval und der automatischen Sprachverarbeitung eine breite, aktuelle und qualitativ hochwertige Datenbasis zu liefern. Außerdem kann er verwendet werden, um existierende, manuell gewartete Thesauri zu ergänzen und zu erweitern.

[2] Einige Tage bis Wochen auf einem üblichen Datenbanksystem, siehe ▸11.1.

2. Einordnung

Im Folgenden wird ein Überblick über den Stand der Forschung in Bereichen gegeben, die in Bezug auf den Gegenstand der vorliegenden Arbeit besonders relevant sind. Insbesondere werden Arbeiten betrachtet, die entweder die Wikipedia selbst zum Gegenstand haben oder Inhalte und Struktur der Wikipedia nutzen, um Aufgaben aus dem Bereich der automatischen Sprachverarbeitung und des Information Retrieval zu bewältigen.

2.1. Untersuchung von Struktur und Qualität der Wikipedia

Die Wikipedia findet seit dem Jahr 2005 zunehmend Beachtung in der Forschung, sowohl als Gegenstand von Untersuchungen als auch als Datenbasis oder Korpus für Verfahren zur Bewältigung einer Vielzahl von unterschiedlichen Aufgaben.

Im Bereich der Arbeiten, die allgemeine Eigenschaften der Wikipedia untersuchen, wie zum Beispiel das Wachstum oder die Verknüpfungsstruktur, sind vor allem die Arbeiten von *J. Voß* zu nennen [Voss (2005A), Voss (2005B)] sowie ein Vortrag von *J. Wales*, dem Gründer[1] der Wikipedia [Wales (2004)]. *A. Capocci* et. Al. untersuchen die Eigenschaften der Wikipedia insbesondere in Hinblick auf das Modell des *Preferential Attachment* [Capocci u. a. (2006)], während *F. Bellomi* und *R. Bonato* gängige Algorithmen wie *HITS* und *PageRank* zur Bestimmung der *Relevanz* bzw. *Autorität* auf die Wikipedia anwenden [Bellomi und Bonato (2005B), Bellomi und Bonato (2005A)].

Gegen die Verwendung der Wikipedia als Datenbasis für Verfahren der Wissens- und Sprachverarbeitung gab es lange Zeit (und gibt es zum Teil noch immer) den Vorbehalt, dass die Qualität der Inhalte zu schlecht oder zumindest zu unzuverlässig sei, um sie bedenkenlos verwenden zu können. Anlass für diese kritische Haltung war und ist zumeist der Umstand, dass die Wikipedia jederzeit von jedermann frei bearbeitet werden kann und Vorkehrungen zur Qualitätssicherung wie in einem traditionellen Redaktionssystem weitgehend fehlen. Dem wurde von Seiten der Wikimedia Foundation sowie der Nutzer der Wikipedia entgegengehalten, dass neben dem Einbringen neuer Informationen eben auch die redaktionellen Aufgaben wie Qualitätssicherung von jedermann übernommen werden können und auch tatsächlich übernommen werden [Wales (2004)]. Seit dem Jahr 2006 ist es erklärtes Ziel der Wikipedia, die Qualität und Verlässlichkeit der Artikel zu verbessern [WP:100K]. In der Wikipedia werden seit ihrer Gründung, aber besonders seit 2006, immer neue Strukturen und Verfahren entwickelt, um die Qualität der Artikel zu verbessern. Beispiele aus der deutschsprachigen Wikipedia sind das Qualitätssicherungsprojekt [WP:QS], die Auszeichnung von *exzellenten Artikeln* [WP:EA] sowie, als Experiment seit Mai 2008, das Verfahren der *gesichteten Versionen* [WP:GS]: Änderungen, die von anonymen oder neuen Benutzern vorgenommen werden, werden nicht mehr sofort, sondern erst nach einer ersten (oberflächlichen) Prüfung der Allgemeinheit präsentiert.

[1] Oder Mitbegründer. Ob die Erfindung der Wikipedia alleine *Jimmy Wales* zuzuschreiben ist, oder inwieweit die Idee auf *Larry Sanger* zurückgeht, ist Gegenstand eines andauernden Streits zwischen eben jenen.

Die erste und wohl bekannteste Studie zur Qualität der Inhalte der Wikipedia wurde von *J. Giles* im Jahr 2005 für die Zeitschrift *Nature* durchgeführt [Giles (2005)]. *Giles* vergleicht hier eine Anzahl von Artikeln aus der Wikipedia mit den entsprechenden Einträgen in der *Encyclopaedia Britannica* und kommt zu dem Ergebnis, dass die Wikipedia dem traditionellen Nachschlagewerk in Bezug auf die Qualität kaum unterlegen und in Bezug auf die Ausführlichkeit der Beschreibungen und die Abdeckung von aktuellen Themen gar überlegen ist. Diese Untersuchung hatte zur Folge, dass die Wikipedia zunehmend als ernstzunehmende und verlässliche Wissensbasis betrachtet und untersucht wurde.

Detailliertere und differenziertere Untersuchungen zur Qualität der Wikipedia wurden vor allem von *R. Hammwöhner* durchgeführt [Hammwöhner (2007a), Hammwöhner (2007b), Hammwöhner (2007)]. Hierbei wurde insbesondere deutlich, dass die Qualität einzelner Artikel stark divergiert: es gibt in Bezug auf die untersuchten Eigenschaften sowohl Beispiele sehr geringer, wie auch sehr hoher Qualität. Diese Untersuchungen beruhen zum Teil auf den Kriterien, die von *Stvilia* et. Al. für die Evaluation von Enzyklopädien entwickelt wurden [Stvilia u. a. (2005)]. Probleme sieht *Hammwöhner* besonders im Bereich der Wissensorganisation zum Beispiel in Form des Kategoriesystems, welches insbesondere für die englischsprachige Wikipedia als schlecht strukturiert und zum Teil fehlerhaft befunden wurde. Er schlägt vor, die Kategoriestrukturen aus mehreren Wikipedias über die Language-Links, die sie verbinden, zu vergleichen und auf diese Weise Probleme aufzudecken und zu beseitigen [Hammwöhner (2007b)].

2.2. Maschinenlesbare Wissensbasen

Es existieren bereits einige Versuche, eine maschinenlesbare Wissensbasis zur Verfügung zu stellen, die als Grundlage für ein weites Feld an Verfahren aus den Bereichen der automatischen Sprachverarbeitung, des Information Retrieval, aber auch der Wissensrepräsentation und künstlichen Intelligenz dienen können. Zwei der ältesten und bekanntesten Systeme dieser Art sind *Cyc* und *WordNet*:

Das Cyc-Projekt bietet eine manuell gepflegte, logikbasierte Ontologie mit etwa 300 000 Konzepten und 3 000 000 Verknüpfungen bzw. Aussagen über diese Konzepte (47 000 Konzepte und 300 000 Aussagen in der Open-Source-Version *OpenCyc*) [Lenat u. a. (1990), CYC, OpenCYC]. Es wird seit 1984 unter der Leitung von *D. Lenat* entwickelt und hat seine Wurzeln in der klassischen KI-Forschung. Ziel des Cyc-Projektes ist es, Softwaresystemen Zugang zu „gesundem Menschenverstand“ (*„Common Sense“*), also *Weltwissen*, zu bieten. Neben der Ontologie enthält es auch ein Lexikon, das (englische) Terme auf Konzepte der Ontologie abbildet.

WordNet ist ein von Hand erzeugtes semantisches Lexikon der englischen Sprache, es besteht aus etwa 117 000 *SynSets* (Konzepten) mit etwa 155 000 Wörtern [Miller (1995), Fellbaum (1998), WordNet]. Zwischen den Konzepten sind die für Thesauri üblichen Relationen definiert, wie *allgemeiner/spezieller* und *verwandt* sowie einige speziellere, wie *Aggregationsbeziehung* (*part-of*). WordNet stammt aus dem Bereich der automatischen Sprachverarbeitung, es wird seit 1985 von *G. A. Miller* und in jüngerer Zeit von *C. Fellbaum* betreut.

Eine neuere Entwicklung stellt SUMO (*Suggested Upper Merged Ontology*) dar, das eine gemeinsame standardisierte „obere“ Ontologie für verschiedene Anwendungen im Bereich Suche,

automatisches Schließen sowie automatische Sprachverarbeitung bieten soll [Niles und Pease (2001), SUMO, IEEE:SUO-WG]. SUMO wird seit dem Jahr 2000 vom IEEE entwickelt und enthält Anfang 2008 etwa 20 000 handverlesene Konzepte und 70 000 Aussagen (Axiome). Für SUMO gibt es eine vollständige Abbildung auf WordNet, mit deren Hilfe Wörter der englischen Sprache an die abstrakten Konzepte der Ontologie gebunden werden können.

Lange vor solchen maschinenlesbaren Wissensbasen, die auf die Verwendung im Bereich der automatischen Sprachverarbeitung und künstlichen Intelligenz ausgerichtet sind, gab es Klassifikationssysteme bzw. kontrollierte Vokabulare für die Indexierung von Dokumenten, insbesondere für die Verwendung in Bibliotheken. Einige prominente Beispiele sind *DDC* (*Dewey Decimal Classification* [Dewey (1965), DDC]) und die komplexere *UDC* (*Universal Decimal Classification* [Otlet und Fontaine (1905), UDC]) sowie die *LLC* (*Library of Congress Classification* [LCC, Chan (1986)]). In Deutschland ist außerdem die *ASB* (*Allgemeine Systematik für Öffentliche Bibliotheken* [VBNW (2003)]) verbreitet.

Neben diesen breit angelegten Systemen gibt es eine Vielzahl von fachspezifischen Thesauri, die für die Indexierung von Dokumenten in bestimmten Fachbereichen und/oder für bestimmte Institutionen entwickelt wurden. Bekannte Beispiele dafür sind unter anderem *MeSH* (die *Medical Subject Headings* der *United States National Library of Medicine* [Lipscomb (2000), MeSH], das von *MEDLINE* und *PubMed* für Arbeiten im Bereich Medizin und Biologie verwendet wird) und die *CR Classification* (das *ACM Computing Classification System* für die Klassifikation von Arbeiten aus dem Bereich Informatik und Informationstechnologie [Coulter u. a. (1998), CCS]).

Wissensbasen, die auf Informationen aus der Wikipedia beruhen, sind in den letzten Jahren ebenfalls entstanden. Zu nennen sind hier einerseits *YAGO*, das WordNet als ein strukturelles Rückgrat verwendet, um die Inhalte der Wikipedia zu organisieren [YAGO]. Andererseits ist *DBpedia* zu erwähnen, welches strukturierte Daten wie Attribut-Wert-Paare aus der Wikipedia extrahiert und in einer RDF-Datenbank ablegt [DBpedia]. Diese beiden Datenbasen, sowie die Systeme, mit deren Hilfe sie erstellt wurden, werden in ▸2.4 bzw. ▸2.5 noch etwas genauer betrachtet.

Ein weiteres derartiges Projekt ist *Freebase*, dessen Ziel es ist, eine frei zugängliche Datenbank zu schaffen, in dem das Wissen der Welt verknüpft ist [Freebase]. Die Idee ist es, Daten aus unterschiedlichen Quellen miteinander zu kombinieren und über eine einheitliche Schnittstelle zugänglich zu machen. Freebase enthält Daten unter anderem aus der Wikipedia, aus *PubMed*[2], dem *CIA World Factbook*[3] und diversen anderen Quellen.

Ziel von WikiWord ist es nun, Daten zu liefern, die solche Systeme ergänzen und bereichern können. Insbesondere die Möglichkeit, Terme aus unterschiedlichen Sprachen auf ein gemeinsames abstraktes Konzept abzubilden, sowie eine Vielzahl von Bezeichnungen für dasselbe Konzept in jeder Sprache zu unterstützen, soll einen Beitrag dazu leisten, die Inhalte solcher Wissensdatenbanken auf natürlichsprachliche Texte wie Dokumente oder Suchanfragen anwenden zu können.

[2] <http://www.pubmedcentral.nih.gov/>
[3] <https://www.cia.gov/library/publications/the-world-factbook/index.html/>

2.3. Automatische Disambiguierung und semantische Verwandtheit

Zwei eng miteinander verwandte Aufgaben aus dem Bereich der automatischen Sprachverarbeitung und des Information Retrieval, die beide ein gewisses Maß an *Weltwissen* benötigen, sind die automatische *Disambiguierung* (*Sinnzuweisung*, *Bedeutungszuweisung*) von Termen sowie die Bestimmung des Grades der semantischen Verwandtheit und Ähnlichkeit von Termen bzw. Konzepten. Die Datenbasis, die für diese Aufgaben benötigt wird, ist ein Assoziationsnetz von Termen und/oder Konzepten, wie es durch einen Thesaurus gegeben ist.

Die Wikipedia erfreut sich in jüngster Zeit wachsender Beliebtheit als Grundlage für die automatische Erstellung einer solchen Datenbasis, da sie ein weites Feld von Themen abdeckt und dabei sehr aktuell ist. Zu nennen sind hier zunächst die Arbeiten von *M. Strube* und *S. P. Ponzetto*, die Verfahren zur Extraktion von Thesauri und Taxonomien aus der Wikipedia entwickelt [Ponzetto (2007), Ponzetto und Strube (2007b)] und verschiedene Methoden zur Berechnung der semantischen Verwandtheit zwischen Konzepten evaluiert haben [Ponzetto und Strube (2007c), Strube und Ponzetto (2006), Ponzetto und Strube (2007a)]. Hierbei wurden vor allem die Kategoriestruktur der Wikipedia sowie die Querverweise zwischen Artikeln verwendet.

R. Mihalcea dagegen betrachtet vor allem die Beziehung zwischen Link-Text und Link-Ziel und interpretiert diese als Bedeutungsannotation (*Sense-Tags*) [Mihalcea (2007)]. Konkret verwendet er Wikipedia als Trainingsmenge für einen Klassifikator zur automatischen Bedeutungsannotation. *E. Gabrilovich* verwendet verschiedene Eigenschaften von Wikipedia-Artikeln, um einen Klassifikator zu trainieren, der Dokumente kategorisiert [Gabrilovich (2006), Gabrilovich und Markovitch (2006), Gabrilovich und Markovitch (2007)]. Ebenfalls mit der Klassifikation von Dokumenten mit Hilfe der Informationen aus der Wikipedia befasst sich *P. Schönhofen* [Schönhofen (2006)]. *A. Krizhanovsky* verwendet die Struktur von Wikipedia, um ähnliche Seiten zu gruppieren und so Synonymgruppen zu finden [Krizhanovsky (2006)].

Die Forschung von *T. Zesch* und *I. Gurevych* gilt besonders der Berechnung von semantischer Verwandtheit und der Evaluation von Wikipedia als Datenbasis für diesen Zweck, im Vergleich unter anderem zu WordNet [Zesch u. a. (2007a), Zesch u. a. (2007b), Zesch und Gurevych (2007)], besonders unter Verwendung der Kategoriestruktur. Auf diese Arbeiten wird insbesondere bei der Evaluation in Kapitel ▹14 Bezug genommen.

K. Nakayama et. Al. untersuchen neben Maßen für die semantische Verwandtheit auch Maße für die Relevanz und Zentralität von Konzepten, insbesondere auf Basis der Struktur der Querverweise über Wiki-Links [Nakayama u. a. (2007b), Nakayama u. a. (2007a)]. Dabei untersuchen sie unter anderem, inwieweit sich eine Beschränkung der Suchtiefe bei der Analyse der Verknüpfungsstruktur auf die Ergebnisse auswirkt.

Die Evaluation all dieser Ansätze zeigt jeweils, dass Wikipedia als Datenbasis für solche Aufgaben beachtliche Ergebnisse liefert, in der Präzision häufig an manuell gewartete Wissensnetze wie WordNet heran reicht und diesen in Bezug auf die Abdeckung von Spezialausdrücken oder aktuellen Themen sogar überlegen ist.

In diesem Zusammenhang ebenfalls erwähnenswert sind die Arbeiten von *M. Jarmasz* und *S. Szpakowicz*, die *Roget's Thesaurus* auf seinen Nutzen für die Bestimmung semantischer

Verwandtheit untersucht haben [Jarmasz und Szpakowicz (2001), Jarmasz und Szpakowicz (2003)]. Dabei haben sie insbesondere den Wert einer reichen Taxonomie für diese Aufgabe herausgestellt.

2.4. Erzeugung von Thesauri und Taxonomien

Die Idee, Wikipedia als lexikalisch-semantische Datenbasis zur Erstellung von Thesauri und Taxonomien zu verwenden, ist schon verschiedentlich Gegenstand der Forschung gewesen. Auch die oben beschriebenen Systeme zur Untersuchung von semantischer Verwandtheit verwenden alle ein Thesaurus-artiges Assoziationsnetzwerk, insbesondere die Arbeiten von *M. Strube* und *S. P. Ponzetto*, die von *E. Gabrilovich*, von *T. Zesch* und *I. Gurevych* sowie die von *K. Nakayama* et. Al. Im Folgenden werden einige Untersuchungen aufgeführt, die den Aufbau eines Thesaurus oder einer Taxonomie unmittelbar zum Ziel haben.

A. Gregorowicz und *M. A. Kramer* bauen auf Grundlage der Wikipedia ein Term-Konzept-Netzwerk auf, das Relationen zwischen Konzepten sowie die Zuordnungen von Termen zu Konzepten enthält, sehr ähnlich der Struktur, die auch in der vorliegenden Diplomarbeit verwendet wird [Gregorowicz und Kramer (2006)]. *D. Milne* et. Al. untersuchen eine ähnliche Struktur insbesondere in Hinblick auf eine Verwendung für die Indexierung von Dokumenten und das Information Retrieval [Milne u. a. (2006), Milne und Nichols (2007), Milne u. a. (2007)]. Sie entwickeln eine wissensbasierte Suchmaschine namens *Koru*, um die Eignung der aus der Wikipedia gewonnenen Daten für Indexierung und Suche zu demonstrieren.

J. Voß untersucht, wie sich das Kategoriesystem der Wikipedia zu anderen Indexierungs- und Klassifikationssystemen verhält [Voss (2006)]. *M. Ruiz-Casado* et. Al. ordnen Wikipedia-Einträge automatisch WordNet-SynSets zu und evaluieren die relative Abdeckung und Granularität dieser beiden Wissensbasen [Ruiz-Casado u. a. (2005)]. *F. M. Suchanek* et. Al. kombinieren ebenfalls Wikipedia und WordNet, allerdings mit dem Ziel, Wikipedia-Einträge als *Individuen* bzw. *Instanzen* von *Klassen* zu verwenden, die von WordNet-Einträgen gebildet werden [Suchanek u. a. (2007)]. Auf diese Weise entsteht eine neue Wissensstruktur, *YAGO*, die die Wohlstrukturiertheit von WordNet mit der Informationsfülle und Aktualität der Wikipedia verbinden soll.

Z. Syed et. Al. verwenden die Einträge der Wikipedia als Vokabular zur Indexierung von Dokumenten. Die durch Querverweise zwischen Artikeln gegebene Assoziation von Themen wird zur Bestimmung relevanter Konzepte zu einer Suchanfrage genutzt, indem das Verfahren der *Spearding Activation* auf die Verknüpfungsstruktur angewendet wird [Syed u. a. (2008)]. *M. Hepp* et. Al. untersuchen, inwieweit sich Wikipedia-Einträge bzw. ihre URLs als *Bezeichner* für Konzepte zur Indexierung von Dokumenten eignen [Hepp u. a. (2006)]. Sie betrachten dabei unter anderem, wie sich das Konzept, das mit einer gegebenen URL bzw. URI verbunden ist, über die Zeit verändert.

L. Muchnik et. Al. beschreiben ein Modell für die Hierarchisierung von Netzwerken im Allgemeinen, und validieren es unter anderem am Beispiel der Wikipedia, indem sie allein auf Grund der Struktur der Querverweise eine Taxonomie der Artikel aufbauen und diese mit der Kategoriestruktur vergleichen [Muchnik u. a. (2007)]. Eines der Maße für die lokale Bestimmung der *Zentralität* von Konzepten, die dabei entwickelt wurden, nämlich der *Local Hierarchy Score* (*LHS*), wird auch im Rahmen dieser Arbeit verwendet (▹9.3).

E.M. van Mullige et. Al. entwickeln eine Erweiterung der MediaWiki-Software, mit deren Hilfe die komplexen, hoch strukturierten Daten eines multilingualen Wörterbuches und Thesaurus namens *OmegaWiki* (ehemals *WiktionaryZ*) kollaborativ in einem Wiki verwaltet und bearbeitet werden können [van Mulligen u. a. (2006), OmegaWiki]. Die MediaWiki-Erweiterung, *Wikidata*, auf der diese Anwendung basiert, wurde vornehmlich für den Einsatz in *OmegaWiki* entwickelt. Sie dient vor allem der Verwaltung von relationalen Datensätzen in einem Wiki. Dabei liegt der Fokus, anders als bei dem weiter unten erwähnten *Semantic MediaWiki*, darauf, eine komplexe aber fest vorgegebene Datenbankstruktur mit Inhalten zu füllen. Zu diesem Zweck werden spezialisierte Eingabehilfen bereitgestellt.

2.5. Wissensextraktion

Wikipedia kann auch als Grundlage für die Erzeugung einer Wissensbasis dienen, die über ein bloßes Assoziationsnetzwerk und einfache Subsumtionsbeziehungen hinausgeht: durch geeignete Verfahren lassen sich Attribut-Wert-Paare sowie semantisch bedeutsame Relationen zwischen Entitäten bzw. Konzepten extrahieren. In den letzten Jahren wurden verschiedene Versuche unternommen, diese Informationen zugänglich zu machen.

S. Auer, *J. Lehmann* et. Al. haben ein System namens *DBpedia* entwickelt, das Attributwerte sowie semantische Relationen aus der Wikipedia extrahiert, vor allem durch Auswertung spezieller Vorlagen, so genannter *Infoboxen* [Auer u. a. (2007), Auer und Lehmann (2007), DBpedia]. Diese bieten strukturierte Daten in einer Form an, die bereits weitgehend dem Modell von Attribut-Wert-Paaren entspricht [WP:IB]. *DBpedia* stellt die so gewonnenen Daten als eine Wissensbasis im RDF-Format [W3C:RDF (2004)] bereit und bietet eine Abfrageschnittstelle, die komplexe logikbasierte Abfragen auf der Basis von SPARQL [W3C:SPARQL (2008)] erlaubt.

Auch *D. P. T. Nguyen* et. Al. verwenden Infoboxen zur Gewinnung von strukturierten Daten sowie zur Klassifikation von Artikeln bzw. von Konzepten [Nguyen u. a. (2007)]. *F. Wu* und *D. S. Weld* benutzen ebenfalls die Informationen aus Infoboxen, trainieren aber ein System zur Erkennung von Textmustern darauf, Informationen aus dem Fließtext mit denen aus Infoboxen abzugleichen und gegebenenfalls Infoboxen zu ergänzen oder zu erzeugen [Wu und Weld (2007)].

Neben der Forschung zur Extraktion von Daten aus der Wikipedia gibt es auch Versuche, Wiki-Systeme so zu erweitern, dass sie unmittelbar zur Erzeugung und Pflege von strukturierten, semantisch bedeutsamen und maschinenlesbaren Daten verwendet werden können. Der wohl bekannteste Versuch ist wohl der von *M. Krötzsch* et. Al., welche mit *Semantic MediaWiki* eine Erweiterung von MediaWiki entwickelt haben, mit dessen Hilfe strukturierte Daten wie semantische Relationen und Attributwerte direkt in einem Wiki-System verwaltet und bearbeitet werden können [Krötzsch u. a. (2005), Krötzsch u. a. (2007), SMW]. Sie schlagen vor, dieses System auch für die Wikipedia selbst zu verwenden. *T. Redmann* und *H. Thomas* untersuchen, wie sich Topic Maps [Garshol (2004), ISO:13250 (2003)] mit Hilfe eines Wiki-Systems bearbeiten lassen, und wie sich umgekehrt die typischen Strukturen eines Wikis in Form einer Topic Map speichern lassen [Redmann und Thomas (2007)]. *R. Witte* und *T. Gitzinger* schlagen vor, Mustererkennung und automatische Sprachverarbeitung direkt in Wiki-Systeme zu integrieren, um so den Aufbau einer Wissensbasis aus Texten sowie

umgekehrt die Beschreibung von strukturierten Daten in Texten als interaktiven Prozess zu gestalten [Witte und Gitzinger (2007)].

2.6. Named Entity Recognition

Eine wichtige Aufgabe im Bereich der automatischen Sprachverarbeitung sowie des Information Retrieval ist die Erkennung und Disambiguierung von *Named Entities*, also vor allem von (Eigennamen von) Personen, Orten, Organisationen und ähnlichem. Wikipedia enthält, genau wie andere Enzyklopädien, eine große Zahl von Einträgen über eben solche Konzepte, und bietet sich daher als Datenbasis für eine Erkennung solcher *Named Entities* an.

S. Cucerzan und *W. Dakka* untersuchten Methoden zur Disambiguierung von Namen sowie zur Klassifikation von Dokumenten unter Verwendung von Verfahren des maschinellen Lernens, konkret eines *naiven Bayes-Klassifikators* sowie *Support Vector Machines* (SVM). Dabei wurden unter anderem Informationen aus den Seitentiteln, Weiterleitungen, Begriffsklärungsseiten sowie aus dem Link-Text von Wiki-Links verwendet [Dakka und Cucerzan (2008), Cucerzan (2007)]. Einen ähnlichen Ansatz verfolgen *R. Bunescu* und *M. Pasca*, die ebenfalls SVM auf der Basis von Wikipedia für die Erkennung von *Named Entities* trainieren [Bunescu und Pasca (2006)].

A. Toral und *R. Munoz* entwarfen ein System zum automatischen Aufbau eines Gazetteers aus der Wikipedia, indem sie ähnlich wie *F. M. Suchanek* Beschreibungen von *Instanzen* aus der Wikipedia mit *Klassen* aus WordNet kombinierten [Toral und Munoz (2007), Toral und Munoz (2006)]. *J. Kazama* und *K. Torisawa* präsentieren eine Methode zur Erkennung und Klassifikation von Wikipedia-Seiten über *Named Entities*, die auf der Analyse des ersten Satzes von Wikipedia-Artikeln beruht [Kazama und Torisawa (2007)].

2.7. Eigenschaften von WikiWord

Dieser Abschnitt gibt einen kurzen Überblick über die Eigenschaften der WikiWord-Software, die im Rahmen dieser Diplomarbeit entwickelt wurde. Dabei werden insbesondere solche Eigenschaften betrachtet, die WikiWord zu den oben erwähnten Arbeiten in Beziehung setzen bzw. von ihnen abgrenzen.

- WikiWord erzeugt einen konzeptorientierten Thesaurus, also ein Netzwerk von Konzepten, die durch Relationen wie Subsumtion, Verwandtheit und Ähnlichkeit verbunden sind, mit einer zusätzlichen Bedeutungsrelation, die Terme mit Konzepten assoziiert und als Lexikon dient (vergleiche [Gregorowicz und Kramer (2006)]). Die meisten der oben beschriebenen Systeme zum Aufbau eines Thesaurus aus der Wikipedia (zum Beispiel die Arbeiten von *T. Zesch* und *I. Gurevych*) übergehen die Bedeutungsrelation weitgehend und befassen sich kaum mit der Identifikation verschiedener Bezeichnungen für Konzepte. In der Regel erschöpft sich die Behandlung von Synonymien in der Auflösung von Weiterleitungen, wie sie in der Wikipedia definiert sind.

- WikiWord erzeugt einen multilingualen Thesaurus, indem es Language-Links verwendet, um Artikel über dasselbe Konzept in unterschiedlichen Wikipedias zu identifizieren und zu vereinigen. Eine solche Konsolidierung der Informationen aus unterschiedlichen Wikipedias zu einer sprachübergreifenden Wissensbasis wurde in den oben erwähnten Studien bislang nicht untersucht. Lediglich [Auer u. a. (2007)] verwendet einige Informationen aus unterschiedlichen Wikipedias, wobei die Grundstruktur an die englischsprachige Wikipedia gebunden bleibt.

- WikiWord verwendet neben Seitentiteln und Weiterleitungsseiten auch Begriffsklärungsseiten sowie insbesondere auch Link-Text, um die verschiedenen Bezeichnungen für ein Konzept zu ermitteln und zu gewichten. Link-Text wird in den oben erwähnten Arbeiten zur Erzeugung eines Thesaurus kaum erwähnt, lediglich die Forschung zur *Named Entity Recognition* scheint sich dieser Informationsquelle bisher bedient zu haben (z. B. *S. Cucerzan* und *W. Dakka*). Dadurch deckt WikiWord eine Vielzahl von Bezeichnungen ab, die anderweitig schwer zu erfassen sind, darunter viele Mehrwortphrasen, Abkürzungen, flektierte Formen sowie ungenaue oder synekdotische Verwendungen von Termen.

- WikiWord liefert zu vielen Konzepten den ersten Satz des entsprechenden Wikipedia-Artikels als Glosse in den gegebenen Sprachen. Die oben erwähnten Verfahren nutzen meist den ersten Absatz als Glosse, statt nur des ersten Satzes, was zu mehr „Rauschen“ führt. Zudem wird sie häufig lediglich zur Bestimmung der semantischen Verwandtheit oder ähnlichem herangezogen und nicht als Eigenschaft eines Konzeptes im Resultat wiedergegeben.

- WikiWord verzichtet auf eine Analyse natürlicher Sprache, nicht einmal Stammformreduktion (*Stemming*) oder Wortarterkennung (*POS-Tagging*) kommen zum Einsatz. Das verwendete einfache *Pattern Matching* beschränkt sich auf Muster, die bestimmte Markup-Konstrukte und Strukturmerkmale von Wikipedia-Seiten erkennen. Das einzige verwendete Verfahren, das entfernt auf Sprachverarbeitung beruht, ist das zur Erkennung von Satzenden: Dieses wird verwendet, um Glossen aus den Wikipedia-Artikeln zu extrahieren.

- WikiWord extrahiert keine Attribut-Wert-Paare oder andere strukturierte Daten aus Vorlagen wie Infoboxen oder ähnlichem. Während Wikipedia eine reiche Quelle für diese Art von Informationen ist, soll die Auswertung dieser Daten Projekten wie DBpedia überlassen bleiben (vergleiche ▹2.5).

- WikiWord verwendet kein maschinelles Lernen und kaum statistische Heuristiken[4]. Stattdessen werden Strukturen und Konventionen, die in den verschiedenen Wikipedia-Projekten anzutreffen sind, explizit modelliert und ausgewertet. So kann die Investition menschlichen Urteils, die in der Struktur der Wikipedia enthalten ist, direkt genutzt werden.

- WikiWord berücksichtigt Konzepte, die nur als *Abschnitte* von Artikeln repräsentiert sind, sowie Konzepte, die zwar referenziert werden, jedoch (noch) keinen eigenen Artikel haben.

[4] eine erwähnenswerte Heuristik mag das Cutoff-Verfahren sein, das in ▹10.4 beschrieben ist.

Damit wird eine deutlich größere Zahl von Konzepten berücksichtigt, wobei allerdings zu vielen davon kaum Informationen vorliegen. Solche „schwachen“ Konzepte werden von WikiWord gekennzeichnet und können daher bei Bedarf ignoriert werden.

- WikiWord liefert zusätzlich zu der Subsumtionsrelation eine grobe Typisierung der Konzepte, die insbesondere verschiedene Arten von *Named Entities* identifizieren, zum Beispiel Personen, Orte, Zeitpunkte und Lebensformen. Diese Typen können zur Filterung der Konzepte je nach Anwendungsgebiet verwendet werden.
- WikiWord konsolidiert Konzepte aus Artikeln mit Konzepten, die in der Wikipedia durch Kategorien repräsentiert sind.

Insgesamt versucht WikiWord, möglichst viele lexikalisch-semantische Informationen, die in der Wikipedia kodiert sind, zugänglich zu machen. Dabei berührt es viele Bereiche, die von den oben beschriebenen Arbeiten über die Erzeugung von Taxonomien und Thesauri aus der Wikipedia behandelt werden, sowie auch von Arbeiten, die sich mit *Named Entity Recognition* befassen. WikiWord verzichtet auf heuristische und statistische Verfahren und beschränkt sich weitgehend auf die bloße Extraktion und Repräsentation der Daten. Die meisten der oben erwähnten Verfahren könnten auf die von WikiWord extrahierten Daten angewendet werden, statt die Daten der Wikipedia direkt zu benutzen — es wäre zu untersuchen, inwiefern sie von den zusätzlichen von WikiWord berücksichtigten Strukturen und Konventionen der Wikipedia profitieren. Die von WikiWord erzeugte Datenbasis ist aber auch als eigenständiger, multilingualer, konzeptorientierter Thesaurus nützlich für Aufgaben wie Indexierung, Disambiguierung und Bestimmung von semantischer Ähnlichkeit (vergleiche ▹IV).

Teil II.

Entwurf

3. Idee

Das zu erstellende Softwaresystem WikiWord soll in effizienter Weise die für einen multilingualen, konzeptorientierten Thesaurus benötigten Relationen aus dem Wiki-Text extrahieren (für eine detaillierte Aufstellung von Anforderungen, siehe ▹5). Die Grundidee ist dabei, die Extraktion im Wesentlichen auf die unterschiedlichen Arten von Hyperlinks, die in Wikipedia-Artikeln vorkommen, zu beschränken (▹A.3):

- Einfache Wiki-Links liefern Informationen über den semantischen Kontext von Konzepten — sie können unter anderem verwendet werden, um semantische Verwandtheit zu bestimmen (vergleiche ▹2.3, ▹2.4 und insbesondere [Gregorowicz und Kramer (2006)]).
- Der Link-Text liefert zudem Informationen über Bezeichnungen, die für das Konzept (das Link-Ziel) verwendet werden (vergleiche ▹2.3 und besonders [Mihalcea (2007), Eiron und Mccurley (2003), Kraft und Zien (2004)]).
- Kategorisierungs-Links liefern die Subsumtionsrelation (vergleiche ▹2.3 und ▹2.4, insbesondere [Ponzetto und Strube (2007b)])
- Language-Links verbinden Beschreibungen des gleichen Konzeptes in unterschiedlichen Sprachen — aus ihnen kann auf die semantische Ähnlichkeit von Konzepten geschlossen werden, sowohl innerhalb einer Sprache als auch zwischen verschiedenen Sprachen. Sie können daher verwendet werden, um einen multilingualen Thesaurus aufzubauen, indem ähnliche Konzepte aus unterschiedlichen Sprachen zu sprachübergreifenden Konzepten zusammengefasst werden (siehe ▹4.3). Dieser Ansatz wurde in den eingangs erwähnten Arbeiten noch nicht untersucht.

Neben den Links können noch weitere Eigenschaften betrachtet werden, zum Beispiel die auf einer Seite verwendeten Vorlagen. Sie können zur Klassifikation von Seiten und Konzepten (▹8.5 bzw. ▹8.6) verwendet werden, vergleiche ▹2.6. Weiterleitungen und Begriffsklärungsseiten sowie bestimmte *Magic Words* können verwendet werden, um Konzepten zusätzliche Terme (Bezeichnungen) zuzuweisen (siehe ▹8.9 und ▹8.10 sowie ▹9.1).

Dieser Ansatz, der sich auf Wiki-Links und andere Markup-Elemente konzentriert, umgeht eine Anzahl von Schwierigkeiten, die bei den klassischen Methoden der automatischen Thesaurusgenerierung auftreten (vergleiche [Nakayama u. a. (2007a)]): insbesondere findet keine Analyse natürlicher Sprache statt und selbst auf der lexikalischen Ebene werden problematische Aufgaben wie Stammformreduktion (*Stemming*) vermieden. Zudem wird anstelle von statistischen Verfahren und Heuristiken vor allem direkt menschliches Wissen genutzt, das über Regeln und Konventionen in den Wikipedia-Seiten kodiert ist (vergleiche ▹1.2, ▹8, ▹9.1).

4. Ablauf

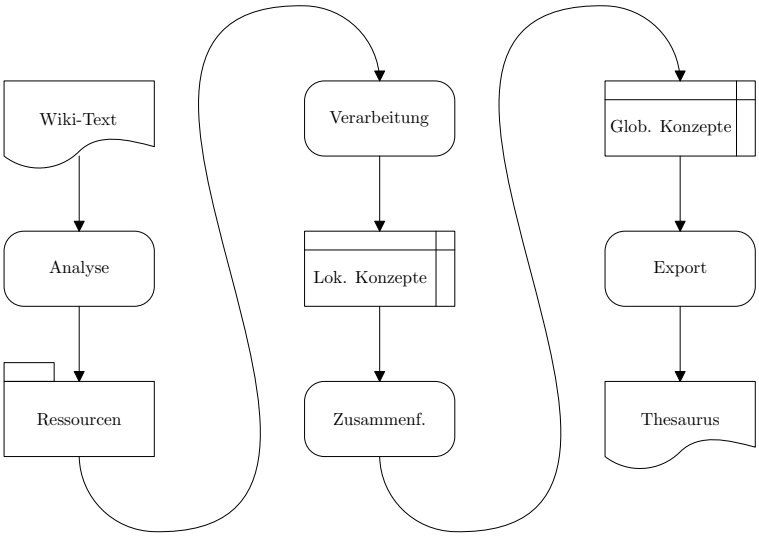


Abb. 4.1: Transformationen

Das Vorgehen von WikiWord lässt sich als eine Folge von Datentransformationen betrachten: In Abb. 4.1 ist eine Aneinanderreihung von Datenmodellen und Verarbeitungsschritten dargestellt, in der jeweils ein Transformationsschritt die Daten von einem Modell in ein anderes überführt[1]. Die Implementation von WikiWord folgt grob dieser Struktur (▹7.1), in der konkreten Umsetzung wurden allerdings einige der Schritte verschachtelt und parallelisiert (▹7.2).

Die einzelnen Transformationen sowie die resultierenden Datenmodelle sind im Folgenden beschrieben.

4.1. Analyse von Wiki-Text

Die erste Transformation ist die Analyse des Wiki-Textes (▹1.2) aus dem XML-Dump (vergleiche [WM:DOWNLOAD, MW:XML]). Hierbei werden die relevanten Informationen aus dem Wiki-Text extrahiert und in das *Ressourcenmodell* überführt, das die Wiki-Seite repräsentiert und direkten Zugriff auf Eigenschaften dieser Seite bietet. Dabei entfällt ein Großteil des textlichen Inhalts, die Seite wird als eine ungeordnete Sammlung von *Features* wie Wiki-Links, Kategorien, Vorlagen und so weiter betrachtet.

Das Ressourcenmodell bietet insbesondere Zugang zu folgenden Eigenschaften der Wiki-Seite:

- der **Titel** der Seite, sowie der Namensraum, zu dem die Seite gehört.

[1] die grafische Aufteilung in mehrere Spalten erfolgt lediglich aus Platzgründen.

- der **Text** der Seite, in verschiedenen Verarbeitungsstufen, vom unveränderten Wiki-Text bis zu reinem Fließtext ohne Markup, siehe ▹8.3.
- alle **Wiki-Links**, also alle Wiki-Seiten, auf die diese Seite verweist, mitsamt dem Link-Text jedes Verweises. Der Mechanismus zur Extraktion der Wiki-Links ist in ▹8.7 beschrieben.

  Manche Wiki-Links haben eine spezielle Semantik (siehe ▹A.3) und werden daher auch von WikiWord gesondert interpretiert. Solche besonderen Wiki-Links definieren:

  – die **Kategorien**, denen die Seite zugeordnet ist.
  – die **Language-Links**, die auf der Seite definiert sind.
- alle **Abschnitte** (Überschriften) der Seite.
- die **Vorlagen**, die auf der Seite verwendet werden, samt den Parametern ihrer Verwendung (▹A.5). Das Verfahren, mit dem die Vorlagen extrahiert werden, ist in ▹8.4 beschrieben.

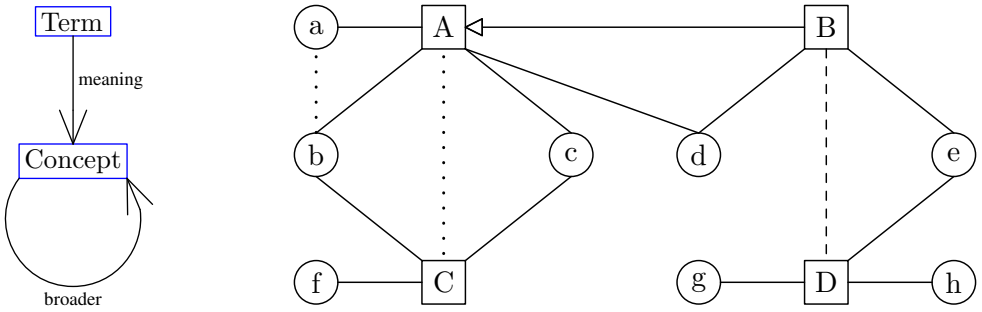
Diese Informationen werden größtenteils durch einfaches *Pattern-Matching* auf Basis des Wiki-Textes gewonnen. Auf Grund dieser Basisinformationen werden (ebenfalls im Wesentlichen durch *Pattern-Matching*) weitere, speziellere Eigenschaften bestimmt:

- der Typ der Seite. Diese Eigenschaft bestimmt, wie die Seite weiter verarbeitet wird. Mögliche Typen von Seiten sind: *Artikel*, *Weiterleitung*, *Begriffsklärung*, *Liste* sowie *Kategorie*. Details dieser Klassifikation sind in ▹8.5 beschrieben.
- der Typ des Konzeptes, das die Seite beschreibt, falls sie ein Artikel ist. Diese Klassifikation hat keinen Einfluss auf die weitere Verarbeitung der Daten, kann aber bei der Verwendung des Thesaurus nützlich sein, insbesondere für die Aufgabe der *Named Entity Recognition*. Mögliche Typen von Konzepten sind: *Ort*, *Person*, *Organisation*, *Name*, *Zeitpunkt*, *Zahl*, *Lebensform* und *sonstige*. Details der Konzeptklassifikation sind in ▹8.6 beschrieben.
- eine Menge von Termen, die als Bezeichnung für das Konzept direkt aus der Seite selbst, insbesondere aus ihrem Titel, abgeleitet werden können (für Details, siehe ▹8.9).
- der erste Satz des Textes der Seite, zur Verwendung als Definition (*Glosse*) für das Konzept, das der Artikel beschreibt. Für das Verfahren zur Extraktion des ersten Satzes, siehe ▹8.8.
- Bedeutungslinks, also Wiki-Links auf einer Begriffsklärungsseite, die auf eine der möglichen Bedeutungen des zu klärenden Begriffs zeigen. Der Algorithmus zur Extraktion von Bedeutungslinks ist in ▹8.10 beschrieben.
- das Ziel der Weiterleitung, falls die Seite eine Weiterleitung ist.

Die verwendeten Muster zur Erkennung bestimmter Informationen im Wiki-Text sind zum Teil Wiki-spezifisch. Zu beachten ist, dass sie zum größten Teil nicht *ad hoc* bestimmt wurden, sondern die vorgegebene Syntax des Wiki-Markups (▹A) oder explizite Konventionen und Richtlinien der einzelnen Wiki-Projekte nachbilden (vergleiche [WP:ARTIKEL, WP:KAT, WP:BKL]).

Die Verfahren, die verwendet werden werden, um die einzelnen Eigenschaften der Wiki-Seite zu bestimmten, wird in ▹8 beschrieben. Das Ressourcenmodell selbst ist in ▹D.1 definiert.

4.2. Aufbau des lokalen Datenmodells



(a) Relationen (b) Struktur

Abb. 4.2: Datenmodell

Die zweite Transformation ist die Verarbeitung der Informationen aus dem Ressourcenmodell, also der Import in das lokale Datenmodell (das *Konzeptmodell*). In diesem Schritt werden die Informationen aus der Wikipedia als Elemente eines Thesaurus *interpretiert* (▹9.1).

Das lokale Datenmodell definiert die Struktur eines monolingualen, konzeptorientierten Thesaurus. Konkrete lokale Datensätze, also konkrete monolinguale Thesauri, werden in dieser Struktur gespeichert. Das Datenmodell ist ein Term-Konzept-Netzwerk (Abb. 4.2), es enthält Beziehungen zwischen Termen und Konzepten (die *Bedeutungsrelation*) und zwischen verschiedenen Konzepten (zum Beispiel *Subsumtion* oder *Verwandtheit*), wie in Abb. 4.2(a) dargestellt. Beziehungen zwischen Termen ergeben sich implizit. Dieses Modell ähnelt den Strukturen, die in der Literatur für ähnliche Zwecke vorgeschlagen wurden, insbesondere z. B. in [Gregorowicz und Kramer (2006)] und anderen Arbeiten, die in ▹2.4 vorgestellt wurden. Die Umsetzung dieses Modells als Schema für ein relationales Datenbanksystem ist in ▹C.1 beschrieben.

Im Einzelnen besteht das Modell aus den folgenden Elementen:

Terme: Terme (in Abb. 4.2(b) als Kreise dargestellt) sind lexikalische Einheiten: Wörter, Wortformen, Wortgruppen, Phrasen, Namen und so weiter. Jeder Term existiert nur einmal in einem Datensatz (also höchstens einmal pro Sprache). Terme sind *Bezeichnungen* für Konzepte.

Konzepte: Konzepte (in Abb. 4.2(b) eckige Kästchen) sind logische Einheiten: sie repräsentieren (abstrakte oder konkrete) Dinge. Konzepte sind *Bedeutungen* von Termen.

Zu jedem Konzept werden neben seiner Identität weitere Eigenschaften gespeichert, unter anderem ein Name zur Anzeige in einer Benutzerschnittstelle und der Typ des Konzeptes (wie oben im Abschnitt über das Ressourcenmodell beschrieben), sowie weitere sekundäre Eigenschaften wie ein IDF-Wert (▹9.3).

Bedeutungsrelation: Die *Bedeutungsrelation* (in Abb. 4.2(b) als durchgezogene Linien dargestellt) verbindet Terme und Konzepte, zum Beispiel die Terme `a`, `b` und `c` mit Konzept `A`. Dabei können beliebig viele Terme einem Konzept und beliebig viele Konzepte einem

Term zugeordnet sein. Die Bedeutungsrelation dient der Überbrückung der *Semantic Gap* zwischen dem Text als Folge von Wörtern und dem Wissen über Konzepte (siehe dazu auch [Suchanek u. a. (2007)] und wieder [Gregorowicz und Kramer (2006)]). Die Bedeutungsrelation ist gewichtet, konkret wird angegeben, wie oft welche Bezeichnung für welches Konzept verwendet wurde.

Die gepunkteten Verbindungen in der Abbildung ergeben sich implizit aus der Bedeutungsrelation: Sie zeigt *Synonymie* zwischen zwei Termen, die beide über die Bedeutungsrelation mit demselben Konzept verbunden sind (zum Beispiel `a` und `b` wegen ihrer Verbindung zu `A`) und *Homonymie* zwischen Konzepten, die über die Bedeutungsrelation mit demselben Term verbunden sind (zum Beispiel zwischen `A` und `C` wegen ihrer Verbindung zu `b` und `c`)[2].

Subsumtionsrelation: Die *Subsumtionsrelation* (in Abb. 4.2(b) als Pfeil dargestellt) verbindet allgemeinere mit spezielleren Konzepten. Sie ist irreflexiv und antisymmetrisch, sowie konzeptionell transitiv, und bildet auf der Menge der Konzepte einen gerichteten azyklischen Graphen. Idealerweise (aber nicht notwendigerweise) sind alle Konzepte von einer einzigen „Wurzel“ aus erreichbar. Die konkrete Semantik der Subsumtion bleibt unbestimmt, die Relation beinhaltet so unterschiedliche Beziehungen wie *Abstraktion* (*subtype*), *Instanziierung* (*is-a*), *Aggregation* (*part-of*) und gelegentlich auch *Attribution* (*property-of*); siehe dazu [Voss (2006), Ponzetto und Strube (2007b), Hammwöhner (2007)] sowie weitere Arbeiten, die in ▹2.4 erwähnt wurden. Diese Relation entspricht der Hyponymiebeziehung zwischen Termen, also der BT/NT-Beziehung nach ISO [ISO:2788 (1986)].

Verwandtheit und Ähnlichkeit: Die semantische *Verwandtheit* und *Ähnlichkeit* von Konzepten (in Abb. 4.2(b) eine gestrichelte Linie) ist eine symmetrische Beziehung. Sie ist unter anderem für die automatische *Disambiguierung* von Termen sowie für die Aufgabe der *Query Expansion* nützlich (vergleiche dazu die Arbeiten, die in ▹2.3 vorgestellt wurden, insbesondere [Zesch u. a. (2007b)] und [Strube und Ponzetto (2006)]).

Lokale Datensätze werden in einer relationalen Datenbank abgelegt (siehe ▹C.1), für die programmatische Verwendung einzelner Konzepte gibt es aber auch eine objektorientierte Repräsentation als *Transferobjekte* der *Datenzugriffsschicht* (siehe ▹C.3). Jedes Konzept wird als Objekt dargestellt, das Zugriff auf die Eigenschaften des Konzeptes bietet, unter anderem auf alle Bezeichnungen für das Konzept sowie auf verwandte und untergeordnete Konzepte (▹D.2).

Die Beziehungen des lokalen Datenmodells werden aus dem Ressourcenmodell aufgebaut, das heißt, die in den einzelnen Ressourcen (Wiki-Seiten) enthaltenen Informationen werden als Beziehungen im lokalen Datenmodell (also im Thesaurus) *interpretiert* und dann als solche gespeichert. Insbesondere werden folgende Interpretationen vorgenommen:

- Seiten, die keine Weiterleitungen, Begriffsklärungen, Listen, Kategorien oder andere besondere Seiten sind, werden als *Artikel* betrachtet, das heißt, es wird angenommen, dass sie genau ein Konzept beschreiben. Entsprechend wird für jeden Artikel ein Konzept im lokalen Datensatz angelegt, mit dem Konzepttyp, den das Ressourcenmodell für diese Seite

[2] Der Übersichtlichkeit halber wurden nicht alle Beziehungen, die sich so ergeben, in der Abbildung dargestellt.

bestimmt hat; das entspricht dem Vorgehen in der Literatur, wie in den Arbeiten in ▹2.3 und ▹2.4 beschrieben.

- die Subsumtionsrelation wird direkt aus den Kategorien abgeleitet, denen eine Seite zugeordnet ist. Dabei wird nicht zwischen Kategorieseiten und Artikeln unterschieden, das heißt, Kategorien sind ebenfalls einfach Konzepte (▹9.1). Auch das entspricht im Wesentlichen dem, was laut ▹2.4 das übliche Vorgehen ist.
- die Verwandtheit von Konzepten wird aufgrund von Querverweisen (Wiki-Links) zwischen Artikeln bestimmt: wenn zwei Artikel sich gegenseitig über Wiki-Links referenzieren, wird davon ausgegangen, dass die Konzepte, die diese Artikel beschreiben, miteinander verwandt sind [Gregorowicz und Kramer (2006)], siehe ▹9.1.3.
- die Ähnlichkeit von Konzepten wird über die Language-Links bestimmt: wenn zwei Artikel über Language-Links denselben Artikel in einer anderen Sprache referenzieren, so werden die Konzepte, die diese beiden Artikel beschreiben, als ähnlich angesehen. Der Grund ist, dass Language-Links immer auf ähnliche (oder, idealerweise, äquivalente) Artikel verweisen [WP:INTERLANG], und die Ähnlichkeitsbeziehung als transitiv und symmetrisch betrachtet wird. Language-Links wurden in [Hammwöhner (2007b)] untersucht, der Ansatz, sie zur Bestimmung von semantischer Ähnlichkeit zu verwenden, scheint jedoch noch nicht verfolgt worden zu sein.
- die Bezeichnungen (Terme) für ein Konzept werden aus einer Vielzahl von Quellen bestimmt, unter anderem:
  - aus dem Titel des Artikels, sowie der Verwendung des *Magic Words* `DISPLAYTITLE`. Letzteres wurde in der Literatur bislang nicht betrachtet.
  - aus den Titeln von Weiterleitungen, die auf den Artikel verweisen. Weiterleitungsseiten werden von so gut wie allen Arbeiten zur Struktur von Wikipedia betrachtet, vergleiche ▹2.3 und ▹2.4.
  - aus den Titeln von Begriffsklärungsseiten, die auf den Artikel verweisen. Begriffsklärungsseiten werden in der Literatur ebenfalls häufig berücksichtigt, vergleiche z. B. [Ponzetto und Strube (2007c)].
  - aus dem Link-Text von Wiki-Links, die auf den Artikel verweisen. Diese Information wurde bei der Analyse der Wikipedia bislang kaum genutzt (mit der Ausnahme von [Mihalcea (2007)] sowie einigen Arbeiten zur *Named Entity Recognition*, siehe ▹2.6), obwohl sie für das Information Retrieval sehr wertvoll sein kann (vergleiche [Eiron und Mccurley (2003), Kraft und Zien (2004)]).
  - aus *Sort-Keys*, die bei der Kategorisierung des Artikels angegeben wurden, sowie der Verwendung des *Magic Words* `DEFAULTSORT`. Diese Quellen für alternative Bezeichnungen wurden in der Literatur bisher nicht untersucht.

Die Interpretation von Informationen aus dem Wiki-Text ist in ▹9.1 näher beschrieben. Nach dem Import der Daten aus den einzelnen Seiten der Wikipedia findet eine Nachbereitung statt (▹9.1.3). Dabei werden insbesondere Weiterleitungen und Kategorisierungs-Aliase aufgelöst (siehe ▹9.1.3 und ▹9.1.2) sowie die Verwandtheit und Ähnlichkeit von Konzepten berechnet (▹9.1.3). Das Ergebnis ist ein lokaler Datensatz, der einen monolingualen Thesaurus repräsentiert.

4.3. Zusammenführen

Die dritte Transformation ist das Zusammenführen der einzelnen lokalen Datensätze zu einem globalen Datensatz: Gruppen von untereinander ähnlichen Konzepten aus unterschiedlichen Sprachen werden zu jeweils einem sprachunabhängigen Konzept vereinigt. Eine solche Methode zum Aufbau eines multilingualen Thesaurus wurde in den eingangs erwähnten Arbeiten nicht beschrieben.

Das globale Datenmodell enthält vor allem Informationen darüber, welche Konzepte aus den einzelnen sprachspezifischen Datensätzen zu einem sprachunabhängigen Konzept zusammengefasst wurden und wie die Beziehungen zwischen den lokalen Konzepten auf die globalen Konzepte abgebildet wurden (siehe ▸C.1). Zusammen mit den lokalen Datensätzen, auf die er sich bezieht, bildet der globale Datensatz eine *Kollektion*, die einen multilingualen Thesaurus repräsentiert.

Jedes Konzept im globalen Datensatz ist eine Menge von lokalen Konzepten, die aus jeder Sprache (höchstens) ein Konzept enthält. Sprachspezifische Eigenschaften (insbesondere Terme und Glossen) werden in dem globalen Datensatz nicht noch einmal abgelegt — statt dessen werden sie bei Bedarf dem betreffenden lokalen Datensatz entnommen.

Die Hauptaufgabe beim Aufbau des globalen Datensatzes ist es also, Gruppen von Konzepten aus unterschiedlichen Sprachen (also lokalen Datensätzen) zu finden, die sich möglichst ähnlich sind, bzw. idealerweise zueinander äquivalent. Dabei ist die unter Umständen sehr unterschiedliche Granularität und Abdeckung der Wikipedias in den einzelnen Sprachen zu berücksichtigen. Der hierfür verwendete Algorithmus lässt sich wie folgt zusammenfassen:

1. Nimm alle Konzepte aus allen Sprachen der Kollektion vorläufig in den multilingualen Thesaurus auf.

2. Bestimme, welche Konzepte über einen Language-Link direkt auf ein anderes Konzept in dem multilingualen Thesaurus, der ja Konzepte aus unterschiedlichen Sprachen enthält, verweisen. Auf diese Weise werden Paare von ähnlichen Konzepten im multilingualen Thesaurus markiert.

3. Bestimme, welche Paare von Konzepten sich auf diese Weise *gegenseitig* referenzieren. Da die über einen Language-Link referenzierten Konzepte im Allgemeinen entweder äquivalent oder allgemeiner sind als das betreffende Konzept selbst, ist davon auszugehen, dass zwei Konzepte, die sich gegenseitig auf diese Weise referenzieren, äquivalent sind oder sich zumindest stark ähneln. Dieses Vorgehen ist analog zu der Methode, die [Gregorowicz und Kramer (2006)] folgend zur Bestimmung verwandter Konzepte innerhalb einer Sprache verwendet wurde.

4. Vereinige Paare solcher Konzepte, wobei alle Eigenschaften und Relationen der beiden Konzepte zusammengefasst werden. Dabei wird mitgeführt, welche Sprachen das vereinigte Konzept abdeckt. Jede Sprache darf in dem resultierenden sprachübergreifenden Konzept höchstens einmal vorkommen.

5. Vereinige so lange, bis kein solches Paare von Konzepten mehr vorhanden ist, das vereinigt werden kann. Sind zwei Konzepte zwar durch gegenseitige Language-Links verbunden, aber die Mengen der von ihnen bereits abgedeckten Sprachen überlappen sich, so stehen diese beiden Konzepte in *Konflikt* zueinander und können nicht vereinigt werden. Solche Konzepte sind sich in der Regel sehr ähnlich (siehe auch ▹11.3).

Dieses Verfahren ist in ▹9.2 genauer beschrieben. Auf das eigentliche Zusammenführen folgt auch hier eine Nachbereitungsphase zur Konsolidierung der Daten (▹9.2.3). Dabei werden aufgrund der neu bestimmten Beziehungen zwischen den sprachübergreifenden Konzepten die Verwandtheit und Ähnlichkeit der Konzepte neu berechnet.

4.4. Export

Die vierte und letzte Transformation ist der Export der lokalen und globalen Datensätze in eine extern weiterverwendbare Form, nämlich das Thesaurus-Modell (siehe ▹10). Es repräsentiert die aus der Wikipedia extrahierten Beziehungen in einer standardisierten, für andere Softwaresysteme nützlichen Form, nämlich RDF/SKOS (▹10.3). Die Modellierung der Thesaurusdaten in RDF/SKOS folgt insbesondere [Gregorowicz und Kramer (2006), van Assem u. a. (2006), W3C (2005)].

Teil III.

Implementation

5. Anforderungen

Im Folgenden werden die Anforderungen aufgeführt, die an *WikiWord*, das System zur automatischen Extraktion eines multilingualen Thesaurus aus der Wikipedia, gestellt werden.

5.1. Aufgaben

WikiWord soll in der Lage sein, auf Basis des Wikipedia-Korpus die folgenden Informationen effizient zu liefern:

1. Eine Liste der Konzepte (Bedeutungen) die zu einem Term (Bezeichnung, Wort, Phrase) einer bestimmten Sprache gehören, mit Ranking nach Häufigkeit/Wahrscheinlichkeit.
2. Eine Liste aller Terme (Bezeichnungen, Wörter, Phrasen) für ein Konzept in einer gegebenen Sprache, mit Ranking nach Häufigkeit/Wahrscheinlichkeit.
3. Übersetzungen eines Terms aus einer Sprache in eine andere, gegebenenfalls getrennt für unterschiedliche Bedeutungen. Das ergibt sich aus einer Kombination der beiden vorausgegangenen Punkte: a) Finden aller möglichen Bedeutungen eines Terms in einer Sprache, b) Anzeige aller Bedeutungen mit jeweils allen möglichen Bezeichnungen in einer anderen Sprache.
4. Über- bzw. untergeordnete Konzepte zu einem gegebenen Konzept, wobei diese Subsumtionsrelation *azyklisch* sein soll.
5. Eine grobe, sprachunabhängige Klassifikation von Konzepten, also Zuordnung eines von einigen wenigen *fest vorgegebenen* Typen (z. B. Person, Ort, Zeit, etc).
6. Die *semantische Verwandtheit* (*Semantic Relatedness, SR*) zwischen zwei gegebenen Konzepten.
7. Eine Zuweisung einer Bedeutung (Konzept) zu jeweils einem Term in einer Menge von Termen (Disambiguierung im Kontext).
8. Eine Glosse (Definitionssatz) zu (möglichst) jedem Konzept.

5.2. Rahmenbedingungen

1. Als Eingabe soll ausschließlich der Wiki-Text (▹A) von den Seiten der Wikipedia dienen, wie er in den Wikipedia XML-Dumps enthalten ist [MW:XML, WM:DOWNLOAD]. Zur Verarbeitung soll nur ein Minimum von sprach- und Wikipedia-spezifischem Wissen benötigt werden.

2. Die Extraktion der gewünschten Daten sowie ihre Verarbeitung und Überführung in die für die Nutzung gewünschte Form soll hinreichend effizient sein: das Verfahren muss auf eine Datenbasis von mehreren Millionen Dateien anwendbar sein und in annehmbarer Zeit terminieren. Anzustreben ist dabei ein in Bezug auf die Anzahl der im Korpus enthaltenen Links subquadratischer Zeit- und Platzaufwand.

3. Die Daten sollen zur weiteren Nutzung in einem relationalen Datenbanksystem abgelegt werden, in einer Form, die eine effiziente Abfrage bezüglich der intendierten Datennutzung erlaubt.

4. Die erzeugten Daten sollen in bestehende Thesaurus- oder Wörterbuchsysteme integrierbar sein.

5. Das Softwaresystem soll leicht anzupassen und zu erweitern sein. Insbesondere soll die Software ohne Schwierigkeiten an Änderungen der Konventionen einzelner Wikipedias, an der Wiki-Text-Syntax oder an dem für die Dumps verwendeten XML Format angepasst werden können.

6. Das Softwaresystem soll auf verschiedenen Betriebssystemen lauffähig bzw. leicht portierbar sein.

6. Plattform

WikiWord, das System zur Extraktion eines multilingualen Thesaurus, soll leistungsfähig, flexibel und portierbar sein. In Hinblick darauf sind vor allem zwei Dinge als Basis für die Implementation auszuwählen: Die Programmiersprache (und Laufzeitumgebung) sowie die Persistenzplattform zur Datenspeicherung. In Hinblick auf die Weiterverwendbarkeit der Daten muss eine für die Weiternutzung durch Dritte geeignete externe Darstellung gefunden werden.

6.1. Laufzeitumgebung: Java

Als Programmiersprache und Laufzeitumgebung wurde **Java** gewählt, unter anderem aus folgenden Gründen:

- Java verbindet gute Performanz mit einem hohen Grad an Abstraktion und Modularität.
- Java bietet eine große Auswahl frei verfügbarer Bibliotheken.
- Java ist im akademischen Umfeld weit verbreitet wegen seiner guten Lesbarkeit und wohldefinierten Semantik.

6.2. Datenbank: MySQL

Als internes Speichersystem wurde ein relationales Datenbanksystem gewählt, nämlich **MySQL** [MYSQL]:

- Die zu speichernden Daten sind relationaler Natur.
- Ein relationales Datenbanksystem bietet effizienten Zugriff auch bei sehr großen Datenmengen.
- Die Nutzung eines Datenbanksystems vermeidet zusätzlichen Programmieraufwand für die Persistenzlogik.
- MySQL ist freie Software und läuft auf allen verbreiteten Betriebssystemen.
- MySQL ist für den Umgang mit sehr großen Datenmengen wohl erprobt und gilt als verhältnismäßig performant.

Die Anbindung von MySQL an die Java-Anwendung erfolgt über Javas Standard Datenbankschnittstelle, JDBC [JDBC], und wird durch eine Datenzugriffsschicht (DAO) gekapselt (siehe ▹C.3).

6.3. Austauschformat: SKOS

Als Datenformat für den Austausch und die Weiterverwendung des erstellten Thesaurus wurde SKOS [W3C:SKOS (2004), Miles (2005b)] gewählt, insbesondere aus den folgenden Gründen [Gregorowicz und Kramer (2006), van Assem u. a. (2006), W3C (2005)]:

- SKOS ist ein vorgeschlagener W3C-Standard und baut auf den wohlbekannten Standards XML, RDF und OWL auf.
- SKOS wurde für die Repräsentation von Thesauri und Wissensnetzen entworfen und eignet sich daher sehr gut für die vorliegenden Daten.
- SKOS ist (im Gegensatz zu dem ISO-Modellen [ISO:2788 (1986)]) konzeptorientiert und passt daher gut zu dem verwendeten Datenmodell.
- SKOS wird von einer Vielzahl von Systemen zur Sprach- und Wissensverarbeitung unterstützt.

7. Framework

In den vorangegangenen Kapiteln wurde ein grober Entwurf für die Funktionalität von WikiWord entwickelt (▹3) und die zu bewältigenden Aufgaben sowie die zu berücksichtigenden Rahmenbedingungen spezifiziert (▹5). Dieses Kapitel gibt nun einen Überblick über die zu diesem Zweck entwickelten Softwarekomponenten (siehe Anhang G für Zugang zum Quellcode). Eine Beschreibung der Kommandozeilenschnittstelle für WikiWord findet sich in Anhang B.

7.1. Import-Architektur

Dieser Abschnitt gibt eine Übersicht über die wichtigsten Klassen und Komponenten von WikiWord. Im Zentrum der Betrachtung steht dabei `ImportConcepts`, das Programm zur Extraktion der Thesaurusdaten aus dem Wiki-Text.

Die Programme, die die Kommandozeilenschnittstelle von WikiWord bilden, sowie die Parameter und Optionen für ihren Aufruf sind in Anhang B beschrieben. Weitere Möglichkeiten zur Konfiguration bieten die sogenannten *Tweak-Werte*, siehe ▹B.5. Eine Auflistung der relevanten Klassen und Packages ist in ▹G gegeben.

Die Basis für alle Programme der Kommandozeilenschnittstelle von WikiWord ist die Klasse `CliApp`: Sie stellt Basisfunktionalität unter anderem zur Verarbeitung von Kommandozeilenparametern und zur Ausgabe von Log-Nachrichten bereit. Alle in ▹B.1 erwähnten Import-Programme mit Ausnahme von `ExportSkos`, bauen auf der von `CliApp` abgeleiteten Klasse `ImportApp` auf: dieses bietet Unterstützung für die Verwendung des *Agenda*-Systems zur Ablaufkontrolle (siehe Anhang B.6) sowie für die Erzeugung eines geeigneten Datenzugriffsobjektes für die Interaktion mit der Datenbank (siehe ▹C.3).

Eine genauere Betrachtung verdient die Klasse `ImportDump`, auf der auch `ImportConcepts` basiert, welches der Einstiegspunkt für einen Großteil der Extraktionslogik ist. `ImportDump` ist von `ImportApp` abgeleitet und bildet das Framework für eine *Consumer-Producer*-Architektur für den Import von Daten: Eine Implementation des Interfaces `ImportDriver` liest „Seiten“ aus einer Datenquelle und übergibt sie, eine nach der anderen, an eine Implementation des Interfaces `WikiWordImporter`, die die Daten dann weiter verarbeitet. Die Klasse `AbstractImporter` bildet die Basis für Implementationen dieses Interfaces und bietet Funktionen insbesondere für die Ablaufsteuerung über das Agenda-Objekt sowie für die Umsetzung allgemeiner Kommandozeilenparameter.

Im Normalfall sollen die Wiki-Seiten aus einem Wikipedia-Dump gelesen werden [MW:XML]. Dies wird von der Klasse `DumpImportDriver` umgesetzt (einer Implementation des Interfaces `ImportDriver`), die ihrerseits auf die MWDumper-Bibliothek von Wikimedia zurückgreift [MW:DUMPER]: Sie verarbeitet die XML-Struktur und ruft für jede enthaltene Wiki-Seite die Methode `handlePage` auf dem `WikiWordImporter` auf. Dabei wird die XML-Datei automatisch dekomprimiert, sollte dies erforderlich ein[1].

[1] Unterstützt werden die weit verbreiteten Formate GZip und BZip2.

Für die Analyse des Wiki-Textes und den Import der so gewonnenen Ressourcen in das lokale Datenmodell verwendet das Programm `ImportConcepts` die Klasse `ConceptImporter` als Implementation von `WikiWordImporter`. Diese überführt den rohen Wiki-Text zunächst unter Verwendung von `WikiTextAnalyzer` in das Ressourcenmodell (siehe ▹4.1, ▹8, ▹D.1), interpretiert die Eigenschaften der Ressource (▹4.2, ▹9.1) und schreibt das Ergebnis dann in das lokale Datenmodell, ein Datenzugriffsobjekt des Typs `LocalConceptStoreBuilder` (siehe und ▹C.4).

Die Interaktion von `DumpImportDriver`, `ConceptImporter`, `WikiTextAnalyzer` und `LocalConceptStoreBuilder` wird im nächsten Abschnitt genauer erläutert.

7.2. Ablauf und Parallelisierung (Pipeline)

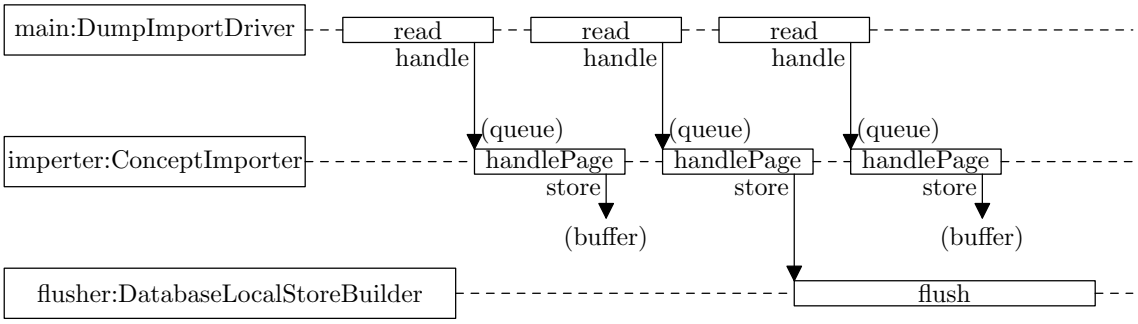


Abb. 7.1: Parallelisierung: die Import-Pipeline

Die Aufgabe von `ImportConcepts` lässt sich in zwei Abschnitte gliedern: den eigentlichen Datenimport, also das Lesen, Interpretieren und Speichern der Daten, und die Nachbereitung, also die Konsolidierung der Daten. Der Datenimport ist selbst in mehrere Schritte gegliedert, von denen einige parallel ausgeführt werden können — diese Parallelisierung soll hier näher beschrieben werden.

Die am Datenimport beteiligten Komponenten sind `DumpImportDriver`, `ConceptImporterWikiTextAnalyzer` und `LocalConceptStoreBuilder` bzw. `DatabaseLocalConceptStoreBuilder`. Diese Komponenten arbeiten zum Teil parallel (siehe Abb. 7.1) und sind wie folgt gekoppelt:

Im Haupt-Thread des `ImportConcepts`-Programms wird die `runImport`-Methode von `DumpImportDriver` aufgerufen — sie ist der Einstiegspunkt für den gesamten Datenimport. `DumpImportDriver` extrahiert mit Hilfe der MWDumper-Bibliothek den Wiki-Text der einzelnen Seiten aus der XML-Struktur der Dump-Datei. Der Wiki-Text jeder Seite wird dann, zusammen mit der zugehörigen Metainformation wie dem Seitentitel, in eine Warteschlange eingefügt, aus der er, gemäß dem *Consumer-Producer*-Muster, von einem anderen Thread herausgenommen wird, der ihn dann an die Methode `handlePage` von `ConceptImporter` übergibt. Die Warteschlange ist auf n Einträge beschränkt: wenn sich schon n unbearbeitete Einträge in der Warteschlange befinden, wenn der Producer (`DumpImportDriver`) einen Eintrag

hinzufügen will, so wird dieser blockiert, bis der Consumer (`ConceptImporter`) einen Eintrag aus der Warteschlange entnommen hat. Die Länge der Warteschlange ist per Default 8 und kann über den Tweak-Wert `dumpdriver.pageImportQueue` konfiguriert werden — die Länge ist aber relativ unerheblich, da die Warteschlange in der Regel voll und der Producer-Thread blockiert ist. Die Warteschlange dient also nicht als Puffer, sondern lediglich der Entkopplung der beiden Aktivitäten, die eine parallele Ausführung erlaubt; eine Entkopplung nach dem *Rendezvous*-Muster wäre hier ebenfalls ausreichend. Wird `dumpdriver.pageImportQueue` auf 0 gesetzt, so wird der Import „durchgekoppelt“ und keine Warteschlange verwendet — in diesem Fall ruft der `DumpImportDriver` direkt die Methode `handlePage` von `ConceptImporter` auf.

Wie oben beschrieben, wird der `ConceptImporter` in einem eigenen Thread betrieben und aus einer Warteschlange bedient. Für jeden Eintrag aus der Warteschlange wird die Methode `handlePage` mit dem Wiki-Text und den zugehörigen Metadaten wie dem Seitentitel aufgerufen. Diese Methode verwendet nun zunächst den `WikiTextAnalyzer`, um den Wiki-Text in das Ressourcenmodell, implementiert durch `WikiTextAnalyzer.WikiPage` (siehe ▹D.1), zu überführen, das heißt, den Text zu parsen und die relevanten Features (▹4.1) wie Wiki-Links, verwendete Vorlagen etc. zu extrahieren. Die so bestimmten Eigenschaften werden dann interpretiert und entsprechende Informationen werden zur Speicherung an die Datenzugriffsschicht übergeben (siehe ▹9.1 und ▹C.4).

Die datenbankbasierte Implementation der Datenzugriffsschicht (DAO), `DatabaseLocalConceptStoreBuilder`, verwendet intern für jede Datenbanktabelle einen Puffer, in dem neue Einträge zwischengespeichert werden. Ist ein Puffer voll, so wird er in eine Warteschlange gelegt und ein neuer Puffer für die betreffende Tabelle verwendet. Ein separater Thread arbeitet im Hintergrund die Puffer aus der Warteschlange ab, indem er einen nach dem anderen entnimmt und an die Datenbank weiterleitet. Das entspricht einer asynchronen *Flush*-Funktion für die Puffer. Bei einem expliziten Aufruf der Methode `flush` werden die Puffer aller Tabellen in die Warteschlange des Flush-Threads gelegt und dann gewartet, bis alle Daten in die Datenbank geschrieben wurden.

Die Länge der Warteschlange für die asynchronen *Flush*-Funktion ist per Default 4 und kann über den Tweak-Wert `dbstore.backgroundFlushQueue` konfiguriert werden — die Länge ist aber relativ unerheblich, da die Warteschlange in der Regel leer sein sollte, da nur recht selten Einträge in ihr abgelegt werden. Sollte die Warteschlange häufig voll sein und damit den Arbeitsthread blockieren, sollte die Größe der Einfügepuffer erhöht werden — das kann über den Tweak-Wert `dbstore.insertionBufferFactor` geschehen. Dabei ist allerdings zu beachten, dass die Größe jedes einzelnen Puffers durch die MySQL-Konfigurationsvariable `max_allowed_packet` begrenzt ist — gegebenenfalls muss auch diese erhöht werden. Die Warteschlange dient also vornehmlich der Entkopplung der langwierigen Flush-Operation, so dass diese parallel zur Analyse des Wiki-Textes ausgeführt werden kann. Wird `dbstore.backgroundFlushQueue` auf 0 gesetzt, so wird die Flush-Operation „durchgekoppelt“ und keine Warteschlange verwendet — in diesem Fall muss der Arbeitsthread warten, während volle Einfügepuffer in die Datenbank übertragen werden. Werden die Tweak-Werte `dbstore.useEntityBuffer` und `dbstore.useRelationBuffer` auf `false` gesetzt, so wird für das Einfügen in die Datenbank gar kein Puffer benutzt, jeder Eintrag wird sofort übertragen. Dieser Modus ist allerdings sehr langsam und nur für die Fehlersuche sinnvoll.

8. Analyse des Wiki-Textes

Dieses Kapitel beschreibt, wie der Wiki-Text analysiert wird, um alle für WikiWord relevanten Eigenschaften (*Features*) zu extrahieren und wie diese Eigenschaften im Ressourcenmodell repräsentiert werden. Relevante Eigenschaften sind unter anderem die Wiki-Links (inklusive Kategorisierung, Language-Links etc., siehe ▹A.3), der Einleitungssatz (Glosse) und die verwendeten Vorlagen und ihre Parameter (▹A.5).

Die Logik zur Analyse von Wiki-Text (und, soweit notwendig, von natürlicher Sprache), sind im Java Package `de.brightbyte.wikiword.analyzer` implementiert. Das Package `de.brightbyte.wikiword.wikis` enthält Klassen, die das sprach- bzw. projektspezifische Wissen repräsentieren, das notwendig ist, um den Wiki-Text zu analysieren. Die in diesen Packages implementierten Regeln und Muster basieren auf Informationen darüber, wie Wiki-Text im Allgemeinen und Wikipedia-Seiten im Speziellen aufgebaut sind. Insbesondere repräsentieren sie:

1. Die **Wiki-Text Markup Syntax**, wie von MediaWiki festgelegt (Anhang A). Diese Regeln sind, soweit sie für WikiWord relevant sind, fest in `PlainTextAnalyzer` kodiert.
2. Die **Konventionen und Richtlinien** der einzelnen Wiki-Projekte, die zum Beispiel festlegen, wie Artikel strukturiert werden [WP:ARTIKEL], wie Kategorien zu verwenden sind [WP:KAT] oder wann und wie Begriffsklärungsseiten anzulegen sind [WP:BKL]. Diese Regeln sind, soweit sie für WikiWord relevant sind, zumeist in der projektspezifischen Subklasse von `WikiConfiguration` definiert, zum Teil aber auch fest in `WikiTextAnalyzer` kodiert.
3. **Beobachtungen** darüber, wann bestimmte Textmuster auftreten oder wie das Wiki-Text-Markup verwendet wird. Das sind zum Beispiel die Menge der in einer Sprache häufig auftretenden Abkürzungen, wichtig für das Erkennen des Endes der Glosse — diese Information ist als sprachspezifisches Wissen in der betreffenden Subklasse von `LanguageConfiguration` abgelegt. Ein anderes Beispiel sind Vorlagen, die häufig inline im Text verwendet werden und daher vor der Verarbeitung aufgelöst werden sollten. Sie sind mit den anderen projektspezifischen Informationen in einer Subklasse von `WikiConfiguration` definiert.

Zu beachten ist, dass WikiWord *ad-hoc*-Heuristiken nach Möglichkeit vermeidet und statt dessen versucht, direkt explizite Konventionen zu modellieren. So profitiert WikiWord stärker von der *Intention* der Autoren des Wiki-Textes und ist weniger auf die Intuition des Programmierers angewiesen.

8.1. Klassen

Dieser Abschnitt beschreibt die Klassen, die für die Analyse des Wiki-Textes und eine Übertragung in das Ressourcenmodell wichtig sind.

PlainTextAnalyzer: Die Klasse `PlainTextAnalyzer` bietet einfache Funktionen zur Analyse natürlichsprachlicher Texte, insbesondere eine Methode zur Extraktion des ersten Satzes aus einem Text, und eine Funktion zur Extraktion einzelner Wörter aus einer Zeichenkette. Instanzen von `PlainTextAnalyzer` sollten über die Methode `getPlainTextAnalyzer` erzeugt werden: Sie lädt die für die gegebene Sprache passende Implementation und Konfiguration. Die Default-Implementation funktioniert bei entsprechender Konfiguration über eine `LanguageConfiguration` für die meisten europäischen Sprachen.

LanguageConfiguration: Die Klasse `LanguageConfiguration` repräsentiert eine Konfiguration von `PlainTextAnalyzer` für eine bestimmte Sprache. Die wichtigste sprachspezifische Information, die diese Klasse kapselt, ist ein Muster für häufig verwendete Abkürzungen dieser Sprache, das bei der *Satzgrenzenerkennung* verwendet wird. Für Deutsch und Englisch sind Konfigurationen im Package `de.brightbyte.wikiword.wikis` definiert, für weitere Sprachen wie Französisch, Niederländisch und Norwegisch ist eine rudimentäre Basiskonfiguration vorhanden.

WikiTextAnalyzer: Die Klasse `WikiTextAnalyzer` enthält Logik für die Verarbeitung vieler Aspekte des von MediaWiki verwendeten Wiki-Text-Markups (▹A). Insbesondere bietet sie Methoden für den Zugriff auf Wiki-Links, Vorlagen und Artikelabschnitte, sowie Funktionalität zum Entfernen von Wiki-Markup, um „einfachen“ Text zur weiteren Verarbeitung zu erzeugen. Instanzen von `WikiTextAnalyzer` sollten über die Methode `getWikiTextAnalyzer` erzeugt werden: Sie lädt die für den gegebenen Korpus passende Implementation und Konfiguration.

WikiTextAnalyzer.WikiPage: Die Klasse `WikiTextAnalyzer.WikiPage` repräsentiert eine Seite im Wiki und implementiert damit das Ressourcenmodell von WikiWord (▹4.1). Sie bietet Zugriff auf die verschiedenen Eigenschaften der Seite, wie sie von `WikiTextAnalyzer` erkannt wurden. Diese sind im Anhang D.1 beschrieben.

WikiConfiguration: Die Klasse `WikiConfiguration` repräsentiert eine Konfiguration von `WikiTextAnalyzer` für ein bestimmtes Wiki-Projekt. Sie enthält Wissen über diverse Konfigurationen und Konventionen für das betreffende Wiki. Konfigurationen sind im Package `de.brightbyte.wikiword.wikis` vorhanden für die deutschsprachige (`de.wikipedia.org`) und die englischsprachige Wikipedia (`en.wikipedia.org`) sowie rudimentär angelegt für die französischsprachige (`fr.wikipedia.org`), niederländischsprachige (`nl.wikipedia.org`) und norwegischsprachige (`no.wikipedia.org`).

WikiTextSniffer: Die Klasse `WikiTextSniffer` erlaubt es, typische Elemente von Wiki-Text zu erkennen. Das ist nützlich für *Sanity-Checks*: so sollten zum Beispiel Konzeptnamen und Terme, die aus Wiki-Links extrahiert wurden, keinen Wiki-Text mehr enthalten. Tun sie das doch, werden sie ignoriert. Das sollte nur dann auftreten, wenn Wiki-Markup auf inkorrekte oder sehr ungewöhnliche Weise verwendet wurde.

Die Klassen und Dateien, die zur Konfiguration Wiki-spezifischer Werte und Muster verwendet werden, sind in Anhang B.7 beschrieben.

8.2. Mechanismen

Zur Verarbeitung des Wiki-Textes werden verschiedene Techniken verwendet, hauptsächlich aber *Mustererkennung* (*Pattern Matching*), wobei die Muster nicht unbedingt direkt auf den vollen Wiki-Text angewendet werden, sondern zum Teil auf schon vorverarbeiteten Text oder Textstücke. Um die verschiedenen Muster und Regeln in `WikiConfiguration` zu definieren, werden vor allem die folgenden Klassen verwendet:

`Pattern`: Ein regulärer Ausdruck, in Javas eigener Implementation [JAVA:RE]. Wird für einfache Mustererkennung im Text verwendet.

`AbstractAnalyzer.Mangler`: Ein Interface für Klassen, die Text manipulieren, also Ersetzungen irgendeiner Form durchführen. Standard-Implementationen dieses Interfaces sind `WikiTextAnalyzer.BoxStripManger` zum Entfernen von Blockstrukturen aus Wiki-Text ▹8.3, `WikiTextAnalyzer.EntityDecodeManger` zum Auflösen von HTML *Character References* [W3C:HTML (1999)] und `AbstractAnalyzer.RegularExpressionMangler`, der beliebige Textersetzungen auf der Basis von regulären Ausdrücken vornimmt.

`AbstractAnalyzer.SuccessiveMangler`: Ebenfalls ein Interface für Klassen, die Text manipulieren. `SuccessiveMangler` hat dabei zusätzlich die Eigenschaft, dass er den gegebenen Text nicht unbedingt bis zum Ende verarbeitet und dass die Verarbeitung dort fortgesetzt werden kann, wo sie unterbrochen wurde. So kann der Text sukzessive verarbeitet werden, genau so weit, wie das Ergebnis tatsächlich benötigt wird. Die bereits erwähnte Klasse `WikiTextAnalyzer.BoxStripManger` zum Entfernen von Blockstrukturen aus Wiki-Text implementiert dieses Interface zusätzlich zu `AbstractAnalyzer.Mangler` — das ist insbesondere für das Aufspüren des ersten Absatzes eines Artikels nützlich.

`AbstractAnalyzer.Armorer`: Interface für Klassen, die Teile des Textes durch Platzhalter ersetzen, um sie so vor weiteren Verarbeitungsschritten zu schützen. Das ist insbesondere für Abschnitte im Wiki-Text sinnvoll, die als *verbatim* markiert sind, auf die also die Regeln des Wiki-Markup nicht angewendet werden sollen (▹8.3). Das sind insbesondere Kommentare sowie Text in *<nowiki>*- und *<pre>*-Tags (siehe ▹A). Die Standard-Implementation dieses Interfaces ist `AbstractAnalyzer.RegularExpressionArmorer` — sie identifiziert zu schützende Teile des Wiki-Textes mit Hilfe eines regulären Ausdrucks.

`WikiTextAnalyzer.Sensor`: Interface für Klassen, die eine `WikiPage` auf Grund einer ihrer Eigenschaften klassifizieren: die Methode `sense` gibt zurück, ob der `Sensor` zu der gegebenen `WikiPage` passt; die Methode `getValue` liefert den Wert, der der Seite gegebenenfalls zugeordnet werden soll. Für dieses Interface gibt es eine ganze Reihe von Implementationen:

`HasCategorySensor`: Stellt fest, ob die Seite zu einer bestimmten Kategorie gehört.

`HasCategoryLikeSensor`: stellt fest, ob die Seite zu einer Kategorie gehört, deren Name zu einem bestimmten regulären Ausdruck passt.

HasSectionSensor: Stellt fest, ob die Seite einen Abschnitt mit einem bestimmten Namen hat.

HasSectionLikeSensor: stellt fest, ob die Seite einen Abschnitt hat, dessen Name zu einem bestimmten regulären Ausdruck passt.

HasTemplateSensor: Stellt fest, ob die Seite eine bestimmte Vorlage verwendet. Optional kann auch verlangt werden, dass die Vorlage mit einem bestimmten Parameter verwendet wird und dass diesem Parameter ein bestimmter Wert zugewiesen wurde.

HasTemplateLikeSensor: Stellt fest, ob wenn die Seite eine Vorlage verwendet, deren Name zu einem bestimmten regulären Ausdruck passt. Optional kann auch verlangt werden, dass die Vorlage mit einem bestimmten Parameter verwendet wird und dass der Wert, der diesem Parameter zugewiesen wurde, zu einem bestimmten regulären Ausdruck passt.

RegularExpressionCleanedTextSensor: Stellt fest, ob der bereinigte Text der Seite (▹8.3) zu einem bestimmten regulären Ausdruck passt.

RegularExpressionTitleSensor: Stellt fest, ob der Titel der Seite zu einem bestimmten regulären Ausdruck passt.

Sensoren dieser Art werden vornehmlich für die Klassifikation von Ressourcen und Konzepten verwendet (siehe ▹8.5 bzw. ▹8.6).

Mit diesen Mitteln lassen sich die Eigenheiten eines Wikipedia-Projektes ausdrücken und in einer Subklasse von `WikiConfiguration` festhalten, auf die `WikiTextAnalyzer` bei der Analyse von Wiki-Text zurückgreifen kann.

8.3. Bereinigen

Für die Verarbeitung von Wiki-Text ist es nützlich, bestimmte Arten oder Aspekte des Wiki-Markups vorab zu entfernen. Wie in ▹D.1 erwähnt, kennt WikiWord vier Versionen des Wiki-Textes:

text: der unveränderte Wiki-Text.

cleanedText: der Wiki-Text ohne störende Elemente, nach Anwendung von `WikiTextAnalyzer.stripClutter`. Diese Form des Textes ist die Basis für die weitere Analyse (Extraktion von Vorlagen, Links, Kategorien etc).

flatText: der Wiki-Text ohne Blöcke, nach Anwendung von `WikiTextAnalyzer.stripBoxes`. Diese Form des Textes ist die Basis für die weitere Analyse auf der Textebene (Extraktion der Glosse, von Absätzen etc.).

plainText: der Text nach dem Entfernen sämtlicher Markup-Elemente, also nach Anwendung von `WikiTextAnalyzer.stripMarkup`. Diese Form des Textes kann für eine weitere Verarbeitung auf der Ebene natürlicher Sprache verwendet werden.

Die einzelnen Methoden zum Bereinigen des Textes werden im Folgenden näher erläutert:

`WikiTextAnalyzer.stripClutter`: Diese Methode entfernt oder transformiert alle solchen Wiki-Text-Elemente, die für die Analyse im Rahmen dieser Arbeit unnötig oder störend sind, und ruft dann `WikiTextAnalyzer.resolveMagic` auf. Alle anderen Analysemethoden werden auf den so bereinigten Text angewendet (oder auf eine weiter bereinigte Version).

Vollständig entfernt werden *<gallery>*-Blöcke, alle eingebundenen Bilder (also Wiki-Links in den Image-Namensraum) [MW:IMAGES] (▹A.3) sowie alle Optionen der Form `__.*?__` [MW:MAGIC] (▹A.6). Definiert werden diese Ersetzungen durch `WikiConfiguration.stripClutterManglers`.

Ebenfalls durch `WikiConfiguration.stripClutterManglers` werden Ersetzungen definiert, die wohlbekannte, wikispezifische Vorlagen ersetzen, insbesondere Formatierungshilfen, die inline im Text verwendet werden. Für die englischsprachige Wikipedia werden zum Beispiel unter Anderem folgende Ersetzungen vorgenommen: *{{dot}}* durch „·“ (Unicode U+00B7), *{{sub|xxx}}* durch „*xxx*“ und *{{frac|a|b}}* durch „*a*/*b*“. Zudem wird eine Vielzahl von Vorlagen einfach entfernt, zum Beispiel *{{fact}}*, das in der englischen Wikipedia auf eine unbelegte Aussage hinweist. Wichtig ist auch, dass in diesem Schritt Vorlagen ersetzt werden, die Teile von Markup-Konstrukten enthalten, die für die weitere Verarbeitung benötigt werden: In der englischsprachigen Wikipedia muss zum Beispiel die Vorlage *{{wrapper}}* durch „*{|*“ (Tabellenanfang) und *{{end}}* durch „*|}*“ (Tabellenende) ersetzt werden.

Einige Arten von Wiki-Text-Konstrukten werden nicht einfach entfernt oder umgewandelt, sondern durch einen Platzhalter ersetzt, so dass sie zwar von der weiteren Verarbeitung ausgeschlossen sind, aber später wieder eingefügt werden können (*Armoring*): das ist vor allem für Blöcke der Fall, in denen Wiki-Text-Markup keine Anwendung findet, per Default also bei Kommentaren (`<!--.*?-->`) sowie bei *<nowiki>*- und *<pre>*-Blöcken (▹A). Diese Blöcke werden durch `WikiConfiguration.stripClutterArmorers` festgelegt.

`WikiTextAnalyzer.resolveMagic`: Diese Methode ersetzt einige Magic Words durch ihren konkreten Wert [MW:MAGIC]: sie verwendet den in `WikiConfiguration.magicPattern` definierten regulären Ausdruck, um solche Magic Words zu finden, und ruft dann `WikiTextAnalyzer.evaluateMagic` auf, um ihren Wert zu ermitteln. Ersetzt werden solche Magic Words, die als Variablen für verschiedene Versionen des Seitentitels oder des Namensraumes der Seite funktionieren, konkret solche, die zu folgendem regulären Ausdruck passen: `\{\{\s*((FULL|SUB|BASE)?PAGENAMEE?|NAMESPACEE?)\s*\}\}` . Andere Magic Words werden später genau wie Vorlagen behandelt (wenn sie der Vorlagen-Syntax folgen, siehe ▹8.4), oder wurden vorher bereits von `WikiTextAnalyzer.stripClutter` entfernt (im Fall von Optionen wie *__NOTOC__*).

`WikiTextAnalyzer.stripBoxes`: Diese Methode liefert den „eigentlichen“ Inhalt der Wiki-Seite (den *Fließtext*), ohne Bilder, Tabellen, Infoboxen, Hinweise und dergleichen, aber noch mit Markup zur Textformatierung sowie Wiki-Links. Die Ersetzungen, die anzuwenden sind, um das zu erreichen, werden von `WikiConfiguration.stripBoxesManglers`

definiert. Als Eingabe wird Text erwartet, der bereits durch `WikiTextAnalyzer.stripClutter` bereinigt wurde.

Per Default enthält `stripBoxesManglers` genau einen Mangler, nämlich eine Instanz von `WikiTextAnalyzer.BoxStripMangler`: Er entfernt alle expliziten Blockstrukturen, also Tabellen, Vorlagen (soweit nicht bereits ersetzt) sowie * *- und * *-Blöcke; diese werden in der Wikipedia fast nie im Fließtext des Artikels verwendet, sondern nur für „Kästen“ (*Floats*), Warnungen, Hinweise, usw. (*<p>*-Blöcke dagegen kommen manchmal im Fließtext vor, und werden eher selten für Floats verwendet). Das Entfernen der Blockstrukturen erfolgt durch rekursives Parsen anhand der öffnenden und schließenden Elemente der einzelnen Strukturen, implementiert als Kellerautomat in `WikiTextAnalyzer.BoxStripMangler.mangle`.

`WikiTextAnalyzer.stripMarkup`: Diese Methode entfernt alle Markup-Elemente aus einem Text — als Eingabe wird Text erwartet, der bereits durch `WikiTextAnalyzer.stripBoxes` bereinigt wurde. Die Ersetzungen, die anzuwenden sind, um das zu erreichen, werden von `WikiConfiguration.stripMarkupManglers` definiert.

Per Default werden folgende Ersetzungen durchgeführt:

- Zeilen, die nur aus `---` bestehen, definieren horizontale Trennlinien; sie werden entfernt.
- Alle Wiki-Links werden durch ihren Link-Text ersetzt.
- Alle externen Links werden durch ihren Link-Text ersetzt oder, wenn kein Link-Text angegeben ist, durch die URL, auf die sie verweisen.
- Doppelpunkte am Anfang einer Zeile werden durch eine entsprechende Anzahl Tabs ersetzt (Unicode U+0009).
- Alle Gruppen von zwei oder drei aufeinanderfolgenden Apostrophen (Unicode U+0027) werden entfernt.
- Alle verbleibenden XML-artigen Tags werden entfernt — dabei bleibt gegebenenfalls der Text zwischen Start- und End-Tag erhalten.
- Alle HTML-artige Entity-Referenzen werden unter Verwendung von `WikiTextAnalyzer.EntityDecodeMangler.mangle` dekodiert [W3C:HTML (1999)].

8.4. Extraktion von Vorlagen

WikiWord analysiert auch die Verwendung von Vorlagen (*Templates*) auf Wiki-Seiten (▹A), vornehmlich, weil diese einen guten Anhaltspunkt für die Klassifikation der Seite bzw. des Konzeptes bietet. Die Verarbeitung der Vorlagen ist in `WikiTextAnalyzer.extractTemplates` implementiert; sie ist konzeptionell als Kellerautomat umgesetzt, wobei allerdings verschachtelt auftretende Vorlagen lediglich gezählt werden. Nur die Verwendung auf der „obersten“ Ebene wird registriert und gespeichert.

Die Vorlagen, die auf einer Wiki-Seite verwendet werden, werden als Datenstruktur vom Typ `java.util.Map` repräsentiert; die Schlüssel in der Map sind die normalisierten Namen der verwendeten Vorlagen, die den Schlüsseln zugeordneten Werte sind selbst wieder Maps, die die Parameter der Vorlage als Schlüssel-Wert-Paare enthalten; unbenannte (*positionale*) Parameter werden, wie in MediaWiki selbst, mit 1 beginnend durchnummeriert. Die Repräsentation in einer Map hat zur Folge, dass, sollte eine Vorlage auf derselben Seite mehrfach verwendet werden, nur das letzte Auftreten der Vorlage beachtet wird. Zwar kommt es recht häufig vor, dass Vorlagen mehrfach verwendet werden, jedoch trifft dies kaum auf die Art von Vorlage zu, die von WikiWord von Interesse für die Klassifikation der Seite wäre — oder es ist für die Klassifikation zumindest unerheblich, wie oft die Vorlage auf der Seite vorkommt.

Vorlagen, deren Name mit „*#*“ beginnt, werden als *Parser Functions* erkannt und übergangen. Vorlagen, deren Namen einen Doppelpunkt „*:*“ enthält, werden daraufhin untersucht, ob der Teil vor dem Doppelpunkt der Name eines relevanten Magic Words ist (`WikiTextAnalyzer.getMagicTemplateId`) — falls das der Fall ist, wird der Schlüssel auf den kanonischen Namen des Magic Words gesetzt und der Teil des ursprünglichen Namens, der auf den Doppelpunkt folgt, als Parameter mit dem Namen „0“ interpretiert. Auf diese Wiese lässt sich insbesondere die Verwendung von *{{DISPLAYTITLE:...}}* und *{{DEFAULTSORT:...}}* festhalten (siehe ▹A.6); diese Information kann später zur Bestimmung der Bezeichnungen für ein Konzept beitragen (siehe ▹8.9 und ▹9.1).

8.5. Ressourcenklassifikation

Die Methode `WikiTextAnalyzer.determineResourceType` bestimmt den `ResourceType`: Das heißt, sie weist einer Wiki-Seite (*Ressource*) einen Typ aus einer fest vorgegebenen Menge zu (vergleiche ▹4.1). Von diesem Typ hängt die weitere Verarbeitung der Seite ab, siehe ▹9.1.

Die Zuweisung wird in `WikiConfiguration.resourceTypeSensors` definiert, die Regeln sind im Allgemeinen Wiki-spezifisch. Die folgenden Typen werden verwendet:

ARTICLE: Die Ressource ist ein Artikel, das heißt, sie beschreibt ein Konzept. Solche Seiten liefern den eigentlichen enzyklopädischen Inhalt der Wikipedia. Dieser Typ wird allen Seiten zugewiesen, die zu keinem der anderen Typen passen.

REDIRECT: Die Ressource definiert eine Weiterleitung auf eine andere Seite (also einen Alias für das entsprechende Konzept). Dieser Typ wird allen Seiten zugeordnet, die zu dem in `WikiConfig.redirectPattern` festgelegten Muster passen.

DISAMBIG: Die Ressource ist eine Begriffsklärungsseite, das heißt, sie führt verschiedene Bedeutungen eines Terms auf (▹8.10). Seiten dieses Typs lassen sich meist an der Verwendung einer speziellen Vorlage erkennen: in der deutschsprachigen Wikipedia zum Beispiel wird dieser Typ allen Seiten zugewiesen, die die Vorlage *{{Begriffsklärung}}* enthalten.

LIST: Die Ressource ist eine explizite Liste von Seiten (und damit Konzepten) mit einer bestimmten Eigenschaft. Seiten dieses Typs lassen sich meist an ihrem Titel oder einer ihrer

Kategorien erkennen: in der deutschsprachigen Wikipedia zum Beispiel wird dieser Typ allen Seiten zugewiesen, die in einer Kategorie sind, die *Liste* heißt oder deren Name mit *Liste_* beginnt.

CATEGORY: Die Ressource ist eine Kategorie, das heißt, eine Sammlung von Seiten bzw. Konzepten, die dem Konzept, das die Kategorie selbst repräsentiert, untergeordnet sind. Dieser Typ wird allen Seiten aus dem Kategorie-Namensraum zugewiesen, das heißt, sie werden an ihrem Titel erkannt.

BAD: Die Seite ist als problematisch markiert; das betrifft vor allem so genannte *Löschkandidaten* [WP:LR], also Seiten, die aus irgendeinem Grund zur Löschung aus der Wikipedia vorgeschlagen wurden. Seiten dieses Typs lassen sich meist über die Verwendung einer speziellen Vorlage erkennen: in der deutschsprachigen Wikipedia zum Beispiel wird dieser Typ unter Anderem allen Seiten zugewiesen, die die Vorlage `{{Löschen}}` enthalten.

8.6. Konzeptklassifikation

Konzepten wird während des Imports ein `ConceptType` aus einer fest vorgegebenen Menge von Typen zugewiesen. Diese Konzepttypen sind unabhängig von der Sprache und dem Wiki (bzw. Korpus) und dienen einer groben Filterung der Konzepte je nach Anwendungsbereich, insbesondere für die *Named Entity Recognition* [Dakka und Cucerzan (2008), Toral und Munoz (2006)]. Sie werden von der Methode `WikiTextAnalyzer.determineConceptType` unter Verwendung der Sensoren in `WikiConfiguration.conceptTypeSensors` bestimmt. Die folgenden Konzepttypen sind definiert:

UNKNOWN: Unbekannter Typ — wird verwendet, wenn der Konzepttyp nicht bestimmt werden konnte, weil es keine beschreibende Ressource (also keine Wiki-Seite) zu dem Konzept gibt. Das kommt insbesondere für solche Konzepte vor, die im Wiki zwar referenziert, aber nicht erklärt werden (insbesondere also „rote Links“ und Navigationskategorien, ▹9.1).

PLACE: Orte, Regionen, geografische Einheiten. Solche Einträge könnten für den Aufbau bzw. die Erweiterung eines *Gazetteers* verwendet werden und sind besonders für Aufgaben wie *Named Entity Recognition* und damit *Topic-Tracking* nützlich. Beim Aufbau eines Wörterbuches dagegen können Orte gegebenenfalls ausgelassen werden.

PERSON: Natürliche Personen. Solche Einträge könnten für den Aufbau bzw. die Erweiterung eines *Personenregisters* verwendet werden und sind ebenfalls besonders für Aufgaben wie *Named Entity Recognition* und damit *Topic-Tracking* nützlich. Beim Aufbau eines Wörterbuches dagegen sollten Personen in der Regel nicht aufgenommen werden.

ORGANISATION: Organisationen wie Firmen, Regierungs- und Nichtregierungsorganisationen (NGOs), etc. Auch solche Einträge sind besonders für Aufgaben wie *Named Entity Recognition* und damit *Topic-Tracking* nützlich. Beim Aufbau eines Wörterbuches dagegen sollten Organisationen meist nicht aufgenommen werden.

NAME: Vor- oder Nachnamen an sich (nicht die betreffenden Personen). Solche Einträge könnten für den Aufbau bzw. die Erweiterung eines *Namenslexikons* verwendet werden und sind unter Umständen für Aufgaben wie *Named Entity Recognition* und damit *Topic-Tracking* nützlich. Beim Aufbau eines Wörterbuches dagegen können Namen im Allgemeinen ausgelassen werden.

TIME: Zeitperioden (wie z. B. 15. Jahrhundert) oder auch ein wiederkehrendes Datum (wie z. B. 6. April). Solche Einträge könnten für den Aufbau bzw. die Erweiterung eines *Almanachs* verwendet werden. Beim Aufbau eines Wörterbuches dagegen sollten sie nicht verwendet werden.

NUMBER: Zahlen an sich. Diese können beim Aufbau eines Wörterbuches in der Regel übergangen werden.

LIFEFORM: Lebensformen, also biologische *Taxa*, Klassen, Familien, Rassen usw. von Tieren und Pflanzen. Diese können beim Aufbau eines Wörterbuches unter Umständen ausgelassen werden.

OTHER: Sonstige Konzepte — alle, denen keiner der oben angegebenen Typen zugeordnet werden konnte.

ALIAS: Pseudo-Konzept für Aliase; wird nur aus technischen Gründen vorübergehend benötigt (▹A.6) und sollte im fertigen Thesaurus nicht vorkommen.

Die Zuordnung von Konzepttypen wird in ▹12.1 evaluiert.

8.7. Links

Wiki-Links (▹A.3) sind eine reiche Informationsquelle: sie bieten Zugang zu menschlichem Urteil darüber, welche Konzepte für das Verständnis eines anderen Konzeptes relevant sind (durch Querverweise, [WP:V] sowie [Zesch u. a. (2007a), Ponzetto und Strube (2007c), Gregorowicz und Kramer (2006)]), sowie welche Bezeichnung (Term) für welches Konzept geeignet ist (durch den Link-Text, [Mihalcea (2007)] sowie [Craswell u. a. (2001), Eiron und Mccurley (2003), Kraft und Zien (2004)]). Die Methode `WikiTextAnalyzer.extractLinks` extrahiert alle Wiki-Links aus dem gegebenen Text. Der Text sollte bereits mit `stripClutter` bereinigt worden sein. Der genaue Algorithmus ist in Anhang E.1 beschrieben.

Die `WikiLink`-Objekte, die dabei erzeugt werden, haben insbesondere die folgenden Eigenschaften (vergleiche ▹A.3):

interwiki: der Präfix für Interwiki-Links (`null`, falls der Link kein Interwiki-Link ist).

namespace: die ID des Namensraumes, in den der Link verweist, wie von `WikiTextAnalyzer.getNamespaceId` geliefert (siehe auch ▹B.7).

page: der Titel der Seite, auf die der Link zeigt (ohne Namensraum, Interwiki-Präfix, Abschnitt, etc.) in normalisierter Form (also mit Unterstrichen statt Leerzeichen und dem ersten Buchstaben als Großbuchstaben).

section: der Abschnitts-Anker (bzw. das „*Dokumentenfragment*“ nach der URL-Spezifikation) auf der Zielseite in normalisierter Form (oder `null`, wenn auf keinen Abschnitt verwiesen wird).

text: der verwendete Link-Text (Label). Falls nicht explizit ein Link-Text angegeben wurde, so wird der Text, der zur Angabe des Link-Zieles verwendet wurde, als Link-Text angenommen und gegebenenfalls der Link-Trail angehängt. Ist der Link ein Kategorisierungslink, hat also die Eigenschaft `magic` den Wert `CATEGORY`, so gibt die Eigenschaft `text` den *Sort-Key* für die Kategorisierung an.

impliedText: ist `true` genau dann, wenn der Text nicht explizit angegeben wurde, sondern aus dem Link-Ziel geschlossen wurde.

magic: gibt an, inwiefern der Link „magisch“ ist, dass heißt, ob er eine spezielle Semantik hat. Die möglichen Werte werden von `WikitextAnalyzer.LinkMagic` definiert:

NONE: „normaler“ Wiki-Link, keine „Magic“.

CATEGORY: Zuweisung einer Kategorie an die Seite, die den Link enthält.

IMAGE: Einbinden eines Bildes; da Bilder schon von `stripClutter` entfernt werden, sollte das hier nicht vorkommen.

LANGUAGE: Verknüpfung mit einer (semantisch) äquivalenten Seite in einer anderen Sprache.

Dabei ist zu beachten, dass z. B. ein Link in den Kategorie-Namensraum auch ohne die Kategorisierungssemantik vorkommen kann, nämlich genau dann, wenn er mit einem führenden Doppelpunkt definiert wurde, also in der Form *`[[:Category:Baum]]`*. Dasselbe gilt analog für Links in den Image-Namensraum und auch für Links mit Interwiki-Präfix (siehe auch ▹A.3).

Die so extrahierten Links dienen, je nach ihrer `magic`, zur Bestimmung der semantischen Verwandtheit von Konzepten (▹9.1.3), dem Zusammenführen von äquivalenten Konzepten aus unterschiedlichen Wikipedias (▹9.2.2), dem Aufbau der Subsumtionsrelation oder, unter Verwendung des Link-Textes, der Zuordnung von Termen zu Konzepten (▹9.1). Sie sind damit von zentraler Bedeutung für den Aufbau des Thesaurus, da sie nahezu alle betrachteten Relationen direkt oder indirekt definieren.

8.8. Extraktion von Glossen

Definitionssätze (*Glossen*) können in einem Artikel recht einfach gefunden werden: per Konvention [WP:WSIGA] definiert der erste Satz immer den Gegenstand des Artikels. Den

ersten Satz zu finden, ist allerdings nicht ganz trivial: aus dem bereinigten Text (nach `WikiTextAnalyzer.stripClutter`) wird zunächst der erste Absatz extrahiert, also der Text bis zur ersten leeren Zeile, oder der ersten Überschrift. Diese Arbeit erledigt die Methode `WikiTextAnalyzer.extractFirstParagraph`, welche auf `WikiConfiguration.extractParagraphMangler` zurückgreift. Per Default ist das eine speziell konfigurierte Instanz von `WikiTextAnalyzer.BoxStripMangler`, die nach Entfernen aller Blöcke (siehe ▹8.3) nur den ersten Absatz liefert — der Rest des Textes wird aus Effizienzgründen nicht weiter behandelt.

Der so gewonnene erste Absatz wird dann mittels `WikiTextAnalyzer.stripMarkup` weiter bereinigt, so dass nur noch reiner Text ohne Markup übrig bleibt. Zusätzlich werden weitere Elemente mittels `LanguageConfiguration.sentenceManglers` entfernt, per Default insbesondere alle geklammerten Teilsätze. Aus dem bereinigten ersten Absatz wird dann der erste Satz als Glosse extrahiert. Der Algorithmus für diese Aufgabe ist in Anhang E.2 detailliert beschrieben.

Bei den so extrahierten Sätzen handelt es sich fast immer um Definitionssätze der Form „*X ist/sind/war (ein) Y*“; allerdings kommt es vor, dass die Aussagekraft der Definition zu wünschen übrig lässt — zum Beispiel lautet der Einleitungssatz für den Artikel *Weltoffenheit*[1]: „*Weltoffenheit ist ein Begriff aus der philosophischen Anthropologie.*“ — dieser Satz beschreibt nicht, was Weltoffenheit *ist*, sondern lediglich, welcher Fachbereich sie untersucht. Aber selbst diese Art der Ungenauigkeit ist verhältnismäßig selten (▹12). Eine mögliche Lösung wäre es, den ganzen ersten Absatz als Definition zu verwenden oder zumindest zusätzlich vorzuhalten, wie es in der Literatur vorgeschlagen wurde (vergleiche die Arbeiten in ▹2.4, z. B. [Strube und Ponzetto (2006)]). Im Falle des Artikels *Weltoffenheit* wäre das zielführend, der Absatz lautet als Ganzes:

> *Weltoffenheit ist ein Begriff aus der philosophischen Anthropologie. Er bezeichnet die Entbundenheit des Menschen von organischen Zwängen (Trieben) und seiner unmittelbaren Umwelt und betont seine Öffnung hin zu einer von ihm selbst hervorgebrachten kulturellen Welt. Hiermit einher geht, dass der Mensch ohne festgelegte Verhaltensmuster geboren wird und sich Verhaltenssicherheit in der Welt immer erst erwerben muss.*

Der erste Absatz enthält aber fast immer auch Text, der nicht Teil der Definition des Artikelgegenstandes ist: der zweite Satz in obigem Beispiel vervollständigt die Definition, der dritte jedoch ist schon eine Einlassung, die über eine bloße Definition hinausgeht. Immer den gesamten Absatz zu verwenden würde also das *Signal-Rausch-Verhältnis* in Bezug auf die Forderung, eine Definition des Konzeptes zu finden (▹5.1), verschlechtern.

Das oben beschriebene Verfahren zur Extraktion von Glossen wird in ▹12.6 evaluiert.

8.9. Terme

Terme, die als Bezeichnungen für ein bestimmtes Konzept dienen, werden aus verschiedenen Quellen bestimmt, zum Beispiel aus dem Link-Text (siehe ▹8.7, vergleiche [Mihalcea (2007)]),

[1] In der deutschsprachigen Wikipedia, zum Zeitpunkt des Erstellens dieser Arbeit:
<http://de.wikipedia.org/w/index.php?title=Weltoffenheit&oldid=44584462>

aus Weiterleitungen (siehe ▹8.5) oder aus Begriffsklärungen (siehe ▹8.10). Aber auch aus dem Artikel selbst können schon Bezeichnungen gewonnen werden: das ist die Aufgabe der Methode `PlainTextAnalyzer.determinePageTerms` (bzw. `PlainTextAnalyzer.getPageTerms`). Sie bestimmt Bezeichnungen für das Konzept des Artikels per Default aus zwei Quellen:

1. Aus dem „Basisnamen“ des Artikels, wie er von `PlainTextAnalyzer.determineBaseName` zurückgegeben wird. Der „Basisname“ ist der Titel des Artikels, der mittels der Muster in `WikiConfiguration.titleSuffixPattern` und `WikiConfiguration.titlePrefixPattern` bereinigt wurde — per Default wird dabei insbesondere der geklammerte Suffix vom Titel entfernt, falls vorhanden: aus *Leiter_(Gerät)* wird also „*Leiter*“. Um die entsprechenden Terme zu erhalten, werden außerdem alle Unterstriche durch Leerzeichen ersetzt, aus *Albert_Einstein* wird also „*Albert Einstein*“.
2. Aus dem *Magic Word* `DISPLAYTITLE`, das eine alternative Bezeichnung für das Konzept definiert (siehe auch ▹A.6). Diese *Magic Words* werden bei der Extraktion der Vorlagen evaluiert (siehe ▹8.4).

Zusätzliche Terme können aus den *Sort-Keys* bestimmt werden, die bei der Kategorisierung der Seite verwendet wurden (zugänglich über die Methode `WikiPage.getCategories`) sowie aus der Verwendung des *Magic Words* `DEFAULTSORT` (zugänglich über die Methode `PlainTextAnalyzer.getDefaultSortKey`), siehe ▹A.3 und ▹A.6.

Der Analyzer definiert außerdem eine Methode zur Bestimmung „schlechter“ Terme, nämlich `WikiTextAnalyzer.isBadTerm`. Per Default ist das lediglich ein „Sanity-Check“ der Länge des Terms: diese wird mit `WikiConfiguration.minTermLength` und `WikiConfiguration.maxTermLength` verglichen.

8.10. Begriffsklärungsseiten

Begriffsklärungsseiten (Disambiguierungen) dienen in der Wikipedia dazu, die möglichen Bedeutungen eines Terms aufzuzählen [WP:BKL]. Um daraus eine Liste von Konzepten zu erzeugen, müssen aus dem Wiki-Text der Begriffsklärungsseite alle Wiki-Links (und damit alle Konzepte) ausgelesen werden, die auf einen Artikel verweisen, der eine der möglichen Bedeutungen beschreibt. Das Problem besteht darin, dass Begriffsklärungsseiten noch weitere Wiki-Links enthalten können, insbesondere Querverweise und Links auf Kontext-Begriffe (vergleiche die in ▹2.4 beschriebenen Arbeiten, insbesondere [Dakka und Cucerzan (2008), Zesch u. a. (2007a), Ponzetto und Strube (2007c)]).

Diejenigen Links auf der Begriffsklärungsseite zu finden, die auf die einzelnen Bedeutungen verweisen, ist Aufgabe der Methode `WikiTextAnalyzer.extractDisambigLinks`. Der Standard-Ansatz macht dabei zwei wesentliche Annahmen:

- die Begriffsklärungsseite enthält eine Liste von Bedeutungen derart, dass in jeder (relevanten) Zeile (höchstens) ein Link steht, der auf eine Bedeutung des zu klärenden Terms zeigt.

- enthält eine Zeile mehr als einen Link, so ist der Link mit der größten lexikographischen Ähnlichkeit zu dem zu klärenden Term auch der, der auf eine Bedeutung dieses Terms verweist.

Auf Basis dieser Annahmen wurde der Algorithmus zur Extraktion der Bedeutungslinks aus der Begriffsklärungsseite entwickelt, der in ▹E.3 beschrieben ist. Die Ergebnisse dieses Verfahrens werden in ▹13 evaluiert.

Auf diese Weise werden aus Wiki-Seiten diejenigen Eigenschaften extrahiert, die für den Aufbau eines (zunächst monolingualen) Thesaurus relevant sind. Diese Eigenschaften werden durch ein `WikiPage`-Objekt repräsentiert und von der Klasse `ConceptImporter` dann hinsichtlich ihrer Bedeutung für das Thesaurusmodell interpretiert (siehe ▹9.1). Die so gewonnenen Entitäten und Relationen werden in einem lokalen Datensatz abgelegt (siehe ▹C.1 und ▹C.4).

9. Verarbeitung

Im Kapitel Framework (▹7) wurde erklärt, wie WikiWord aufgebaut ist und wie der Import-Prozess als Ganzes abläuft, in Analyse des Wiki-Textes (▹8) wurde gezeigt, wie Informationen aus dem Wiki-Text extrahiert werden. Dieses Kapitel soll nun erläutern, wie aufgrund dieser Informationen ein Thesaurus aufgebaut werden kann.

9.1. Import von Wiki-Seiten

Der Import von Wiki-Seiten in den lokalen Datensatz (das *Konzeptmodell*) wird von der Methode `handlePage` in der Klasse `ConceptImporter` gesteuert: sie initiiert die Umwandlung der Wiki-Textes in ein Ressourcenobjekt vom Typ `WikiPage` (siehe ▹4.1 bzw. ▹8 und ▹D.1) und interpretiert anschießend die Eigenschaften des Ressourcenobjekts, um daraus die Relationen eines Thesaurus aufzubauen. Diese werden dann im lokalen Datenmodell abgelegt, das durch ein Datenzugriffsobjekt vom Typ `LocalConceptStoreBuilder` modelliert wird (siehe ▹C.4). Diese Interpretation der Ressourceneigenschaften als Informationen über die Konzepte, Terme und Relationen eines Thesaurus soll nun hier genauer erklärt werden.

Vorab ist zu bemerken, dass Relationen im Datenmodell zum Teil mit Metainformationen über ihre Herkunft versehen sind (vergleiche z. B. ▹C.2 und ▹C.2): insbesondere wird häufig die ID der Ressource (Wiki-Seite) gespeichert, aus der eine Beziehung abgeleitet wurde, sowie in einigen Fällen zusätzlich die Regel, nach der die Relation extrahiert wurde. Die Regeln sind in der Enumeration `ExtractionRule` definiert; sie werden in diesem Kapitel an den betreffenden Stellen genannt.

Bei der Interpretation der Ressourceneigenschaften werden unter anderem die folgenden Aspekte betrachtet:

Kategorisierung: Von den Wiki-Links der Seite (▹8.7) werden diejenigen betrachtet, deren `magic` Property den Wert `LinkMagic.CATEGORY` hat (▹8.7, ▹A.3). Diese Kategorisierungslinks auf der Seite werden als Angabe von Konzepten interpretiert, die dem Konzept, das von der Seite selbst repräsentiert wird, übergeordnet sind: sie sind also eine Manifestation der Subsumtionsrelation (vergleiche [WP:KAT] sowie die in ▹2.4 vorgestellten Arbeiten, insbesondere [Voss (2006), Ponzetto und Strube (2007b), Gregorowicz und Kramer (2006)]). Für jeden dieser Links wird die Methode `LocalConceptStore.storeConceptBroader` aufgerufen (▹C.4), um diese Zuordnung im lokalen Datenmodell festzuhalten (Regel `ExtractionRule.BROADER_FROM_CAT`).

Die Zuordnung von übergeordneten Konzepten wird unter Umständen im Nachbereitungsschritt noch überarbeitet (▹9.1.3), insbesondere werden Kategorisierungs-Aliase aufgelöst (siehe ▹9.1.2).

Ein Problem, das bei der Interpretation der Kategoriestruktur von Wikipedia auftreten kann, sind so genannte *Wartungskategorien*, die sich nicht auf den Inhalt (das

Konzept) der Seite beziehen, sondern auf die Seite selbst — zum Beispiel die Kategorie *Kategorie:Wikipedia:Lückenhaft* [Voss (2006)]. Solche Kategorien sind für einen Thesaurus nicht von Belang und können sogar irreführend sein, da sie inhaltliche Verwandtschaft suggerieren, wo keine solche existiert. WikiWord umgeht dieses Problem auf einfache Weise: Wartungskategorien werden so gut wie nie direkt einem Artikel zugewiesen, sie stammen fast immer aus einer Vorlage, die zur Markierung des Artikels verwendet wurde. Da WikiWord den Inhalt von Vorlagen aber nicht auswertet, tauchen Wartungskategorien in den von WikiWord gefundenen Wiki-Links gar nicht auf.

Eine weitere Art von Kategorie, die für einen Thesaurus von zweifelhaftem Wert sind, sind *Navigationskategorien* [Gregorowicz und Kramer (2006), Hammwöhner (2007)]. Das sind häufig sogenannte *Schnittmengenkategorien*, die solche Artikel enthalten, denen zwei Eigenschaften gemein sind, zum Beispiel *Kategorie:Fluss_in_Bayern* oder *Kategorie:Politiker_(19._Jahrhundert)*. Oder es handelt sich um *Metakategorien*, die Kategorien mit ähnlicher Struktur zusammenfassen, zum Beispiel *Kategorie:Fluss_nach_Staat* oder *Kategorie:Person_nach_Epoche*. Solche Navigationskategorien werden von WikiWord als vollwertige Konzepte behandelt — zwar sind sie als eigenständige Begriffe eher ungebräuchlich oder gar künstlich, dennoch liefern sie über ihre Zuordnung von über- und untergeordneten Konzepten Informationen über die semantischen Verwandtheitsbeziehungen und können überdies auch im Thesaurus der Navigation dienen.

Die Kategoriestruktur der Wikipedia soll zwar per Konvention (poly-)hierarchisch sein, diese Anforderung wird aber von MediaWiki nicht geprüft. So kann die Struktur insbesondere auch Zyklen enthalten [Hammwöhner (2007)]. Diese werden von WikiWord während der Nachbereitung entfernt (▹9.1.3).

Link-Text Terme: Es werden alle Wiki-Links der Seite betrachtet (▹8.7) und für jeden Link per `LocalConceptStore.storeReference` wird ein Eintrag im lokalen Datenmodell vorgenommen, der das Konzept, auf das der Link verweist, als Bedeutung mit dem verwendeten Link-Text verbindet (siehe auch ▹8.9). Die Verwendung eines Link-Textes für eine Referenz auf einen bestimmten Artikel wird also als Zuordnung einer Bezeichnung an das Konzept interpretiert, das der Ziel-Artikel beschreibt (vergleiche [WP:V, Mihalcea (2007)] sowie [Craswell u. a. (2001), Eiron und Mccurley (2003), Kraft und Zien (2004)]): zum Beispiel würde *`[[Vereinigte Staaten|USA]]`* dafür sorgen, dass der Term „*USA*“ dem Konzept Vereinigte_Staaten zugeordnet wird. Diese Extraktionsregel wird mit `ExtractionRule.TERM_FROM_LINK` identifiziert.

Links, deren `magic`-Property nicht `LinkMagic.NONE` ist oder die in einen anderen Namensraum oder gar auf ein anderes Wiki verweisen, werden dabei übergangen. Für alle anderen Link-Ziele wird zunächst angenommen, dass es Artikel sind, die Konzepte beschreiben, es sich also um Seiten mit dem Typ `ResourceType.ARTICLE` handele. Im Nachbereitungsschritt (▹9.1.3) werden dann Links auf Weiterleitungsseiten aufgelöst (`DatabaseLocalConceptStoreBuilder.finishAliases`), sowie alle Links auf andere Typen von Ressourcen, insbesondere auf Begriffsklärungen und als „schlecht“ markierte Seiten, entfernt (`DatabaseLocalConceptStoreBuilder.finishBadLinks`).

Neben der Nutzung der Zuordnung von Link-Text zu Link-Ziel zum Aufbau der Bedeutungsrelation dienen Wiki-Links auch dazu, den semantischen Kontext von Konzepten zu definieren (siehe auch ▹2.3): Querverweise zwischen Artikeln werden mit der Methode `LocalConceptStore.storeLink` gespeichert, die sowohl die Bedeutungszuweisung als auch die Assoziation von Konzepten berücksichtigt (vergleiche ▹C.2). Diese Information wird später vor allem zur Bestimmung der semantischen Verwandtheit verwendet (▹9.1.3).

Seitennamen: Eine Wiki-Seite selbst definiert bereits Terme, die als Bezeichnung für das Konzept der Seite dienen. Diesen Term zu bestimmen, ist Aufgabe von `WikiPage.getPageTerms` (siehe ▹8.9), Termzuordnungen auf dieser Basis werden mit `ExtractionRule.TERM_FROM_TITLE` identifiziert. Der erste Term wird aus dem Seitentitel bestimmt, gegebenenfalls bereinigt von einem Klammerzusatz. Falls nicht nur ein Titel, sondern ein ganzer Artikel zur Verfügung steht, liefert die Methode `WikiPage.getPageTerms` zusätzlich Terme, die mit Hilfe des *Magic Words* *DISPLAYTITLE* angegeben wurden (▹A.6).

Während der Verarbeitung werden die Konzeptnamen, Terme und Definitionstexte (Glossen) *Sanity Checks* unterzogen, um zu verhindern, dass fehlerhafte Daten in den Thesaurus aufgenommen werden. Im Einzelnen werden geprüft:

1. Die Länge von Konzeptnamen und Ressourcennamen. Sie muss zwischen den Werten liegen, die durch `WikiConfiguration.minNameLength` und `WikiConfiguration.maxNameLength` liegen (gemessen in Zeichen) — Terme und Namen, die zu kurz oder zu lang sind, werden übergangen. Per Default liegt der Minimalwert bei 1 und der Maximalwert bei 128. `DatabaseLocalStoreBuilder` verlangt zusätzlich, dass Namen in UTF-8-kodierter Form [UNICODE (2007), s. 103] nicht länger als 255 Byte sein dürfen, sonst werden sie abgeschnitten[1]. Diese Beschränkung ergibt sich aus den technischen Vorgaben der Datenbankstruktur (▹C.1).

2. Die Länge von Termen. Es gelten dieselben Beschränkungen wie für Konzeptnamen, nur dass gegen die Werte von `WikiConfiguration.minTermLength` und `WikiConfiguration.maxTermLength` geprüft wird. Auch die Begrenzung auf maximal 255 Byte effektive Länge gilt für Terme.

3. Die Länge von Definitionstexten. `DatabaseLocalStoreBuilder` verlangt, dass der Name in UTF-8-kodierter Form nicht länger sein als 8192 Byte (8KB) sein darf — längere Texte werden abgeschnitten.

4. Terme und Konzeptnamen dürfen kein Markup enthalten, sonst werden sie übergangen — geprüft wird das mit Hilfe eines `WikiTextSniffer`-Objektes. Die Prüfung ist insbesondere darauf ausgelegt, auch Markup-Teile zu erkennen, die aus fehlerhafter Verwendung der Wiki-Text-Syntax herrühren. So kann es vorkommen, dass Konzepte und Terme mit

[1] Dabei ist zu beachten, dass jedes Zeichen in UTF-8 bis zu vier Bytes benötigt. Für lateinische Buchstaben wird jedoch nur ein Byte pro Zeichen benötigt, für Umlaute und andere in Europa übliche diakritische Zeichen typischerweise zwei.

legitimem, aber „verdächtigem“ Inhalt übergangen werden — das ist jedoch recht selten. Definitionstexte, die Markup enthalten, sind erlaubt, lösen aber eine Warnung aus.

Die Interpretation der Eigenschaften der Ressource, repräsentiert durch ein `WikiPage`-Objekt, hängt stark vom Typ der Ressource ab, wie er von `WikiTextAnalyzer` bestimmt wurde (siehe ▹8.5). Die Verarbeitung beginnt in jeden Fall damit, dass in der Tabelle `resource` ein Eintrag für diese Ressource (bzw. Wiki-Seite) gespeichert wird. Das weitere Vorgehen ist für die einzelnen Ressourcetypen wie folgt definiert:

Kategorieseiten (`CATEGORY`): Für Kategorieseiten wird lediglich der Kategorisierungsaspekt der Seite betrachtet, wie oben beschrieben.

Zu beachten ist dabei, dass für die Kategorie zunächst kein Konzepteintrag angelegt wird, obwohl Kategorien grundsätzlich als Konzepte interpretiert werden. Der Grund ist, dass es eine Artikelseite geben könnte, die dieses Konzept beschreibt (siehe dazu ▹9.1.2). Existiert keine solche Artikelseite, so wird im Nachbereitungsschritt `finishMissingRessources` ein Konzepteintrag vom Typ `ConceptType.UNKNOWN` für das Kategorie-Konzept erstellt (▹9.1.3), allerdings erst nachdem alle Kategorisierungsaliase aufgelöst wurden (▹9.1.2).

Eine weitere Information in Kategorieseiten, deren Verarbeitung vielversprechend ist, sind die Language-Links. Sie referenzieren Kategorien in anderen Wikis (also in anderen Sprachen) und könnten die Verknüpfungen zwischen den Artikeln unterschiedlicher Sprachen ergänzen [Hammwöhner (2007b)]. Die Auswertung dieser Information wurde hier jedoch aufgrund technischer Schwierigkeiten nicht weiter verfolgt und verbleibt als Gegenstand zukünftiger Forschung.

Artikelseiten (`ARTICLE`): Da Artikelseiten per Definition stets ein Konzept beschreiben, wird zuerst die Methode `LocalConceptStore.storeConcept` aufgerufen, um einen Eintrag für das betreffende Konzept anzulegen. Der volle Seitentitel dient dabei als eindeutiger Bezeichner des Konzeptes. Dann wird die Definition des Konzeptes (die Glosse) aus dem Artikeltext extrahiert (▹8.8) und mittels `LocalConceptStore.storeDefinition` im lokalen Datenmodell abgelegt.

Als nächstes werden alle Wiki-Links der Seite bestimmt (▹8.7). Auf dieser Grundlage werden die Link-Text-Terme gespeichert und die Kategorisierung bestimmt, wie oben beschrieben. Zusätzlich werden noch die folgenden Aspekte betrachtet:

1. Querverweise zwischen Artikeln werden als *Referenzbeziehung* bzw. *Assoziation* zwischen Konzepten interpretiert. Daher wird beim Speichern der Link-Text-Terme statt der Methode `LocalConceptStore.storeReference` die Methode `LocalConceptStore.storeLink` verwendet, die, zusätzlich zu der Beziehung zwischen Link-Text und Ziel-Konzept, eine Beziehung zwischen dem im Artikel beschriebenen Quell-Konzept und dem Ziel-Konzept herstellt (vergleiche ▹C.2). Diese Daten dienen unter anderem der Berechnung der semantischen Verwandtheit zwischen Termen (▹9.1.3).
2. Wiki-Links können auch auf *Teile* von Seiten verweisen — solche Abschnitte, die als Verweis-Ziele dienen, werden als eigene Konzepte interpretiert, die dem Konzept, das vom Gesamtartikel beschrieben wird, untergeordnet sind (▹9.1.1).

3. Bei der Kategorisierung angegebene *Sort-Keys* werden verwendet, um dem Konzept, auf das sich der Artikel bezieht, per `LocalConceptStore.storeReference` weitere Terme zuzuordnen (`ExtractionRule.TERM_FROM_SORTKEY`). So würde zum Beispiel die Angabe *`[[Kategorie:Pazifist|Einstein, Albert]]`* auf der Seite *Albert_Einstein* dafür sorgen, dass der Term „*Einstein, Albert*“ dem Konzept ALBERT_EINSTEIN zugeordnet wird (▹A.3). Der *Sort-Key* wird ebenfalls berücksichtigt, wenn er über das Magic Word *`DEFAULTSORT`* angegeben wurde (zugänglich über die Methode `PlainTextAnalyzer.getDefaultSortKey`, siehe auch ▹A.6).
4. Bei der Bestimmung der Kategorisierung des Artikels kann der Artikel als *Hauptartikel* der betreffenden Kategorie erkannt werden, also als der Artikel, der das Konzept definiert, auf das sich die Kategorie bezieht: siehe ▹9.1.2. In diesem Fall wird zusätzlich der Name der Kategorie als Term für das Konzept des Artikels gespeichert (`ExtractionRule.TERM_FROM_CAT_NAME`).
5. Language-Links, also Wiki-Links, die den `magic`-Wert `LinkMagic.LANGUAGE` haben, werden als Verweise auf eine Beschreibung desselben oder eines ähnlichen Konzeptes in einer anderen Sprache interpretiert (siehe ▹A.3 sowie [WP:INTERLANG]). Diese Verweise werden mittels `LocalConceptStore.storeLanguageLink` im lokalen Datenmodell abgelegt. Sie dienen der Berechnung der semantischen Ähnlichkeit zwischen Termen sowie dem Zusammenführen von sprachspezifischen zu sprachübergreifenden Konzepten (▹9.2.2).

Begriffsklärungsseiten (`DISAMBIG`): Zunächst werden die Link-Text-Terme auf der Seite interpretiert, wie oben beschrieben.

Dann werden alle Disambiguierungslinks bestimmt (▹8.10), also Verweise auf Artikel, die eine der Bedeutungen definieren, die zum Titel der Begriffsklärungsseite passen. Jeder dieser Bedeutungen wird nun mittels `LocalConceptStore.storeReference` der Name der Begriffsklärungsseite als Bezeichnung zugewiesen (`ExtractionRule.TERM_FROM_DISAMBIG`). Der Name der Begriffsklärungsseite wird also als Bezeichnung für diejenigen Konzepte interpretiert, die die Begriffsklärung als mögliche Bedeutungen auflistet. Das entspricht genau der Intention von Begriffsklärungsseiten, wie sie in [WP:BKL] beschrieben ist (vergleiche auch [STRUBE UND PONZETTO (2006), GREGOROWICZ UND KRAMER (2006)]).

Weiterleitungsseiten (`REDIRECT`): Über die Methode `WikiPage.getRedirect` wird die Zielseite der Weiterleitung bestimmt. Mittels `LocalConceptStore.storeReference` wird dann der Name der Weiterleitungsseite als Bezeichnung für das Konzept registriert, auf das die Weiterleitung zeigt (`ExtractionRule.TERM_FROM_REDIRECT`). Dabei kann es durchaus vorkommen, dass der Name der Weiterleitung eigentlich für ein Konzept steht, das dem Zielkonzept untergeordnet ist — das kann aber bei der Interpretation von Link-Text genauso geschehen; auch scheint diese Art von Fehlern keinen allzu problematischen Einfluss auf das Ergebnis zu haben, siehe ▹12.5. Häufig tritt dieser Fall zum Beispiel für Stadtteile auf, die keinen eigenen Artikel haben, sondern in dem Artikel beschrieben werden, der sich mit der Stadt als Ganzes befasst.

Zusätzlich wird mittels `LocalConceptStore.storeConcept` ein vorläufiger Konzepteintrag vom Typ `ALIAS` angelegt und per `LocalConceptStore.`

`storeConceptAlias` als Alias für das Zielkonzept registriert. Der vorläufige Konzepteintrag ist aus technischen Gründen notwendig, er wird in der Nachbereitungsphase von der Methode `LocalConceptStore.finishAliases` zur Auflösung von Referenzen auf die Weiterleitungsseite verwendet und dann gelöscht (siehe ▸9.1.3).

Listen (`LIST`): Für Listen werden lediglich die Link-Text-Terme interpretiert.

Konzeptionell könnten aus Listenartikeln auch Einträge für die Subsumtionsrelation gefolgert werden: Einträge in einer *Liste von* X würden dann als X *untergeordnet* interpretiert. Dabei ergeben sich allerdings zwei Probleme:

1. Das übergeordnete Konzept, zu dem die Liste gehört, müsste namentlich identifiziert und gegebenenfalls mit der Liste assoziiert werden. Die Extraktion des Konzeptnamens aus dem Listentitel wäre für einfache Fälle machbar, z. B. für Titel der Form *Liste_der_X* — bei manchen Titeln würde das aber fehlschlagen, z. B. bei der Seite *Comicformat*, einer Auflistung unterschiedlicher Comic-Formate. Eine Zuordnung zu dem entsprechenden Artikel, der dieses Konzept beschreibt, kann aber bei mehrdeutigen Namen schwierig sein.
2. Listenartikel sind sehr unterschiedlich formatiert. Einfache Aufzählungen könnten leicht verarbeitet werden, Listenartikel sind aber häufig in Abschnitte gegliedert. Zum Teil sind Listen auch als Tabellen formatiert, bei denen die Bedeutung einzelner Spalten nicht automatisch zu erkennen ist (z. B. bei den Seiten *Marientitel* und *Präfixe_von_Schiffsnamen*[2]). Manchmal sind die Auflistungen hierarchisch, mit Über- und Untereinträgen. Alles in allem gibt es, anders als bei Begriffsklärungsseiten, keine einheitliche Struktur für Listenartikel, so dass es schwierig ist, sie sinnvoll zu verarbeiten.
3. Ein Problem mit der Erkennung von Listenartikeln ist, dass zum Teil Seiten als Listen klassifiziert werden, die selbst keine Liste *sind*, sondern eine Liste *beschreiben*, zum Beispiel *Rote_Liste_gefährdeter_Arten*. In manchen Fällen trifft sogar beides zu: der Artikel beschreibt eine Liste *und* enthält die Liste selbst — was eine Unterscheidung vollends unmöglich macht.

Da es parallel zu einem Listenartikel in der Regel auch eine Kategorie mit ähnlichem Umfang gibt, lohnt sich der Versuch, die Listenartikel zu interpretieren, kaum. In der vorliegenden Implementation von WikiWord wird deshalb die Verknüpfungsstruktur von Seiten, die als Liste erkannt wurden, nicht untersucht.

Schlechte Seiten (`BAD`): Seiten, die zur Löschung vorgeschlagen wurden [WP:LR] oder anderweitig als problematisch markiert sind, werden ignoriert.

9.1.1. Konzepte aus Abschnitten

MediaWiki erlaubt (genau wie HTML) Verknüpfungen auf „Anker“ [W3C:HTML (1999)], also auf bestimmte Stellen *innerhalb* eines Artikels bzw., in URI-Terminologie, auf Dokument-Fragmente

[2] Beides nebenbei auch Beispiele für schwer zu interpretierende Titel.

[RFC:3986 (2005)]. In MediaWiki erzeugt jede Überschrift einen Anker, es ist also möglich, Wiki-Links direkt auf einzelne Abschnitte von Artikeln zu setzen (siehe auch ▹8.7 und ▹A.3). Solche Verknüpfungen auf Abschnitte werden vor allem dann verwendet, wenn auf ein spezielles Konzept verwiesen werden soll, das in diesem Abschnitt beschrieben ist und keinen eigenen Artikel hat; dabei handelt es sich in der Regel um ein dem Thema des Gesamtartikels untergeordnetes Konzept. So könnte z. B. der Wiki-Link *[[Myanmar#Geschichte|Burmesische Geschichte]]* auftreten: er verweist unter dem Term „*Burmesische Geschichte*“ auf das Konzept MYANMAR#GESCHICHTE, welches im Abschnitt *Geschichte* im Artikel *Myanmar* behandelt wird — wir gehen davon aus, dass es sich dabei um ein Konzept handelt, das auf irgendeine Art ein „Teil“ des Konzeptes MYANMAR ist, diesem also über die Subsumtionsrelation untergeordnet ist.

Allerdings ist es schwierig, über ein solches Konzept, das aus einem Seitenabschnitt entstanden ist, mehr zu sagen, als dass es dem Konzept, das der Gesamtartikel beschreibt, untergeordnet ist: ein solcher Abschnitt enthält in der Regel keinen Definitionssatz und er kann auch nicht einzeln kategorisiert werden; selbst die Zuordnung von Templates wäre schwierig. So wird auf eine weitere Analyse des Abschnitts verzichtet: Konzepte, die sich aus Verweisen auf Abschnitte ergeben, werden lediglich vorgemerkt (▹C.2) und dann in der Nachbereitung des Imports als Konzepteinträge von unbekanntem Typ (`ConceptType.UNKNOWN`) erzeugt und ihrem Oberkonzept zugeordnet (▹9.1.3).

9.1.2. Kategorien und Konzepte

Wie oben beschrieben, versteht WikiWord Kategorien zwar als Konzepte in so fern, als dass die Kategorisierung als eine Unterordnung eines Konzepts unter ein anderes im Sinne der Subsumtionsrelation interpretiert wird. Jedoch erzeugt WikiWord für Kategorie-Seiten zunächst keinen eigenen Konzepteintrag, da es einen Artikel geben könnte, der das Konzept näher beschreibt — ein solcher Artikel wird *Hauptartikel* der Kategorie genannt.

Es ist nun wünschenswert, die Kategorie und ihren Hauptartikel als Repräsentationen desselben Konzeptes zu interpretieren und ihre Eigenschaften zu vereinigen. Dabei sind drei Fälle zu unterscheiden:

1. Es gibt einen Artikel (also eine Seite mit dem Typ `ARTICLE`), der genau den gleichen Titel hat wie die Kategorie (ohne Namensraum-Präfix, aber gegebenenfalls mit Klammerzusatz). Dann geht WikiWord davon aus, dass dieser Artikel der Hauptartikel der Kategorie ist, das heißt, dass sie sich auch auf dasselbe Konzept beziehen. Da der Titel als eindeutiger Bezeichner für das Konzept dient, zeigen in diesem Fall alle Verknüpfungen der Kategorie automatisch auf das betreffende Konzept, es ist keine weitere Anpassung notwendig.

2. Es gibt einen Hauptartikel für die Kategorie, aber er hat einen anderen Titel. Dies ist der kritische Fall, der unten näher betrachtet wird.

3. Es wurde kein zu der Kategorie passender Artikel gefunden. In diesem Fall wird während der Nachbereitung im Schritt `finishMissingConcepts` ein „leeres“ Konzept vom Typ `ConceptType.UNKNOWN` für die Kategorie erstellt. Die einzige Information, die über dieses

Konzept bekannt ist, ist seine Position in dem durch die Subsumtionsrelation beschriebenen Unterordnungsgraphen. Das ist besonders für Navigationskategorien der Fall, kann aber auch dann auftreten, wenn die Heuristik zur Bestimmung eines passenden Artikels fehlschlägt oder ein solcher Artikel einfach fehlt.

Der kritische Fall ist nun der, dass es zwar einen Hauptartikel für die Kategorie gibt, der Titel dieses Artikels aber von dem Titel der Kategorie verschieden ist. Besonders häufig kommt dies in Projekten vor, die per Konvention für Kategorien den Plural verwenden, während für Artikel der Singular benutzt wird — das ist zum Beispiel in der englischsprachigen Wikipedia der Fall. Ansonsten tritt dieses Problem oft dann auf, wenn der Titel des Hauptartikels einen Klammerzusatz trägt, die Kategorie aber nicht (die Seite mit dem entsprechenden Titel ohne Klammerzusatz im Hauptnamensraum ist dann in der Regel eine Begriffsklärungsseite).

Es gilt nun einerseits, diesen Fall zu erkennen, also Artikel korrekt als Hauptartikel von Kategorien zu identifizieren und entsprechend zuzuordnen, und andererseits, die durch die Unterschiedlichkeit der Titel verursachte Inkonsistenz in den Relationen zu beseitigen: das bedeutet, den Titel der Kategorie als Alias für das Konzept des Hauptartikels zu behandeln.

Die Heuristik, um Hauptartikel von Kategorien zu erkennen, basiert auf den zur Kategorisierung verwendeten Sort-Keys: Per Konvention ist der Hauptartikel einer Kategorie dieser direkt zugeordnet und zwar unter Verwendung eines speziellen Sort-Keys, der dafür sorgt, dass der Artikel in der Kategorie ganz vorne, vor allen anderen Seiten aufgeführt wird; konkret werden dafür einzelne Sonderzeichen verwendet, die in der *Kollationsreihenfolge* vor den alphanumerischen Zeichen stehen. Welche Sort-Keys als Markierung für Hauptartikel interpretiert werden, wird von `WikiConfiguration.mainArtikeMarkerPattern` festgelegt — per Default wird der reguläre Ausdruck `^[- !_*$@#+~/%]?$` verwendet [WP:KAT].

Allerdings erhält nicht unbedingt ausschließlich der Hauptartikel einer Kategorie so einen speziellen Sort-Key: es können weitere Artikel auf diese Weise markiert sein, die für die Kategorie besonders wichtig sind oder anderweitig vom „normalen“ Inhalt der Kategorie abgesetzt werden sollen, zum Beispiel *Frühmittelalter* und *Spätmittelalter* in *Kategorie:Mittelalter*. Daher werden nur solche Artikel als Hauptartikel in Betracht gezogen, deren Titel dem Titel der Kategorie lexikographisch ausreichend ähnlich ist: die beiden Titel werden von Klammerzusätzen und Großschreibung befreit und dann ihre Levenshtein-Ähnlichkeit[3] bestimmt. Nur wenn der Ähnlichkeitswert über dem Wert liegt, den `WikiConfiguration.maxWordFormDistance` festlegt (per Default $1/3$), wird der Artikel als Hauptartikel der Kategorie in Betracht gezogen. Dieser Vergleich ist in der Methode `WikiTextAnalyer.mayBeFormOf` implementiert.

Ein Artikel wird dementsprechend (vorläufig) als Hauptartikel einer Kategorie angesehen, wenn er der Kategorie unter Verwendung eines speziellen Sort-Keys zugeordnet ist und sein Titel dem Titel der Kategorie lexikographisch hinreichend ähnlich ist. In diesem Fall wird die Zuordnung per `LocalConceptStoreBuilder.storeConceptAlias` registriert, unter Verwendung des `scope`-Wertes `AliasScope.BROADER`, um anzuzeigen, dass es sich um einen Alias bezüglich der Kategorisierung handelt: so wird ein *Kategorisierungsalias* definiert.

[3] Die normierte Levenshtein-Distanz nach [Levenshtein (1966)], wie durch `DefaultLinkSimilarityMeasure.calculateLevenshteinSimilarity` implementiert; vergleiche ▸E.3

Um nun die Eigenschaften der Kategorie — nämlich ihre Einordnung in die Subsumtionsrelation — auf den Konzepteintrag des Hauptartikel zu übertragen, werden im Nachbereitungsschritt `finishMissingConcpets` in der Subsumtionsrelation sämtliche Referenzen auf die Kategorie durch Referenzen auf das Konzept des Hauptartikels ersetzt.

Bevor die Kategorisierungsaliase so angewendet und aufgelöst werden können, müssen sie zunächst noch bereinigt werden: im Schritt `finishBadLinks` der Nachbereitung werden einige Kategorisierungsaliase wieder gelöscht, und zwar nach folgenden Regeln:

1. Ein Kategorisierungsalias ist zu löschen, wenn es einen Artikel gibt, der den exakt gleichen Titel wie die betreffende Kategorie hat — dieser Artikel ist dann automatisch der Hauptartikel der Kategorie, ohne dass eine Auflösung von Aliasen notwendig wäre, selbst dann, wenn es andere Artikel gibt, die als Hauptartikel in Betracht kämen. Das ist zum Beispiel für *Kategorie:Mittelalter* der Fall: *Frühmittelalter* und *Spätmittelalter* sind mögliche Hauptartikel der Kategorie, da jedoch der Artikel *Mittelalter* existiert, wird dieser als Hauptartikel bestimmt und alle Kategorisierungsaliase für MITTELALTER werden gelöscht.

2. Ein Kategorisierungsalias ist zu löschen, wenn es mehr als einen Artikel gibt, der als Hauptartikel in Betracht käme. In diesem Fall sind die Konzepte der einzelnen Artikel vermutlich dem Konzept der Kategorie *untergeordnet* und die Kategorie sollte somit einen eigenen Konzepteintrag erhalten.

   Zum Beispiel ist das für *Kategorie:Arbeit* der Fall: die Seite *Arbeit* existiert zwar, ist aber eine Begriffsklärung[4]; Kandidaten für Hauptartikel der Kategorie sind *Arbeit_(Sozialwissenschaften)*, *Arbeit_(Philosophie)* und *Arbeiter*. Da die Zuweisung eines Hauptartikels aber nicht eindeutig ist, werden alle Kategorisierungsaliase für ARBEIT wieder gelöscht und ARBEIT als eigenes Konzept eingeführt, das den Konzepten ARBEIT_(SOZIALWISSENSCHAFTEN), ARBEIT_(PHILOSOPHIE) und ARBEITER übergeordnet ist.

Für eine Evaluation der hier beschriebenen Heuristiken siehe ▸12.2.

9.1.3. Nachbereitung

Die Nachbereitung des Konzeptimports dient dazu, die gewonnenen Daten zu bereinigen und zu konsolidieren. Das erfolgt vor allem durch Operationen direkt auf der Datenbank, eine programmatische Verarbeitung der Daten wird weitgehend vermieden. Die Nachbereitung ist in die folgenden Schritte gegliedert, die in `LocalConceptStoreBuilder` definiert und in `DatabaseLocalConceptStoreBuilder` implementiert sind:

`finishBadLinks`: Dieser Schritt dient dazu, Verknüpfungen zu löschen, die für die Erstellung eines Thesaurus nutzlos oder irreführend wären — das sind insbesondere solche Links oder Kategorisierungen, die auf eine Seite zeigen, die kein Konzept repräsentiert — also vor allem Begriffsklärungsseiten. Verweise auf Begriffsklärungsseiten können nicht als

[4] Dass der Titel einer Kategorie mit dem Titel einer Begriffsklärung zusammenfällt, kommt recht häufig vor.

Beziehungen zwischen Konzepten (oder zwischen Termen und Konzepten) interpretiert werden, da unklar ist, welches Konzept als Ziel gemeint ist. Daher ist es besser, sie vollständig zu ignorieren.

Neben solchen „ungültigen“ Wiki-Links werden in diesem Schritt auch unerwünschte Kategorisierungen, Weiterleitungen und Kategorisierungsaliase (▹9.1.2) gelöscht. Im einzelnen bedeutet das:

1. Es werden aus der Tabelle `link` (▹C.2) alle Einträge gelöscht, deren `target_name` auf einen Eintrag in der Tabelle `resource` (▹C.2) verweist, der einen der folgenden Typen hat: `ResourceType.DISAMBIG`, `ResourceType.LIST` oder `ResourceType.BAD`. Verweise auf Ressourcen mit dem Typ `ResourceType.REDIRECT` bleiben zunächst erhalten, diese werden im Schritt `finishAliases` aufgelöst.
2. Aus der Tabelle `alias` (▹C.2) werden ebenfalls die Einträge gelöscht, die über `target_name` auf eine „schlechte“ Ressource verweisen, nur dass hier zusätzlich zu den Ressourcen mit den Typen `ResourceType.DISAMBIG`, `ResourceType.LIST` oder `ResourceType.BAD` auch die mit Typ `ResourceType.REDIRECT` als „schlecht“ betrachtet werden — auf diese Weise werden Weiterleitungen auf Weiterleitungen entfernt; solche mehrfachen Weiterleitungen werden von MediaWiki nicht unterstützt und sind daher als Fehler in der Datenbasis anzusehen.
3. Aus der Tabelle `alias` werden alle die Einträge gelöscht, deren `scope` `AliasScope.BROADER` ist und deren Feld `source_name` auf ein existierendes Konzept verweist. So werden alle Kategorisierungsaliase für diejenigen Kategorien entfernt, für die es einen Hauptartikel mit gleichem Namen gibt (vergleiche ▹9.1.2).
4. Aus der Tabelle `alias` werden außerdem all die Einträge gelöscht, deren `scope` `AliasScope.BROADER` ist und deren Feld `source_name` nicht eindeutig ist, also mehrfach vorkommt. So findet keine Zuweisung eines Hauptartikels statt, wenn mehrere Artikel als Hauptartikel der gleichen Kategorie in Frage kommen (vergleiche ▹9.1.2).
5. Aus der Tabelle `section` (▹C.2) werden alle Einträge gelöscht, deren `concept_name` auf einen Eintrag in der Tabelle `resource` verweist, der einen der folgenden Typen hat: `ResourceType.DISAMBIG`, `ResourceType.LIST` oder `ResourceType.BAD`. Das bedeutet, dass Konzepte aus Abschnitten (▹9.1.1) nur für „richtige“ Artikelseiten definiert sind.
6. Aus der Tabelle `section` werden alle Einträge gelöscht, deren Feld `concept_name` nicht auf ein existierendes Konzept verweist. Das bedeutet, dass Wiki-Links ignoriert werden, die auf einen Abschnitt in einem Artikel verweisen, den es gar nicht gibt.

`finishSections`: Dieser Schritt dient der Verarbeitung der Artikelabschnitte, die als Link-Ziele vorkommen und daher als Konzepte modelliert werden sollen (vergleiche ▹9.1.1). Für jeden Abschnitt wird ein Konzepteintrag angelegt und dem Konzept des Artikels, von dem der Abschnitt ein Teil ist, im Sinne der Subsumtionsbeziehung untergeordnet. Konkret bedeutet das:

1. Für jeden Eintrag in der Tabelle `section` (▹C.2) wird ein Eintrag in der Tabelle `concept` (▹C.2) angelegt, wobei `concept.name` auf den Wert von `section.concept_name` gesetzt wird und `concept.type` auf `ConceptType.UNKNOWN`.
2. Für jeden Eintrag in der Tabelle `section` wird ein Eintrag in der Tabelle `broader` (▹C.2) vorgenommen, so dass der Wert des Feldes `broader.broad_name` dem Wert von `section.concept_name` entspricht und der Wert des Feldes `broader.narrow_name` dem von `section.section_name`. Die verwendete Regel wird als `ExtractionRule.BROADER_FROM_SECTION` angegeben.

`finishMissingConcpets`: In diesem Schritt werden Einträge für Konzepte erzeugt, auf die es Referenzen gibt, zu denen aber kein beschreibender Artikel existiert. Das sind vor allem Einträge für Wiki-Links auf noch nicht existierende Artikel („rote“ Links) sowie Einträge für Kategorien, die keinen Hauptartikel haben, also vornehmlich Navigationskategorien (vergleiche ▹9.1.2). Für solche Konzepte ohne eigene Wiki-Seite wird ein Eintrag mit dem Typ `ConceptType.UNKNOWN` angelegt. Im Einzelnen:

1. Für jeden Wert von `link.target_name` (▹C.2), der nicht auf ein existierendes Konzept zeigt, wird ein Konzepteintrag mit dem entsprechenden Namen in der Tabelle `concept` angelegt.
2. Als Vorbereitung für den nächsten Teilschritt werden nun die Kategorisierungsaliase aufgelöst: In der Tabelle `broader` (▹C.2) werden alle Werte in den Feldern `broad` und `broad_name` bzw. `narrow` und `narrow_name`, für die es einen passenden Eintrag in der Tabelle `alias` (▹C.2) mit einem `scope` von `AliasScope.BROADER` gibt, mit den entsprechenden Werten von `alias.target` und `alias.target_name` ersetzt.
3. Für jeden Wert der Felder `broad_name` sowie `narrow_name` in der Tabelle `broader` (▹C.2), der nicht auf ein existierendes Konzept zeigt, wird ein Konzepteintrag angelegt. Das sind vor allem Navigationskategorien, zu denen es keinen entsprechenden Artikel gibt.

`finishIdReferences`: In vielen Fällen werden in der Datenstruktur von WikiWord Referenzen redundant über den Namen eines Konzeptes sowie die ID des Konzepts geführt (zum Beispiel in den Tabellen `link` (▹C.2) und `broader` (▹C.2)). Einer der Gründe dafür ist, dass so Referenzen gespeichert werden können, ohne dass die laufende ID für das betreffende Konzept bekannt sein oder ermittelt werden muss — das Feld, dass sich auf die ID bezieht, bleibt dann vorerst `NULL`. In diesem Schritt der Nachbereitung werden nun die fehlenden Werte in den Referenzfeldern, die sich auf die Konzept-ID beziehen, nachgetragen. Im Einzelnen:

1. In der Tabelle `link` werden fehlende IDs in der Spalte `target` anhand der Konzeptnamen in der Spalte `target_name` nachgetragen. Die Zuordnung von IDs zu Namen wird der Tabelle `concept` entnommen.
2. In der Tabelle `broader` werden fehlende IDs in den Spalten `broad` bzw. `narrow` anhand der Konzeptnamen in den Spalten `broad_name` bzw. `narrow_name` nachgetragen.

3. In der Tabelle `alias` (▹C.2) werden fehlende IDs in der Spalte `target` anhand der Konzeptnamen in der Spalte `target_name` nachgetragen.

Es ist auch möglich, alle Relationen direkt mit numerischen IDs aufzubauen und damit den zusätzlichen Aufwand dieses Nachbereitungsschritts zu vermeiden: Eine Abbildung von Konzeptnamen zu IDs wird dafür im Arbeitsspeicher gehalten (und zusätzlich in einer Datei gespeichert, um nach einer Unterbrechung eine Fortsetzung des Imports mit Hilfe des *Agenda*-Objekts zu erlauben).

Dieser Modus kann über den *Tweak*-Wert `dbstore.idManager=true` aktiviert werden, wobei darauf zu achten ist, dass der Java VM über den Parameter `-Xmx` ausreichend Platz auf dem Heap zugewiesen wurde, um die ID-Map im Arbeitsspeicher zu halten: Für jeden Eintrag in der Map ist mit einem Overhead von mindestens 150 Byte zu rechnen, in der Praxis hat es sich als sinnvoll herausgestellt, inklusive aller Puffer und interner Strukturen mit insgesamt einem Kilobyte Speicherbedarf pro Konzept zu rechnen, wobei alle „unbekannten“ (artikellosen) Konzepte mitgezählt werden. Für die Verarbeitung der englischsprachigen Wikipedia mit acht Millionen Konzepten ergibt sich damit ein Speicherbedarf von bis zu acht Gigabyte[5]. Wird keine ID-Map im Speicher gehalten, so sind 512 Megabyte für den Import von Wikis aller Größen ausreichend.

`finishAliases`: In diesem Schritt werden Aliase aufgelöst (vergleiche ▹A.6 und ▹1.2) — das heißt, in sämtlichen Tabellen werden Referenzen auf solche Konzepte vom Typ `ConceptType.ALIAS` durch diejenigen ersetzt, auf die der betreffende Alias-Eintrag in der Tabelle `alias` (▹C.2) verweist. Im Einzelnen:

1. In der Tabelle `link` werden Aliase für Werte in den Spalten `target` und `target_name` aufgelöst: Einträge, die zu Werten in `alias.source` bzw. `alias.source_name` passen, werden durch die entsprechenden Werte von `alias.target` bzw. `alias.target_name` ersetzt.
2. In der Tabelle `broader` werden die Spalten `narrow` und `narrow_name` sowie `broad` und `broad_name` entsprechend den Einträgen in der `alias` Tabelle angepasst.
3. Aus der Tabelle `concept` (▹C.2) werden alle Einträge mit dem Typ `ConceptType.ALIAS` gelöscht.
4. Aus der Tabelle `link` werden alle Einträge gelöscht, deren Feld `target` auf kein existierendes Konzept mehr verweist — das heißt solche Links, die auf fehlerhafte Weiterleitungen verwiesen.
5. Für jeden Wert der Felder `broad_name` sowie `narrow_name` in der Tabelle `broader` (▹C.2), der nicht auf ein existierendes Konzept zeigt, wird ein Konzepteintrag angelegt. Diese Wiederholung aus dem Schritt `finishMissingConcpets` dient unter anderem dazu, Konzepte neu zu erzeugen, die gelöscht wurden, weil sie als Alias markiert waren, die aber als Teil der Subsumtionsrelation noch benötigt werden[6].

[5] Zur Speicherung der Abbildung wird die Standard-Klasse `java.util.HashMap` verwendet. Der Platzaufwand ließe sich eventuell durch die Verwendung einer geeigneten Implementation eines Trie oder DAWG reduzieren (vergleiche [Heyer u. a. (2006), s. 68]).

[6] Das tritt nur ein, wenn es sich um einen „schlechten“ Alias gehandelt hat, z. B. einen, der auf eine Begriffsklärung zeigte und deshalb keinen Eintrag in der der `alias`-Tabelle hatte. Ansonsten wurden die entsprechenden Einträge in der `broader`-Tabelle bereits angepasst.

finishRelations: In diesem Schritt werden semantische Verwandtheit und Ähnlichkeit zwischen Konzepten bestimmt und in die Tabelle `relation` (▹C.2) eingetragen. Genauer:

1. Alle Paare von Konzepten, die einen gemeinsamen Language-Link haben (die also Einträge in der Tabelle `langlink` (▹C.2) haben, die bezüglich der Felder `language` und `target` übereinstimmen), werden im Feld `langmatch` markiert. Solche Konzepte werden als *ähnlich* angesehen, da Language-Links per Konvention immer auf Artikel mit gleichem oder sehr ähnlichem Gegenstand verweisen (vergleiche ▹9.2.2 und [WP:INTERLANG]). Wie viele Konzepte es gibt, die in diesem Sinne als ähnlich erkannt werden, hängt stark von der *Granularität* des Wikis ab: in einem Wiki mit hoher Granularität (also vielen spezialisierten Einträgen) ist die Wahrscheinlichkeit hoch, dass mehrere spezielle Artikel auf denselben, allgemeineren Artikel in einem anderen Wiki verweisen, wenn das andere Wiki eine geringere Granularität hat (siehe ▹12.3 für die Evaluation).

   Bei der Bestimmung der Ähnlichkeit von *globalen*, sprachübergreifenden Konzepten wird zusätzlich das Feld `langref` verwendet (siehe ▹9.2.3). Es modelliert die Ähnlichkeit von Konzepten, die sich gegenseitig *direkt* über einen Language-Link referenzieren (siehe ▹9.2.2). Die so bestimmte Ähnlichkeit von Termen wird in ▹12.3 evaluiert.

2. Alle Paare von Konzepten, die sich gegenseitig referenzieren (die also jeweils einen Eintrag in der Tabelle `link` haben, der auf das andere Konzept verweist), werden im Feld `bilink` markiert. Solche Konzepte werden als *verwandt* angesehen: wenn ein Konzept für die Erläuterung eines anderen relevant ist und daher in dem entsprechenden Artikel erwähnt und verlinkt wird, kann das als inhaltliche *Assoziation* verstanden werden. Diese Art der Assoziation ist allerdings oft zu allgemein: zum Beispiel wird in sehr vielen Artikeln auf Maßeinheiten und Jahreszahlen verwiesen, ohne dass ein inhaltlicher, thematischer Zusammenhang zu den Maßen oder Jahren bestünde — die Verknüpfungen dienen lediglich der Navigation. Ist die Assoziation aber *gegenseitig*, ist also jeweils das eine Konzept relevant für die Erläuterung des anderen, so ist es wahrscheinlich, dass beide Konzepte zu einem gemeinsamen Thema gehören und eine (unspezifizierte) semantische Beziehung zueinander haben [Gregorowicz und Kramer (2006)]. Die so bestimmte Verwandtheit von Termen wird in ▹12.4 evaluiert.

Eine weitere interessante Informationsquelle für die Bestimmung von semantischer Verwandtheit zwischen Konzepten wäre sicherlich die Untersuchung von (signifikanten) Kookkurrenten bezüglich der Linkstruktur sowie von Ähnlichkeiten der Kookkurrenzmengen, also der Untersuchung von Kookkurrenten höherer Ordnung [Biemann u. a. (2004), Mahn und Biemann (2005)]. Dieser Möglichkeit wurde im Rahmen der vorliegenden Arbeit aufgrund des zusätzlichen Aufwands an Programmlogik, Rechenzeit und Speicherplatz nicht nachgegangen. Im Rahmen des Wortschatzprojektes [Quasthoff (1998), Wortschatz] hat *Matthias Richter* Versuche zur Kookkurrenzanalyse der Verknüpfungsstruktur der deutschsprachigen Wikipedia durchgeführt[7].

[7] Keine Publikation bekannt, eine experimentelle Web-Schnittstelle zu den Ergebnissen befindet sich auf <`http://wortschatz.uni-leipzig.de/WP/`>.

buildTermsForMissingConcepts: In diesem Schritt, der im Gegensatz zu den anderen aus technischen Gründen in der Klasse `ConceptImporter` implementiert ist, wird für jedes Konzept mit dem Typ `ConceptType.UNKNOWN` (also jedem Konzepteintrag, zu dem es keinen Artikel gibt) der Titel-Term bestimmt und gespeichert. Genauer wird der „Basisname“ des Konzeptes bestimmt (▹8.9) und dieser dann per `LocalConceptStore.storeReference` in die Tabelle `link` eingetragen.

finishMeanings: In diesem Schritt wird die *Bedeutungsrelation* zwischen Termen und Konzepten berechnet. Diese stellt ein Kernstück des Thesaurus dar, das dazu dient, die *Semantic Gap* zwischen Text und Inhalt zu überbrücken [Gregorowicz und Kramer (2006), Mihalcea (2007), Gabrilovich und Markovitch (2006)]. Die Termzuordnung über die Bedeutungsrelation wird in ▹12.5 evaluiert.

Die Bedeutungsrelation wird durch die Tabelle `meaning` (▹C.2) repräsentiert. Diese Tabelle wird nun befüllt, indem die Einträge in der Tabelle `link` über die Felder `target` und `term_text` gruppiert und die Einträge pro Gruppe gezählt werden: so wird bestimmt, welcher Term wie oft als Bezeichnung für welches Konzept verwendet wird.

Dabei wird für jede Bedeutungszuordnung, also für jedes Paar von Werten für `target` und `term_text`, das Gruppen-Maximum des Codes im Feld `rule` mit in die Tabelle `meaning` übernommen. Da der Code für die Regel `ExtractionRule.TERM_FROM_LINK` kleiner ist als die Codes der anderen Regeln, lässt sich so bestimmen, ob eine Bedeutungszuweisung ausschließlich aufgrund von Link-Texten vorgenommen wurde. Der Hintergrund ist, dass explizite Zuordnungen von Termen zu Konzepten, wie sie durch Seitentitel, Weiterleitungen, Begriffsklärungen etc. definiert werden, zuverlässiger sind als solche, die ausschließlich auf Link-Texten beruhen (▹12.5). Es kann also nützlich sein, für Bedeutungszuordnungen, die nur auf Link-Texten beruhen, zusätzliche Bedingungen zu definieren, zum Beispiel eine Mindestfrequenz von zwei oder drei Wiederholungen (▹10.4).

finishFinish: Dies ist der letzte Schritt der Nachbereitung, er dient dazu, letzte Inkonsistenzen zu beseitigen. Er besteht aus den folgenden Operationen:

1. Entfernen aller Schleifen aus der Verknüpfungsrelation, also alle Einträge aus der Tabelle `link`, deren `anchor`- und `target`-Felder denselben Wert haben.
2. Entfernen aller Schleifen aus der Subsumtionsrelation, also alle Einträge aus der Tabelle `broader`, deren `broad`- und `narrow`-Felder denselben Wert haben.
3. Entfernen aller Zyklen aus dem Graphen, der durch die Subsumtionsrelation definiert ist. Hierfür wird der in Anhang E.5 beschriebene Algorithmus verwendet. Dies ist notwendig, da die Kategoriestruktur der Wikipedia zwar per Konvention ein zusammenhängender, azyklischer Graph ist [WP:KAT], die MediaWiki Software diese Eigenschaften aber nicht erzwingt, Zyklen in der Struktur also durchaus auftreten können [Hammwöhner (2007)]. Die Semantik der Subsumtionsrelation verlangt aber, dass sie eine *Halbordnung* bildet und somit azyklisch sein muss.

Das Ergebnis der Nachbereitung ist ein vollständiger *lokaler Datensatz*, der einen monolingualen Thesaurus repräsentiert. Die Konzepte und Relationen in diesem Thesaurus werden in ▹12 evalu-

iert. Der nächste Abschnitt beschreibt, wie die lokalen Datensätze für unterschiedliche Sprachen zu einem multilingualen Thesaurus zusammengefasst werden können.

9.2. Zusammenführen des multilingualen Thesaurus

Das Zusammenführen der einzelnen lokalen Datensätze zu einem globalen Datensatz ist die Aufgabe des Programms `BuildThesaurus`. Die einzelnen Schritte der Zusammenführung sind in `DatabaseGlobalConceptStoreBuilder` implementiert und werden im Folgenden näher erläutert.

Zur schnelleren Verarbeitung in globalen Datensätzen wird jedem lokalen Datensatz (bzw. jeder Sprache) eine Zahl zwischen 1 und 32 zugewiesen (mehr als 32 Sprachen können nicht zusammengefasst werden). Entsprechend diesem Code ist jeder Sprache auch eine Bit-Maske zugeordnet, in der genau das Bit mit dieser Nummer gesetzt ist - die Sprache mit dem Code 1 hätte also Maske 1, die Sprache mit dem Code 2 hätte die Maske 2, die Sprache mit dem Code 3 hätte die Maske 4, die Sprache mit dem Code 4 hätte die Maske 8 und so weiter — dieser Wert wird in dem Feld `language_bits` in der Tabelle `concept` gespeichert (▹C.2).

9.2.1. Import der Konzepte

Zunächst müssen die Basisdaten aus den lokalen Datensätzen in das globale Datenmodell übernommen werden, damit sie gemeinsam betrachtet und verarbeitet werden können. Dabei wird zunächst jedes lokale Konzept aus jedem lokalen Datensatz als eigener globaler Konzepteintrag in den globalen Datensatz aufgenommen. Dazu werden für jeden lokalen Datensatz die folgenden Schritte ausgeführt, die direkt auf der Datenbank implementiert sind:

`importOrigins`: In diesem Schritt wird für jedes Konzept aus dem lokalen Datensatz in der Tabelle `origin` (▹C.2) des globalen Datensatzes ein Eintrag angelegt. Dieser Eintrag weist dem Konzept aus dem lokalen Datensatz eine ID zu, die für einen Eintrag in der Tabelle `concept` des globalen Datensatzes verwendet wird. Die Tabelle `origin` assoziiert also ein Paar aus Sprachcode und lokaler ID mit der ID eines globalen Konzepteintrags — diese Abbildung wird im Folgenden immer dann benutzt, wenn zwischen dem lokalen und dem globalen Datensatz eine Verbindung hergestellt werden muss.

`importConcepts`: In diesem Schritt werden die globalen Konzepteinträge für die Tabelle `concept` erzeugt (▹C.2). Dafür wird die in der Tabelle `origin` definierte globale Konzept-ID verwendet, der Konzepttyp (▹8.6) wird aus dem lokalen Datensatz übernommen, wie auch der Name des Konzeptes, dem allerdings noch der Sprachcode vorangestellt wird, um ihn eindeutig zu machen. Dabei ist zu beachten, dass der Name der globalen Konzepte keinerlei Bedeutung für die Verarbeitung oder Interpretation hat — er wird lediglich zur Kontrolle mitgeführt.

importLanglinks: Um die Language-Links der Konzepte aus verschiedenen lokalen Datensätzen vergleichen zu können, werden sie in diesem Schritt in die Tabelle `langlink` des globalen Datensatzes übernommen (▹C.2). Die IDs der lokalen Konzepte werden dabei über die `origin` in die IDs der globalen Konzepteinträge umgewandelt.

Die in den globalen Datensätzen importierten Language-Links verweisen nun zum Teil auf Konzepte, die ebenfalls in den globalen Datensatz aufgenommen wurden. Um diese Zuordnung korrekt auswerten zu können, müssen aber auch auf dieser Relation Aliase ausgewertet werden. Für jeden lokalen Datensatz werden deshalb die folgenden Schritte ausgeführt:

resolveLanglinkAliases: In der Tabelle `langlink` (▹C.2) werden für diejenigen Einträge, deren `target`-Feld zu dem Sprachcode des gerade betrachteten lokalen Datensatzes passt, die Werte in der Spalte `target` anhand der Tabelle `alias` (▹C.2) aus diesem lokalen Datensatz angepasst. Zum Beispiel wird während der Verarbeitung der Daten für den en-Datensatz (also des englischsprachigen Thesaurus) der Language-Link von DE:3-SAT auf EN:3-SAT angepasst und auf EN:BOOLEAN_SATISFIABILITY_PROBLEM geändert, da in der englischsprachigen Wikipedia die Seite *3-SAT* eine Weiterleitung auf *Boolean_satisfiability_problem* ist.

deleteAliasedLanglinks: In diesem Schritt werden aus der Tabelle `langlink` sämtliche Einträge entfernt, die noch auf ein Konzept verweisen, das in der Tabelle `alias` als Alias vermerkt ist. Damit werden doppelte Weiterleitungen ignoriert.

9.2.2. Aufbau der sprachübergreifenden Konzepte

Ein Hauptgegenstand dieser Arbeit sowie eine der Kernaufgaben von WikiWord ist der Aufbau von sprachunabhängigen Konzepten. Praktisch besteht die Aufgabe darin, Gruppen von äquivalenten Konzepten aus unterschiedlichen Sprachen zu identifizieren und zu einem sprachunabhängigen Konzept zusammenzuführen. Die folgenden Postulate bilden dabei die Grundlage für den Vereinigungsalgorithmus:

1. Die Language-Links in einem Artikel verweisen auf Artikel in einer anderen Sprache, die dasselbe oder zumindest ein ähnliches Konzept zum Gegenstand haben [WP:INTERLANG]. Wenn zwei Artikel per Language-Link auf denselben dritten Artikel verweisen, so beschreiben daher auch diese beiden Artikel ein ähnliches, oder das gleiche Konzept. Konzepte, die aus diesen Gründen als ähnlich gelten können, sollten im Thesaurus auch durch eine entsprechende Relation miteinander verknüpft sein. Die Qualität von Language-Links wurde in [HAMMWÖHNER (2007B)] näher untersucht.

2. Pro Sprache kann es in einem Artikel nur einen (ausgehenden) Language-Link geben[8]. Umgekehrt jedoch können mehrere Artikel aus einer Sprache auf denselben Artikel in einer

[8] Per Konvention. Diese Einschränkung wird von MediaWiki nicht erzwungen, jedoch werden mehrere Language-Links für dieselbe Sprache auf einer Seite nicht korrekt unterstützt.

anderen Sprache verweisen, es kann also pro Artikel mehrere *eingehende* Language-Links pro Sprache geben. Das ist besonders dann häufig der Fall, wenn die beiden Wikipedias in dem betreffenden Bereich sehr unterschiedliche Granularität besitzen, meist bedingt durch die Gesamtgröße der Wikipedia in der jeweiligen Sprache.

3. Referenzieren sich zwei Artikel aus unterschiedlichen Sprachen *gegenseitig* über Language-Links, so beschreiben sie *dasselbe* Konzept, oder zumindest eines, das in Bezug auf die Ziele dieser Arbeit hinreichend ähnlich ist, um als äquivalent angesehen zu werden. Diese beiden Artikel sollten also nach Möglichkeit demselben sprachunabhängigen Konzept zugewiesen werden.

   Dieses Postulat ist die Hauptvoraussetzung für die Effektivität des vorgeschlagenen Algorithmus. Es ist analog zu der in ▹9.1.3 genutzten Annahme, dass zwei Artikel, die sich gegenseitig referenzieren, verwandte Konzepte beschreiben, wie von [Gregorowicz und Kramer (2006)] vorgeschlagen.

4. Es gibt pro Sprache nur einen Artikel für jedes Konzept[9] — zwei Artikel aus derselben Wikipedia sollten also niemals demselben sprachunabhängigen Konzept zugewiesen werden.

5. Sollten zwei Artikel derselben Sprache aufgrund der gegenseitigen Verknüpfung mit Artikeln in anderen Sprachen als äquivalent gelten, so wird diese Situation als *Konflikt* bezeichnet (siehe Evaluation in ▹11.3) — diese Situation kann dann auftreten, wenn die folgende Verweisstruktur von Language-Links besteht: $A_1 \leftrightarrow B \leftrightarrow C \leftrightarrow A_2$. Dann stehen A_1 und A_2 in Konflikt zueinander: sie können als *sehr ähnlich* gelten, dürfen aber nicht demselben sprachunabhängigen Konzept zugewiesen werden.

6. Sprachunabhängige Konzepte werden als eine Menge von sprachspezifischen Konzepten modelliert, die so ausgewählt sind, dass sie sich untereinander möglichst ähnlich, idealerweise äquivalent sind; jedes sprachunabhängige Konzept enthält aus jeder Sprache höchstens ein sprachspezifisches Konzept. Die Eigenschaften und Beziehungen der sprachunabhängigen, globalen Konzepte ergeben sich aus der Vereinigung der Eigenschaften und Beziehungen der sprachspezifischen Konzepte.

Vor dem eigentlichen Zusammenführen von Konzepten müssen zunächst Paare von ähnlichen Konzepten aus unterschiedlichen Sprachen identifiziert werden. Die Methode `prepareGlobalConcepts` füllt dazu die Tabelle `relation` (▹C.2) in den folgenden zwei Schritten:

`buildLangMatch`: In die Spalte `langmatch` der Tabelle `relation` wird für Paare von Konzepten, die *gemeinsame* Language-Links haben, ein Eintrag eingefügt. Diese (schwache) Art von Ähnlichkeit wird zwar für den fertigen Thesaurus, nicht jedoch für das Vereinigen von Konzepten verwendet.

[9] Wieder per Konvention [WP:RED]. Es kommt vor, dass zwei Seiten zum selben Thema angelegt werden, das wird aber innerhalb der Wikipedia als zu behebender Fehler angesehen und ist selten genug, um für die Zwecke dieser Arbeit nicht ins Gewicht zu fallen.

buildLangRef: In die Spalte `langref` der Tabelle `relation` wird für Paare von Konzepten ein Eintrag eingefügt, wenn das eine Konzept über einen Language-Link direkt auf das andere verweist. Ist diese Beziehung gegenseitig, existiert also ein Eintrag dieser Art sowohl für das Paar (A, B) wie auch für das Paar (B, A), so wird, wie oben erklärt, davon ausgegangen, dass diese Artikel dasselbe Konzept beschreiben. Diese Information wird dann für das Zusammenfassen von lokalen zu sprachunabhängigen Konzepten verwendet.

Der Aufbau der sprachübergreifenden Konzepte geschieht dann dadurch, dass Paare von Konzepten, die sich gegenseitig über Language-Links referenzieren aber keine Überlappung in den Sprachen haben, die sie bereits abdecken, sukzessive vereinigt werden, so lange, bis es kein solches Paar mehr gibt. Dieser Algorithmus ist in Anhang E.4 detailliert beschrieben, die Ergebnisse werden in ▸11.3 analysiert.

Zum Vergleich wäre es interessant, alternative Algorithmen für das Finden und Zusammenfassen von „äquivalenten“ Konzepten zu erproben. Eine mögliche Alternative zur Nutzung von gegenseitigen Language-Links, wie sie oben beschrieben ist, wäre es, die Language-Links eines Artikels als *Featurevektor* zu betrachten [Salton u. a. (1975)]. Die Idee dabei ist, kontinuierlich diejenigen Konzeptgruppen miteinander zu vereinigen, die die meisten Language-Links gemeinsam haben, aber sich bezüglich der von ihnen abgedeckten Sprachen nicht überlappen. Der Algorithmus wäre also ähnlich wie der oben vorgestellte, nur dass statt gegenseitiger Referenzierung über Language-Links die *Ähnlichkeit* bezüglich des durch die Language-Links definierten Featurevektoren entscheidend wäre. Dieser Ansatz konnte hier jedoch aufgrund seiner höheren Komplexität und des größeren Rechenaufwands nicht umgesetzt werden. Seine Implementation und Untersuchung verbleibt als Gegenstand zukünftiger Forschung.

9.2.3. Nachbereitung

Nachdem die Konzepte in das globale Datenmodell importiert und zusammengeführt wurden, verbleibt noch die Aufgabe, die verschiedenen Beziehungen aus den einzelnen lokalen Datensätzen ebenfalls in den globalen Datensatz zu überführen und zu konsolidieren. Die Methode `finishGlobalConcepts` importiert zunächst aus jedem lokalen Datensatz die verbleibenden relevanten Relationen, im Einzelnen:

1. Die Tabelle `link` wird unter Verwendung der ID-Abbildung in der Tabelle `origin` aus dem lokalen Datensatz in den globalen Datensatz kopiert. Dabei wird lediglich die Information übernommen, welches Konzept auf welches verweist, nicht aber der Link-Text, da dieser ja sprachspezifisch ist.
2. Ebenso unter Verwendung der Tabelle `origin` wird die Tabelle `broader`, also die Subsumtionsrelation, übernommen. Zyklen, die eventuell durch die Vereinigung mit den Subsumtionsrelationen aus anderen lokalen Datensätzen entstehen, werden im nächsten Schritt aufgelöst.

Auf diese Weise werden die verschiedenen Beziehungen aus den einzelnen lokalen Datensätzen zusammengeführt. Nachdem alle Relationen so importiert wurden, führt `finishGlobalConcepts` schließlich die folgenden Schritte aus:

1. Entfernen aller Schleifen aus der Verknüpfungsrelation, also alle Einträge aus der Tabelle `link` (▹C.2), deren `anchor`- und `target`-Felder denselben Wert haben.

2. Entfernen aller Schleifen aus der Subsumtionsrelation, also alle Einträge aus der Tabelle `broader` (▹C.2), deren `broad`- und `narrow`-Felder denselben Wert haben.

3. Entfernen aller Zyklen aus dem Graphen, der durch die Subsumtionsrelation definiert ist. Hierfür wird der in Anhang E.5 beschriebene Algorithmus verwendet. Es ist notwendig, dies nach dem Zusammenführen der Subsumtionen aus den einzelnen lokalen Datensätzen noch einmal zu tun, da dabei ja wieder Zyklen entstanden sein können.

   In [Hammwöhner (2007b)] wird vorgeschlagen, die Kategoriestruktur verschiedener Wikipedias zu vergleichen und auf diese Weise Fehler in den Kategorisierungsstrukturen zu erkennen und zu beseitigen. Ein entsprechendes Verfahren könnte an dieser Stelle ohne weiteres eingefügt werden, diese Möglichkeit weiter zu untersuchen, geht jedoch über den Rahmen dieser Arbeit hinaus und bietet sich als Gegenstand künftiger Forschung an.

4. Wie schon zuvor für die einzelnen lokalen Datensätze, wird nun die Verwandtheit von Konzepten bestimmt: Alle Paare von Konzepten, die sich gegenseitig referenzieren (die also jeweils einen Eintrag in der Tabelle `link` haben, der auf das andere Konzept verweist), werden im Feld `bilink` der Tabelle `relation` (▹C.2) markiert.

5. Die Spalte `langref` in der Tabelle `relation` wurde bereits in ▹9.2.2 für Konzepte befüllt, die sich über einen Eintrag in der Tabelle `langlink` (▹C.2) referenzieren. In diesem Schritt wird diese bislang asymmetrische Beziehung nun symmetrisch ergänzt: für jeden Eintrag mit $langref(A, B) > 0$ wird $langref(B, A) := langref(B, A) + langref(A, B)$ gesetzt. Dabei ergibt sich genau für jene Paare, die sich gegenseitig referenzieren, ein Wert > 1 — das sind genau solche Paare, deren Konzepte sich *sehr ähnlich* sind, die aber nicht verschmolzen wurden, weil sie zueinander in Konflikt stehen (siehe ▹9.2.2). Für Paare mit $langref(A, B) = 1$ wird eine einfache Ähnlichkeit angenommen (▹11.3).

9.3. Aufbau der statistischen Daten

Das Programm `BuildStatistics` dient dazu, statistische Daten über einen (lokalen oder globalen) Datensatz zu erstellen. Insbesondere werden dabei die Verteilungen der Knotengrade im Term-Konzept-Graphen berechnet. Die Routinen zur Berechnung dieser Werte sind in `DatabaseLocalConceptStoreBuilder.DatabaseLocalStatisticsStoreBuilder` bzw. `DatabaseGlobalConceptStoreBuilder.DatabaseGlobalStatisticsStoreBuilder` in der Methode `buildStatistics` implementiert. Diese Statistiken sind einerseits für die Evaluation von WikiWord nützlich, andererseits dienen sie aber auch der Gewichtung der Konzepte nach verschiedenen Kriterien für unterschiedliche Aufgaben. Im Einzelnen werden die folgenden Statistiken erstellt:

indegree, der Knotengrad bezüglich eingehender Verweise, wird aus der Tabelle `link` (▹C.2) bestimmt und im Feld `in_degree` der Tabelle `degree` (▹C.2) gespeichert; der Rang bezüglich dieses Wertes wird im Feld `in_rank` abgelegt.

outdegree, der Knotengrad bezüglich ausgehender Verweise, wird aus der Tabelle `link` bestimmt und im Feld `out_degree` der Tabelle `degree` gespeichert; der Rang bezüglich dieses Wertes wird im Feld `out_rank` abgelegt.

linkdegree, der Knotengrad bezüglich der Summe von eingehenden und ausgehenden Verweisen, wird aus der Tabelle `link` bestimmt und im Feld `link_degree` der Tabelle `degree` gespeichert; der Rang bezüglich dieses Wertes wird im Feld `link_rank` abgelegt.

idf ist die *Inverse Document Frequency* bezüglich der Anzahl der Konzepte, die über die Tabelle `link` auf ein gegebenes Konzept verweisen. Dieser Wert ist, angelehnt an [Salton und McGill (1986)], definiert als

$$idf(c_i) = log\left(\frac{|C|}{indegree(c_i)}\right)$$

Der IDF-Wert dient zur Abschätzung der *Selektivität* eines Konzeptes: Konzepte mit einem hohen IDF-Wert sind stark selektiv, also eher spezialisiert, während Konzepte mit niedrigem IDF-Wert eher allgemeiner Natur sind. Die Verknüpfung eines Konzeptes mit einem anderen stark selektiven Konzept sagt also mehr über die thematische Zusammengehörigkeit der Konzepte aus, als eine Verknüpfung mit einem schwach selektiven Konzept. Der Rang jedes Konzeptes bezüglich seines IDF-Wertes wird in der Spalte `idf_rank` abgelegt.

lhs ist der *Local Hierarchy Score* nach [Muchnik u. a. (2007)] bezüglich eingehender und ausgehender Verweise, wie sie durch die Tabelle `link` definiert sind. Dieser Wert ist definiert als

$$lhs(c_i) = \frac{indegree(c_i) \cdot \sqrt{indegree(c_i)}}{indegree(c_i) + outdegree(c_i)}$$

Dabei wird die von [Muchnik u. a. (2007)] vorgeschlagene symmetrische Ergänzung nicht verwendet, da diese eine Anpassung für ungerichtete Graphen darstellt. Der LHS-Wert dient zur Abschätzung der *Zentralität* eines Konzeptes, also des Grades, zu dem es zur *Konnektivität* des gesamten Verweisnetzwerkes beiträgt, soweit sich das lokal bestimmen lässt: Konzepte mit einem hohen LHS-Wert sind eher allgemeiner Natur, sie eignen sich zur Kategorisierung bzw. als Zentren thematischer Cluster. Der Rang jedes Konzeptes bezüglich seines LHS-Wertes wird in der Spalte `lhs_rank` abgelegt.

freq, die Termfrequenz für die einzelnen Terme, wird aus der Tabelle `link` bestimmt und im Feld `freq` der Tabelle `term` gespeichert; der Rang bezüglich dieses Wertes wird im Feld `rank` abgelegt. Der Rang eines Terms kann auch als eine eindeutige ID für diesen Term verwendet werden (vergleiche [Heyer u. a. (2006), s. 59]).

Für die Knotengrade sowie die Termfrequenzen wird eine Verteilung gemäß dem Zipfschen Gesetz [Zipf (1935)] erwartet (und in ▹11.4 bzw. ▹11.5 auch bestätigt). In Hinblick darauf werden in der Methode `DatabaseWikiWordConceptStoreBuilder.buildDistributionStats` für jede dieser Verteilungen die folgenden Werte erhoben und in der Tabelle `stats` (▹C.2) abgelegt:

***total distinct types*:** Die Anzahl der „Typen“, also unterschiedlicher Terme (für die Verteilung der Termfrequenzen) bzw. Konzepte (für die Verteilung der Knotengrade), auf die sich die Verteilung bezieht, im Folgenden t.

***total token occurrences*:** Die Anzahl der „Tokens“ N, also die Summe der Verwendungen bzw. Referenzen V der einzelnen Terme bzw. Konzepte x_i:

$$N = \sum_i V(x_i)$$

***average type value*:** Die durchschnittliche Anzahl der Verwendungen jedes Typs (also jedes Terms bzw. Konzeptes) $\bar{v}$:

$$\bar{v} = \frac{N}{t}$$

***average type value x rank*:** Der durchschnittliche Wert des Rang-Wert-Produkts, k: Sei $R(x)$ der Rang des Typs x, und $V(x)$ die Anzahl der Verwendungen von x, so ist dieser Wert gegeben durch

$$k = \frac{\sum_i R(x_i) \cdot V(x_i)}{t}$$

Nach dem Zipfschen Gesetz wird nun erwartet, dass für alle x_i gilt:

$$R(x_i) \cdot V(x_i) \approx k$$

***characteristic fitting*:** Die normierte durchschnittliche Anzahl der Verwendungen jedes Typs:

$$c = \frac{k}{N} = \frac{\sum_i R(x_i) \cdot V(x_i)}{t \cdot N}$$

Dieser Wert ist der charakteristische *Anpassungskoeffizient* des Korpus.

***deviation from char. fit.*:** Die Standardabweichung des Rang-Wert-Produktes:

$$d_k = \sqrt{\frac{\sum_i \left(c - \frac{R(x_i) \cdot V(x_i)}{N}\right)^2}{t}} = \sqrt{\frac{\sum_i (k - R(x_i) \cdot V(x_i))^2}{t \cdot N^2}}$$

Dieser Wert gibt Auskunft darüber, wie genau die Verteilung dem Zipfschen Gesetz folgt.

Die einzelnen Verteilungen und ihre statistischen Eigenschaften für konkrete Daten werden in ▸11.4 und ▸11.5 diskutiert.

9.4. Vorbereitung von Konzeptdaten für den schnellen Zugriff

Das Programm `BuildConceptInfo` bereitet Konzeptdaten für einen schnellen Zugriff vor. Der Hintergrund ist, dass zu den Informationen, die für die Konzepte im Thesaurus relevant sind, vor allem ihre Relationen zu anderen Konzepten und zu Termen gehören. Diese Informationen manifestieren sich in Form von Listen, zum Beispiel in den Listen aller übergeordneter oder aller ähnlicher Konzepte. Bei einer rein relationalen Speicherung der Beziehungen zwischen den Konzepten ist für jede derartige Liste für jedes Konzept eine eigene Datenbankabfrage notwendig — das ist sehr aufwändig, besonders wenn diese Eigenschaften für mehr als ein Konzept abgefragt werden sollen.

Die Lösung für dieses Problem besteht darin, diese Listen im Vorhinein für jedes Konzept zu berechnen und serialisiert als einzelne Werte in der Datenbank abzulegen (als „*Property Cache*“). Gespeichert werden diese vorausberechneten Listen in den Tabellen `concept_info` und `concept_description` (siehe ▹C.2 und ▹C.2). Die Berechnung ist implementiert in den Klassen `DatabaseLocalConceptStoreBuilder.DatabaseLocalConceptInfoStoreBuilder` bzw. `DatabaseGlobalConceptStoreBuilder.DatabaseGlobalConceptInfoStoreBuilder` in der Methode `buildConceptInfo`.

Die jeweilige Liste von Konzepten oder Termen hat für jeden Eintrag mehrere Werte, die bei der Serialisierung berücksichtigt werden müssen: Möglich sind *ID*, *Name*, *Frequenz* und *Relevanz* — für jede Eigenschaft ist im Voraus bekannt, welche dieser Werte verwendet werden. Zu serialisieren ist also eine Folge von Wert-Tupeln. Dazu werden die Werte (Felder) jedes Tupels unter Verwendung eines Trennzeichens konkateniert und alle Tupel werden ihrerseits unter Verwendung eines anderen Trennzeichens konkateniert. Zahlenwerte werden in Dezimaldarstellung repräsentiert.

Das Trennzeichen für die Werte eines Tupels ist das „*Unit Separator*“-Zeichen des ASCII-Standards [ISO:646 (1991)] (Hex-Code `1F`), es ist in `ConceptInfoStoreSchema.referenceFieldSeparator` definiert und kann über den Tweak-Wert `dbstore.cacheReferenceFieldSeparator` konfiguriert werden (siehe ▹B.5). Analog ist das Trennzeichen für die Tupel untereinander das „*Record Separator*“-Zeichen des ASCII-Standards (Hex-Code `1E`), es ist in `ConceptInfoStoreSchema.referenceSeparator` definiert und kann über den Tweak-Wert `dbstore.cacheReferenceSeparator` konfiguriert werden. Des Weiteren ist die Gesamtlänge der serialisierten Daten pro Feld beschränkt — die Maximallänge kann über den Tweak-Wert `dbstore.listBlobSize` angepasst werden und liegt per Default bei 65025 Bytes.

Die vorberechneten Relationen sind die Basis für die Klassen `GlobalConcept` und `LocalConcept` für Transferobjekte (siehe ▹D.2). Jeder Eintrag in einer solchen Liste wird typischerweise durch ein Transferobjekt vom Typ `GlobalConceptReference`, `GlobalConceptReference` bzw. `TermReference` repräsentiert. Die Deserialisierung ist entsprechend in `WikiWordReference.parseList` implementiert.

10. Export

Um die von WikiWord gewonnenen Informationen für andere Systeme verfügbar zu machen, müssen sie in ein standardisiertes Modell überführt und in einem wohlbekannten Datenformat repräsentiert werden. In Hinblick auf die Kompatibilität mit einer möglichst großen Zahl von existierenden Systemen wurde RDF (*Resource Description Framework*, [W3C:RDF (2004)]) als die Basis des Datenmodells gewählt. RDF ist eine Empfehlung des W3C, die als flexibles Modell zur Datenrepräsentation im *Semantic Web* entworfen wurde. RDF ist weit verbreitet für die Darstellung von Faktenwissen in *Wissensorganisationssystemen* (*Knowledge Organisation Systems*, KOS) wie Thesauri, Taxonomien, Ontologien und ähnlichem, unter anderem auch im Bereich *Information Retrieval* und *Automatische Sprachverarbeitung* (siehe zum Beispiel [Auer u. a. (2007)] und [Gregorowicz und Kramer (2006)], vergleiche auch [van Assem u. a. (2006)]).

Das Datenmodell, das RDF zugrunde liegt, basiert auf sogenannten *Aussagen* (*Statements*), die als Tripel von *Subjekt*, *Prädikat* und *Objekt* dargestellt werden. Subjekte und Prädikate sind *Ressourcen*, die eindeutig durch eine URI identifiziert sind. Das Objekt kann entweder auch eine solche Ressource sein, oder ein Literal, also ein atomarer Wert. Literalen kann, zusätzlich zu ihrem textuellen Wert, eine Sprache zugeordnet sein, so dass im Kontext unterschiedlicher Sprachen verschiedene Werte gewählt werden können. Des weiteren kann ihnen ein Datentyp zugewiesen sein, der seinerseits eine (durch eine URI identifizierte) RDF-Ressource ist — auf diese Weise lässt sich angeben, wie das Literal zu interpretieren ist, zum Beispiel als Text (*String*), als Zahl, als Datumsangabe, usw.

Fest definierte RDF-Ressourcen, die in einem logischen Zusammenhang stehen, bilden zusammen ein sogenanntes *Vokabular*. Typischerweise definiert ein Vokabular Prädikate und Klassen zur Beschreibung bestimmter Arten (*Domänen*) von Objekten und ihrer Beziehungen. Die URIs der Ressourcen eines Vokabulars haben meist einen gemeinsamen *Präfix* — um diesen nicht ständig wiederholen zu müssen, kann er, nach vorheriger Festlegung, durch eine Abkürzung ersetzt werden; so entsteht aus der URI ein sogenannter *QName* (also ein *qualifizierter Name*), was in etwa der Verwendung von *Namensräumen* in XML entspricht [W3C:XML (2006), W3C:N3 (2006)]. Zum Beispiel kann so die URI `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` als `rdf:type` geschrieben werden — `rdf:type` ist ein in RDF vordefiniertes Prädikat, das einer Ressource einen *Typ* zuweist, also aussagt, dass das Subjekt eine Instanz derjenigen Klasse ist, die von dem Objekt der Aussage repräsentiert wird.

Für RDF sind verschiedene Datenformate (*Serialisierungen*) definiert, insbesondere *N3* (*Notation 3* [W3C:N3 (2006)], mit den Varianten *Turtle* [Backett (2007)] und *N-Triples* [W3C:NTriples (2004)]) sowie *RDF/XML* [W3C:RDF/XML (2004)]. WikiWord verwendet per Default *Turtle* für die Ausgabe von RDF. Für die Konvertierung zwischen diesen Repräsentationen steht eine Vielzahl von Werkzeugen zur Verfügung, zum Beispiel `rapper` aus der *Redland*-Bibliothek von *Dave Beckett*[1].

Im Folgenden wird erläutert, wie WikiWord-Datensätze auf das Datenmodell von RDF abgebildet werden können. Das Vorgehen folgt dabei im Wesentlichen [W3C (2005)] und orientiert sich weiter an [Gregorowicz und Kramer (2006)] sowie [van Assem u. a. (2006)].

[1] <`http://librdf.org/`>, Versionen vor 1.4.17 haben unter Umständen Probleme mit bestimmten Escape-Sequenzen in URIs

10.1. URIs

Ein wichtiger Aspekt der Datenrepräsentation auf der Basis von RDF ist es, gute URIs für die einzelnen RDF-Ressourcen zu wählen. URIs sind universell eindeutige Bezeichner (*Identifikatoren*) für eine beliebige Ressource (also irgend eine „Sache“), deren Syntax in [RFC:3986 (2005)] festgelegt ist. URIs ähneln äußerlich stark URLs, unterscheiden sich aber insbesondere durch ihre Semantik und Verwendung: URLs erlauben einen Zugriff auf eine elektronische Ressource, indem sie das Zugriffsprotokoll sowie den Ort der Ressource angeben, während URIs die Ressource lediglich eindeutig identifizieren [W3C:URI (2001)]. Dabei kann die URL eines Dokumentes gleichzeitig als seine URI dienen, und umgekehrt kann die URI einer Ressource so gestaltet sein, dass sie als URL der Ressource selbst, oder zumindest einer Beschreibung der Ressource fungieren kann.

Eine wichtige Eigenschaft von URIs (und idealerweise auch URLs) ist, dass die URI einer Ressource möglichst stabil bleibt, sich also nicht verändert, und dass sie möglichst unabhängig ist von rein technischen Aspekten wie dem physischen Speicherort und dem Datenformat [Berners-Lee (1998)]. Bei der Wahl von URIs ist daher auch entscheidend, wie *aussagekräftig* sie sind: beliebige Bezeichner, die zum Beispiel auf automatisch zugewiesenen numerischen IDs in irgend einer Datenbank basieren, sind weniger geeignet als URIs, die menschenlesbare Informationen über die Ressource enthalten, für die sie stehen. Solche „lesbaren“ URIs bleiben in der Regel über die Zeit stabil, sind von technischen Gegebenheiten unabhängig und können häufig ohne zusätzliches Wissen aus einem Teil des Namens hergeleitet werden. Zum Beispiel kann die URL (und URI) des Wikipedia-Artikels *Wasser* einfach durch Voranstellen eines Präfix erzeugt werden: `<http://de.wikipedia.org/wiki/Wasser>`.

Bei der Vergabe von URIs ist außerdem das Problem der *Identitätskrise* zu beachten [Miles (2005a), Pepper und Schwab (2003), Halpin (2006)], welches im Kontext von RDF häufig auftritt und oft nicht ausreichend gelöst wird: Wenn ein Dokument gegeben ist, das ein bestimmtes Konzept beschreibt, dann liegt es nahe, die URL des Dokuments als URI für dieses Konzept zu verwenden. Das wäre aber ein Fehler, da dann nicht zwischen dem Dokument selbst und dem *Gegenstand* des Dokuments unterschieden werden kann. Wenn wir zum Beispiel die URL der Wikipedia-Seite `http://de.wikipedia.org/wiki/Urfaust` als URI für das Konzept Urfaust ansehen und dementsprechend Johann_Wolfgang_von_Goethe als Autor zuordnen, führt das zu Verwirrung, weil Goethe dann als Mitautor der Wikipedia-Seite verstanden werden könnte. Es muss daher stets zwischen der URI des Dokumentes (die meist gleich der URL ist) und der URI für das Konzept, das das Dokument beschreibt, unterschieden werden — wobei die URI des Konzeptes durchaus auch als URL für das betreffende Dokument dienen kann: die URLs dürfen zu denselben Daten führen, solange bei der Verwendung der URIs klar zwischen Dokument und Konzept unterschieden wird.

Es ist außerdem der Unterschied zwischen *RDF-Ressourcen* und *WikiWord-Ressourcen* zu beachten: eine RDF-Ressource ist irgendein durch eine URI identifiziertes Objekt, das Gegenstand von Aussagen (*RDF-Statements*) sein kann; eine WikiWord-Ressource ist dagegen immer eine Wikipedia-Seite. In dem hier beschriebenen RDF-basierten Datenmodell werden unter anderem WikiWord-Konzepte, Wikipedia-Seiten (also WikiWord-Ressource), Korpora und Konzepttypen durch RDF-Ressourcen modelliert — was nichts weiter heißt, als dass ihnen eine eindeutige URI zugewiesen wird.

Zusätzlich zu den in ▸10.2 beschriebenen Standard-Vokabularen benutzt WikiWord die folgenden URI-Schemata für die Repräsentation der Thesaurusdaten:

`http://`*`Sprache`*`.wikipedia.org/wiki/`*`Seitentitel`*: Die URL einer Wikipedia-Seite repräsentiert die Seite selbst als Ressource.

`http://brightbyte.de/vocab/wikiword#`*`Element`*: Dieses URI-Schema dient als Basis für das Vokabular, das zur Repräsentation der WikiWord-Strukturen in RDF definiert ist. Es enthält verschiedene Datentypen und Prädikate, die weiter unten beschrieben und in der Datei `/WikiWord/WikiWord/src/main/wikiword.rdf` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/WikiWord/src/main/wikiword.rdf`> formal definiert werden. In Folgenden wird der URI-Präfix des WikiWord-Vokabulars entsprechend der *QName*-Notation mit `ww` abgekürzt.

`http://brightbyte.de/vocab/wikiword/concept-type#`*`Konzepttyp`*: Dieses URI-Schema wird für die Codes der Konzepttypen verwendet (siehe ▸8.6).

`http://brightbyte.de/vocab/wikiword/resource-type#`*`Seitentyp`*: Dieses URI-Schema wird für die Codes der Ressourcetypen verwendet (siehe ▸8.5). Allerdings werden Ressourcetypen in der hier beschriebenen Repräsentation des Thesaurus nicht wiedergegeben — sie werden vor allem während des Imports benötigt.

`http://brightbyte.de/vocab/wikiword/dataset/`*`Qualifier`*`/`*`Kollektion`*`:`*`Datensatz`*: Dieses URI-Schema wird für die eindeutige Identifizierung von Datensätzen benutzt: dabei stehen *`Datensatz`* und *`Kollektion`* für den Namen des Datensatzes und die Datensatz-Kollektion wie eingangs in ▸1.1 beschrieben. *`Qualifier`* repräsentiert einen Bezeichner (zum Beispiel einen *Domain Name*) der Organisation, die den Datensatz erstellt. Er kann über den Tweak-Wert `rdf.dataset.qualifier` konfiguriert werden. Aus der Kombination dieser Angaben ergibt sich idealerweise eine global eindeutige URI für den Datensatz.

`http://brightbyte.de/vocab/wikiword/concept/from/`*`Korpus-URL`*`/`*`Seitentitel`*: Dieses URI-Schema repräsentiert Konzepte anhand eines eindeutigen Seitennamens in einem Korpus. *`Korpus-URL`* ist dabei die URI (bzw. URL) des Korpus (typischerweise, aber nicht unbedingt, eines Wikipedia-Projektes) und *`Seitentitel`* ist der Name der Ressource in diesem Korpus, die das gewünschte Konzept definiert oder beschreibt. Dieser Weg der Identifikation von RDF-Ressourcen sollte bevorzugt werden, wenn das Konzept eindeutig einem Dokument in einem Korpus zugeordnet werden kann. Solche URIs sind *relativ* stabil gegenüber Änderungen an der Datenbasis[2] und dem Modus der Verarbeitung.

`http://brightbyte.de/vocab/wikiword/concept/in/`*`Thesaurus-URI`*`/`*`Konzept-ID`*: Dieses URI-Schema repräsentiert Konzepte anhand ihrer eindeutigen, numerischen ID innerhalb eines Datensatzes. *`Thesaurus-URI`* ist dabei die URI eines Datensatzes

[2] Es kommt vor, dass sich der in der Wikipedia für ein Konzept verwendete Seitentitel ändert. Dabei wird aber nur selten ein Titel gewählt, der vorher für ein anderes Konzept verwendet wurde. Meist wird auch der alte Titel als Weiterleitung (Alias) behalten. Vergleiche [Hepp u. a. (2006)].

(typischerweise, aber nicht unbedingt, wie oben beschrieben) und *Konzept-ID* ist die numerische ID, in hexadezimaler Notation, mit einem vorangestellten „*x*“. Dieser Weg der Identifikation von RDF-Ressourcen kann verwendet werden, wenn die direkte Adressierung über einen Namen nicht möglich ist. Solche URIs sind instabil insofern, als dass sie für alternative oder zukünftige Versionen des Thesaurus nutzlos sind.

Die zwei unterschiedlichen Wege, eine URI für ein Konzept zu bestimmen, werden auch in [van Assem u. a. (2006)] diskutiert. WikiWord benutzt beide: Konzepte aus einem lokalen Datensatz werden über das namensbasierte URI-Schema adressiert, weil sie direkt einem Dokument in einem Korpus zugeordnet werden können, nämlich dem Wikipedia-Artikel, der sie beschreibt. Eine solche URI ist relativ stabil und unabhängig von dem Kontext, in dem das Konzept verarbeitet oder betrachtet wird. Das Konzept wird dabei letztlich dadurch definiert, wie die Seite in der Wikipedia *verwendet* wird [Black (2006)].

Konzepte aus einem globalen Datensatz können auf diese Weise nicht adressiert werden: sie sind nicht unbedingt einem, sondern möglicherweise mehreren Dokumenten zugeordnet. Außerdem könnte dasselbe Konzept unterschiedlichen Mengen von Dokumenten zugeordnet sein, je nachdem, welche Sprachen bei der Erstellung des mehrsprachigen Thesaurus betrachtet werden, wie sich die Language-Links in den einzelnen Artikeln entwickeln oder ob neue Artikel hinzukommen, die dieses Konzept beschreiben.

Daher verwendet WikiWord für globale Konzepte das zweite URI-Schema, das auf der internen ID des Konzeptes beruht: diese ID bezieht sich direkt auf den konkreten Datensatz, sie ist also „empfindlich“ gegen jede Änderung an den Ausgangsdaten oder der Konfiguration von WikiWord. Um nun URIs identifizieren zu können, die dasselbe Konzept in verschiedenen Datensätzen repräsentieren, eignet sich die Methode des „*semantic Handshake*“: den sprachunabhängigen Konzepten werden dazu über das Prädikat `owl:sameAs` „ihre“ sprachspezifischen Konzepte zugeordnet. Zwei sprachunabhängige Konzepte aus unterschiedlichen globalen Datensätzen (also multilingualen Thesauri) können dann als äquivalent angesehen werden, wenn sie beide mit dem selben sprachspezifischen Konzept identifiziert sind (für das es ja eine stabile URI gibt).

Auf diese Weise ergibt sich zwar keine eindeutige Zuordnung von Konzepten aus einem Thesaurus zu Konzepten in dem anderen — eine solche Zuordnung existiert im allgemeinen gar nicht, da je nach Konfiguration und Datenbasis lokale Konzepte unterschiedlich zu globalen gruppiert werden können. Aber der *semantic Handshake* erlaubt doch in der Regel eine weitgehende *semantische Integration* (vergleiche [Maicher (2007)]).

10.2. Vokabulare

Basierend auf dem sehr einfachen Datenmodell von RDF sind eine Vielzahl von Standard-Vokabularen definiert. Einige dieser Vokabulare werden auch von WikiWord eingesetzt:

RDF Schema (`http://www.w3.org/2000/01/rdf-schema#`, [W3C:RDFS (2000)], im Folgenden `rdfs`) ist ein Vokabular, um RDF-Vokabulare zu beschreiben. Es definiert

unter anderem Prädikate zum Ausbau einer Ableitungshierarchie von Klassen und Eigenschaften.

Web Ontology Language (`http://www.w3.org/2002/07/owl#`, [W3C:OWL (2004)], im Folgenden `owl`) ist ein Vokabular zum Aufbau von Ontologien. Ähnlich wie RDFS definiert es Prädikate zur Beschreibung von Klassen und Eigenschaften sowie ihrer Beziehungen untereinander, bietet dabei jedoch weit größere Ausdrucksmächtigkeit. Die Semantik von OWL entspricht einer (gerade noch) entscheidbaren Untermenge der Prädikatenlogik erster Stufe, der sogenannten *Beschreibungslogik*.

XML Schema (`http://www.w3.org/2001/XMLSchema#`, [W3C:XSD (2004)], im Folgenden `xsd`) ist ein Vokabular zur Spezifikation von XML-Formaten. Im Kontext von RDF werden aus dem XSD Vokabular fast ausschließlich die Datentypen verwendet, die es für Literale definiert.

Dublin Core (`http://purl.org/dc/elements/1.1/`, [DCME (2008)], im Folgenden `dc`) ist ein Vokabular zu Beschreibung von Medien. Es definiert unter anderem Prädikate zur Angabe von Autoren, Quellen, Publikationsdaten, etc. *Dublin Core* wird vor allem für dokumentbezogene Metadaten verwendet, unter anderem auch oft in HTML-Dateien und, in diesem Fall, Thesauri.

Simple Knowledge Organization System (`http://www.w3.org/2004/02/skos/core#`, [W3C:SKOS (2004)], im Folgenden `skos`) ist ein Vokabular zur Beschreibung einfacher Wissensstrukturen, insbesondere entworfen für Thesauri, Taxonomien und ähnliche sprachorientierte Wissensbasen [Miles (2005b)]. Es definiert Relationen für den Aufbau einer Subsumtionshierarchie von Konzepten, für die Angabe von Verwandtheit zwischen Konzepten, sowie zur Zuweisung von Termen (*Labels*) zu Konzepten.

Das SKOS-Vokabular bietet sich zur Repräsentation der WikiWord-Daten an, da das Datenmodell dem Modell der WikiWord-Datensätze bereits recht ähnlich ist (vergleiche ▹4.2 und ▹C.1, siehe auch [Gregorowicz und Kramer (2006), van Assem u. a. (2006)]). Zudem unterstützen zahlreiche Wissensorganisationssysteme im Bereich der automatischen Sprachverarbeitung und des *Information Retrieval* das SKOS-Vokabular zur Repräsentation von Term-Konzept-Graphen.

Aus den Standard-Vokabularen werden vor allem die folgenden Klassen und Eigenschaften benutzt:

`rdf:type` weist einer RDF-Ressource einen Typ, also eine Klasse zu.

`owl:sameAs` gibt an, dass zwei RDF-Ressourcen identisch sind, dass also zwei URIs dasselbe Ding bezeichnen.

`skos:Concept` ist die Klasse für Konzepte im Thesaurus.

`skos:ConceptScheme` ist die Klasse für den Thesaurus selbst.

`skos:inScheme` gibt an, zu welchem Thesaurus ein Konzept gehört.

skos:broader und skos:narrower verbinden allgemeinere mit spezielleren Konzepten.

skos:related verbindet Konzepte, die miteinander semantisch verwandt sind.

skos:prefLabel gibt den bevorzugten *Label* für ein Konzept an. Muss pro Sprache und Thesaurus (*Scheme*) eindeutig sein.

skos:altLabel gibt alternative *Labels* für ein Konzept an.

skos:definition ist die Definition eines Konzeptes, als Literal oder URL eines Dokumentes.

WikiWord definiert auch ein eigenes RDF-Vokabular (▹F.3), das die Möglichkeiten von SKOS etwas erweitert, um mehr Informationen aus den Datenmodellen repräsentieren zu können (vergleiche [van Assem u. a. (2006)]). Das Programm `ExportRdf` (▹10.3, ▹B.1) kann dieses Vokabular verwenden, um einen Thesaurus zu beschreiben. Alternativ kann aber mit Hilfe der Option `--skos` aber auch die Ausgabe von „reinem“ SKOS erwirkt werden, um so (auf Kosten der Aussagekraft) die Kompatibilität zu erhöhen.

Das WikiWord-Vokabular definiert die folgenden Prädikate:

ww:similar ist eine Relation, die semantische *Ähnlichkeit* zwischen zwei Konzepten ausdrückt (siehe ▹9.1.3 und ▹C.2). Diese Beziehung ist Spezialisierung von `skos:related`.

ww:assoc ist eine Relation, die eine unspezifizierte Assoziation von Konzepten repräsentiert. Diese Beziehung ist schwächer als `skos:related` und muss nicht symmetrisch sein. Sie wird vor allem zur Repräsentation von Querverweisen (Hyperlinks) verwendet.

ww:concept-type ist die Klasse für Konzepttypen, wie in ▹8.6 beschrieben. Die Standard-Konzepttypen sind im Namensraum `http://brightbyte.de/vocab/wikiword/concept-type#` definiert.

ww:type bestimmt den Typ des Konzeptes, wie in ▹8.6 beschrieben. Die einzelnen Typen werden als RDF-Ressourcen vom Typ `ww:concept-type` modelliert.

ww:displayLabel gibt (pro Sprache) einen Namen an, der zur Anzeige des Konzeptes verwendet werden kann. Diese Eigenschaft ist ähnlich zu `skos:prefLabel`, mit einigen wichtigen Unterschieden: ein Wert von `ww:displayLabel` kann auch gleichzeitig noch einmal als Wert von `skos:altLabel` auftreten und `ww:displayLabel` impliziert auch nicht, dass der Wert in natürlicher Sprache für das Konzept verwendet wird.

Den Konzepten sind Bewertungen und Ränge bezüglich verschiedener Maße zugeordnet (▹9.3, ▹C.2). Diese Werte können auch in RDF repräsentiert werden:

ww:score ist eine Relation, die einem Konzept eine Bewertung bezüglich irgend eines Maßes zuweist. Bewertungen sind beliebige reelle Zahlen. Diese Relation dient als Basis für die im Folgenden definierten spezifischen Bewertungen.

ww:idfScore ist die *Inverse Document Frquency*.

ww:lhsScore ist der *Local Hierarchy Score*.

ww:inDegree ist die Anzahl der eingehenden Hyperlinks.

ww:outDegree ist die Anzahl der ausgehenden Hyperlinks.

ww:linkDegree ist die Summe der eingehenden und ausgehenden Hyperlinks.

Für die Bestimmung und Bedeutung der einzelnen *Scores*, siehe ▹9.3. Bezüglich jeder dieser Bewertungen ist jedem Konzept ein Rang zugeordnet:

ww:rank ist eine Relation, die einem Konzept einen Rang bezüglich irgend eines Maßes zuweist. Ränge sind natürliche Zahlen und ihre Zuordnung ist eindeutig insofern, als dass keine zwei Konzepte in einem Thesaurus bezüglich desselben Maßes denselben Rang haben können. Diese Relation dient als Basis für die im Folgenden definierten spezifischen Ränge.

ww:idfRank ist der Rang bezüglich der *Inverse Document Frequency*.

ww:lhsRank ist der Rang bezüglich des *Local Hierarchy Score*.

ww:inRank ist der Rang bezüglich der Anzahl der eingehenden Hyperlinks.

ww:outRank ist der Rang bezüglich der Anzahl der ausgehenden Hyperlinks.

ww:linkRank ist der Rang bezüglich der Summe der eingehenden und ausgehenden Hyperlinks.

10.3. Abbildung des Datenmodells auf RDF

Das in vorausgegangenen Kapiteln beschriebene Datenmodell (▹4.2, ▹C.1) kann fast direkt auf SKOS abgebildet werden, und zwar sowohl das lokale Datenmodell für einen monolingualen als auch das globale Datenmodell für einen multilingualen Thesaurus. Diese Abbildung folgt im Wesentlichen [W3C (2005), Matthews (2003)] und [van Assem u. a. (2006)] sowie dem Vorgehen von [Gregorowicz und Kramer (2006)]. Dabei sind die RDF-Daten des multilingualen Thesaurus, anders als bei den internen Datensätzen, unabhängig von den monolingualen Thesauri.

Der Export als RDF ist in der Klasse `ExportRdf` implementiert. Ausgehend von den internen Datenmodellen werden *RDF-Statements* generiert, wobei verschiedene Optionen und Modi zu berücksichtigen sind:

Monolingualer Thesaurus: Die Relationen aus dem lokalen Datensatz werden direkt ausgegeben.

Multilingualer Thesaurus: Die Relationen aus dem globalen Datensatz werden direkt ausgegeben. Für jede Sprache, auf die sich der globale Datensatz bezieht, werden die sprachspezifischen Informationen aus dem entsprechenden lokalen Datensatz ausgelesen, mit den globalen Konzepten verknüpft und ausgegeben. Alle Relationen beziehen sich auf globale Konzepte.

`--skos-Option`: Diese Option bewirkt, dass das WikiWord-Vokabular nicht verwendet und statt dessen auf das SKOS-Vokabular zurückgegriffen wird. Das verbessert unter Umständen die Kompatibilität, es gehen aber mehr Informationen verloren.

`--assoc-Option`: Diese Option bewirkt, dass zusätzlich die `ww:assoc`-Relation ausgegeben wird, die die Hyperlinks zwischen Wikipedia-Artikeln als „lose“ Assoziation der entsprechenden Konzepte interpretiert.

`--noscore-Option`: Diese Option bewirkt, dass keine Bewertungen und Rankings für die einzelnen Konzepte ausgegeben werden.

`--cutoff` und `--force-cutoff`: Diese Optionen steuern das Cutoff-Verhalten für die Termzuordnung, siehe ▸10.4.

`--min-langmatch`: Diese Optionen bestimmt den Cutoff-Wert für die Bestimmung ähnlicher Konzepte: sie gibt an, wie viele Language-Links zwei Konzepte gemeinsam haben müssen, um als ähnlich angesehen zu werden (vergleiche ▸12.3).

Als erstes wird dem Thesaurus an sich (also dem Datensatz) nach dem oben beschriebenen Muster eine URI zugeteilt und Metadaten zugewiesen. Dafür wird vor allem das *Dublin Core* Vokabular verwendet:

`rdf:type` wird auf `skos:ConceptScheme` gesetzt, um anzugeben, dass es sich bei der Entität, die die URI bezeichnet, um ein *ConceptScheme* im Sinne von SKOS handelt, also um eine Wissensstruktur oder Konzept-Term Netz, wie sie ein konzeptorientierter Thesaurus darstellt.

`dc:identifier` weist dem Thesaurus eine eindeutige ID zu. Hier wird die URI des Thesaurus als Literal angegeben.

`dc:creator` wird immer mit „WikiWord“ angegeben, da die Erzeugung ja vollautomatisch ist. Wurde der Tweak-Wert `rdf.dataset.qualifier` definiert, so wird der angegebene Wert hier mit ausgegeben, da der „*Qualifier*“ ja die Person oder Institution beschreiben soll, die den Thesaurus generiert.

`dc:created` gibt den Zeitpunkt an, zu dem der Thesaurus exportiert wurde.

`dc:title` gibt den Titel des Thesaurus an, also den Namen des Datensatzes, inklusive der Kollektion.

`dc:description` ist ein kurzer Text, der besagt, dass die Daten mittels WikiWord aus der Wikipedia erzeugt wurden.

dc:rights gibt Hinweise zum Urheberrecht an den Thesaurusdaten. Dieser Text besagt, dass der Thesaurus automatisch erstellt wurde und daher nicht urheberrechtlich geschützt ist. Für die Rechte an zitierten Texten aus der Wikipedia, konkret also den Glossen, wird auf die GFDL [GFDL] sowie die Autorenliste auf den betreffenden Wikipedia-Seiten verweisen.

dc:source gibt die Quelle der Thesaurusdaten an: das sind die Wikipedia-Projekte, aus denen die Daten extrahiert wurden, repräsentiert durch ihre URL. Im Falle eines multilingualen Thesaurus wird dieses Prädikat einmal für jede Sprache gesetzt.

dc:language gibt die Sprache der Thesaurusdaten an. Im Falle eines multilingualen Thesaurus wird diese Eigenschaft nicht angegeben.

Bei einem multilingualen Thesaurus wird für jede enthaltene Sprache zusätzlich das *ConceptScheme* für den betreffenden monolingualen Thesaurus deklariert: die URI des betreffenden lokalen Datensatzes wird bestimmt und ihm werden die folgenden Prädikate zugewiesen:

rdf:type wird auf `skos:ConceptScheme` gesetzt.

dc:language gibt die Sprache des lokalen Datensatzes an.

Die so deklarierten lokalen Thesauri werden später als *ConceptScheme* bei der Zuordnung von sprachspezifischen Konzepten zu sprachunabhängigen Konzepten verwendet.

Für jedes Konzept im Datensatz wird nun eine URI bestimmt (nach einem geeigneten Verfahren, je nachdem, ob es sich um einen multilingualen oder monolingualen Thesaurus handelt, siehe ▸10.1). Den Konzepten werden dann die folgenden Eigenschaften zugewiesen:

rdf:type wird auf die Klasse `skos:Concept` gesetzt, um anzugeben, dass die URI ein SKOS-Konzept identifiziert.

skos:inScheme wird auf die URI des Thesaurus gesetzt, zu dem das Konzept gehört.

skos:definition wird die URL der Wiki-Seite zugewiesen, die dieses Konzept definiert — diese Information wird dem Feld `resource` in der Tabelle `concept` entnommen (▸C.2). Im Falle eines multilingualen Thesaurus (und damit eines potentiell sprachübergreifenden Konzeptes) wird für jede Sprache, in der das Konzept definiert ist, eine Wiki-Seite zugewiesen; die Zuordnung erfolgt dabei über die Tabelle `origin` (▸C.2).

skos:definition wird noch einmal gesetzt, diesmal auf den Definitionssatz, der aus der Wikipedia-Seite extrahiert wurde — der Text wird der Tabelle `definition` entnommen (▸C.2). Dabei wird die Sprache angegeben, in der die Definition verfasst ist. Im Falle eines multilingualen Thesaurus würde für jede Sprache, in der das Konzept definiert ist, (potentiell) eine Definition zugewiesen; die Zuordnung erfolgt dabei über die Tabelle `origin`.

Für jedes Konzept werden außerdem verschiedene Bewertungen und *Rankings* angegeben, falls diese in der Datenbank verfügbar sind (also falls sie mit `BuildStatistics` erzeugt wurden) und nicht die `--skos`-Option die Verwendung des WikiWord-Vokabulars verbietet oder die Ausgabe dieser Eigenschaften mit der `--noscore`-Option deaktiviert wurde. Es handelt sich im Einzelnen um folgende Eigenschaften:

`ww:idfScore` und `ww:idfRank` sind die *Inverse Document Frequency* bzw. der Rang bezüglich dieses Wertes.

`ww:lhsScore` und `ww:lhsRank` sind der *Local Hierarchy Score* bzw. der Rang bezüglich dieses Wertes.

`ww:inDegree` und `ww:inRank` sind die Anzahl der eingehenden Hyperlinks bzw. der Rang bezüglich dieses Wertes.

`ww:outDegree` und `ww:outRank` sind die Anzahl der ausgehenden Hyperlinks bzw. der Rang bezüglich dieses Wertes.

`ww:linkDegree` und `ww:linkRank` sind die Summe der eingehenden und ausgehenden Hyperlinks bzw. der Rang bezüglich dieses Wertes.

Diese Eigenschaften sind der Tabelle `degree` entnommen (▹C.2). Für multilinguale Thesauri wird außerdem für jedes sprachübergreifende Konzept seine Beziehung zu sprachspezifischen Konzepten angegeben: über die Tabelle `origin` werden alle zugehörigen sprachspezifischen Konzepte bestimmt. Diesen wird, nach dem Schema für den betreffenden lokalen Datensatz, eine URI zugeordnet und dann die folgenden Eigenschaften definiert:

`rdf:type` wird auf `skos:Concept` gesetzt.

`skos:inScheme` wird auf die URI des *lokalen* Datensatzes gesetzt.

`owl:sameAs` wird auf die URI des sprachübergreifenden Konzeptes gesetzt; diese Relation ist symmetrisch und wird auch explizit in beide Richtungen angegeben: auch das sprachübergreifende Konzept bekommt über das Prädikat `owl:sameAs` das sprachspezifische Konzept zugewiesen.

Diese Zuordnung drückt einen wichtigen Aspekt der Funktionsweise von WikiWord aus: *dasselbe* Konzept kann in mehreren Korpora und unterschiedlichen Sprachen repräsentiert sein. Die Eigenschaften des sprachübergreifenden Konzeptes ergeben sich aus der Vereinigung der Eigenschaften einer Menge paarweise äquivalenter sprachspezifischer Konzepte (▹9.2.2). Jede der lokalen Repräsentationen und auch die sprachunabhängige, globale Repräsentation des Konzeptes haben eine URI. Das OWL-Prädikat `owl:sameAs` wird nun hier verwendet, um anzugeben, dass all diese URIs dasselbe Konzept bezeichnen.

Auf diese Weise ergibt sich auch eine Zuordnung von stabilen URIs für sprachspezifische Konzepte zu einer instabilen URI eines sprachübergreifenden Konzeptes. So können die URIs, die für dasselbe sprachübergreifende Konzept in unterschiedlichen multilingualen Thesauri verwendet werden, gefunden und gleichgesetzt werden (*Semantic Handshake*).

Terme im WikiWord-Modell, wie sie in der Tabelle `meaning` (▹C.2) gespeichert sind, werden als „*Labels*“ in SKOS interpretiert. Für alle Einträge der Bedeutungsrelation werden *Statements* mit den folgenden Prädikaten erzeugt:

`ww:displayLabel` ist der Name des Konzeptes, der zur Anzeige verwendet werden kann, also zur Repräsentation des Konzeptes in einer Benutzerschnittstelle. Der Wert wird dem Feld `name` in der Tabelle `concept` entnommen — in einem lokalen Datensatz entspricht er auch dem Namen der Wikipedia-Seite, die das Konzept definiert. Dieser Name muss nicht unbedingt ein Wort oder ein Phrase aus der natürlichen Sprache sein, er kann künstlich erzeugt werden[3]. Im Falle eines multilingualen Thesaurus wird der synthetische Name des globalen Konzeptes verwendet (▹9.2.2). Der Wert dieser Eigenschaft kann noch einmal als Wert der Eigenschaft `skos:altLabel` vorkommen. `ww:displayLabel` wird nicht im `--skos`-Modus verwendet.

`skos:prefLabel` wird im `--skos`-Modus als Alternative zu `ww:displayLabel` verwendet: der Wert wird genau so wie oben beschrieben aus dem Namen des Konzeptes bestimmt. Allerdings wird im Falle eines multilingualen Thesaurus dieser Wert für jede der betrachteten Sprachen einzeln gesetzt (es werden also die Namen der einzelnen sprachspezifischen Konzepte verwendet, nicht der Name des globalen Konzeptes). Auch hat `skos:prefLabel`, nach der SKOS-Spezifikation, eine etwas andere Semantik als `ww:displayLabel` und ist von allen Werten der Eigenschaft `skos:altLabel` in der betreffenden Sprache verschieden (das heißt, die entsprechende Zuweisung des Wertes über das Prädikat `skos:altLabel` wird übergangen).

`skos:altLabel` weist dem Konzept, für jede Sprache, eine Menge von Termen (*Labels*) zu. Diese Terme sind alle Bezeichnungen, die für das Konzept über die Bedeutungsrelation der Tabelle `meaning` definiert sind (mit Ausnahme des Werte von `ww:displayLabel`, falls diese Eigenschaft gesetzt wurde). Im Falle eines multilingualen Thesaurus werden die Terme für jede Sprache angegeben.

Bei diesem Export der Bedeutungsrelation geht eine wichtige Information verloren, die im lokalen Datenmodell gespeichert ist: das *Vertrauen* (*Confidence*) in die Termzuordnung, wie sie durch die Felder `freq` (Frequenz) und `rule` (Regel) der Tabelle `meaning` repräsentiert ist. Diese Information kann nicht ohne weiteres in RDF ausgedrückt werden, da RDF nur binäre Relationen kennt. Daher müssten die zusätzlichen Eigenschaften entweder als Prädikate einer *Reifikation* der betreffenden Relation modelliert werden, oder das Objekt der Relation, also der Term, müsste als komplexes Objekt definiert sein. Beide Wege sind mit einer deutlich höheren Komplexität des Datenmodells und einem größeren Datenaufkommen verbunden. Diese Möglichkeiten auszuloten geht über den Rahmen dieser Arbeit hinaus.

Allerdings kann die Information über das Vertrauen in die einzelnen Termzuordnungen dennoch beim Export genutzt werden: Die Zuordnung von Terme an Konzept über das Prädikat `skos:altLabel` kann danach gefiltert werden, wie *verlässlich* diese Zuordnung ist. Zuordnungen,

[3] insbesondere kommen häufig *Klammerzusätze* vor und Leerzeichen sind durch Unterstriche ersetzt.

die nicht „sicher“ genug sind, werden dann übergangen („*Cutoff*“, siehe ▹10.4). Über diesen Mechanismus lässt sich die *Präzision* der Termzuordnung auf Kosten der *Abdeckung* erhöhen.

Thesauri sind zumeist (poly-)hierarchisch entlang einer Subsumtionsrelation strukturiert, das heißt, zu jedem Konzept werden allgemeinere und speziellere Konzepte angegeben. Zusätzlich können Konzepte als „verwandt“ und/oder „ähnlich“ gekennzeichnet sein. Das drückt sich in den folgenden RDF-Prädikaten aus:

`skos:broader` und `skos:narrower` sind reziproke Relationen, die Konzepte mit allgemeineren bzw. spezielleren Konzepten verbinden und so die Subsumtionsrelation des Thesaurus definieren. Diese Verbindungen werden direkt der Tabelle `broader` (▹C.2) entnommen.

`ww:similar` verbindet zwei Konzepte, die zueinander semantisch „ähnlich“ sind; diese Information wird während des Imports bestimmt (siehe ▹9.1.3) und dann beim Export den Feldern `langmatch` und `langref` in der Tabelle `relation` (▹C.2) entnommen, wobei mit `--min-langmatch` der Minimalwert für das `langmatch` angegeben werden kann. So kann die Qualität der Ähnlichkeitsrelation bestimmt werden, siehe ▹12.3.

Paare von Konzepten, die bereits durch `skos:broader` oder `skos:narrower` verbunden sind, werden nicht als ähnlich ausgegeben. Im `--skos`-Modus wird `skos:related` statt `ww:similar` verwendet.

`skos:related` verbindet zwei Konzepte, die miteinander semantisch „verwandt“ sind; diese Information wird während des Imports bestimmt (siehe ▹9.1.3) und dann beim Export dem Feld `bilink` in der Tabelle `relation` entnommen. Paare von Konzepten, die bereits durch `skos:broader`, `skos:narrower` oder `ww:similar` verbunden sind, werden dabei übergangen. Im `--skos`-Modus werden „ähnliche“ Konzepte (siehe oben) einfach ebenfalls als „verwandt“ behandelt.

`ww:assoc` repräsentiert eine „schwache“ Assoziation zwischen zwei Konzepten: sie spiegelt die Verknüpfung durch Hyperlinks (Querverweise) wieder und wird der Tabelle `link` entnommen. Per Default wird diese Relation nicht ausgegeben, sie kann aber über die Option `--assoc` aktiviert werden (falls nicht die Option `--skos` gesetzt ist). Die Relation `ww:assoc` wird *nicht* als symmetrisch betrachtet.

Im Ergebnis werden alle wesentlichen Informationen aus den internen Datenmodellen als RDF-Strukturen repräsentiert, die Ausgabe erfolgt serialisiert in *Turtle*-Notation [Backett (2007)]. In Anhang F.3 sind die RDF-Statements für einige Konzepte als Beispiele aufgeführt.

10.4. Cutoff

Obgleich Link-Texte eine gute Quelle für Bezeichnungen für Konzepte sind (vergleiche [Mihalcea (2007)] sowie [Craswell u. a. (2001), Eiron und Mccurley (2003), Kraft und Zien (2004)]), können sie doch irreführend sein: es kommen Tippfehler vor, oder es wird irrtümlich auf das falsche Konzept verwiesen. Eine Möglichkeit, solche Fehler zu vermeiden, ist das *Gesetz der Großen Zahl*: es

werden einfach nur solche Terme als Bezeichnung für ein Konzept betrachtet, die oft genug als Link-Text in Verknüpfungen zu diesem Konzept vorkamen — die verlangte minimale Zahl von Wiederholungen (*Frequenz*) wird dann als *Cutoff-Wert* bezeichnet. Dabei ergibt sich ein *Trade-Off* zwischen der Präzision (*Precision*) und der Abdeckung (*Coverage*)[4] der Bedeutungsrelation: ein hoher *Cutoff-Wert* liefert mit großer Wahrscheinlichkeit nur korrekte Bezeichnungen für ein Konzept, schließt aber mit ebenfalls großer Wahrscheinlichkeit auch korrekte Bezeichnungen aus, die nützlich sein könnten.

Im dem von WikiWord verwendeten Datenmodell basiert die Bedeutungsrelation aber nicht ausschließlich auf Link-Texten: Terme werden auch wegen anderer Informationen wie dem Seitentitel, Weiterleitungen und Begriffsklärungen mit einem Konzept assoziiert (siehe ▹8.9 und ▹9.1). Aufgrund welcher Regel die Termzuordnung erfolgt, wird bei der Analyse des Wiki-Textes festgehalten und beim Aufbau der Bedeutungsrelation in der Tabelle `meaning` insoweit beibehalten, dass aus dem Feld `rule` ersichtlich ist, ob die Zuordnung *ausschließlich* erfolgte, weil der Term als Link-Text für dieses Konzept verwendet wurde (vergleiche ▹9.1.3 und ▹C.2).

Termzuweisungen, die *nicht* ausschließlich wegen einer Verwendung als Link-Text vorgenommen wurden, können als *explizite* Zuweisung betrachtet werden – es ist anzunehmen, dass diese Assoziationen eine höhere Qualität (also geringere Fehlerquote) haben als die Verwendungen als Link-Text. Daher scheint es sinnvoll, den Cutoff-Wert nur für Termzuweisungen anzuwenden, die nicht explizit vorgenommen wurden. Die im folgenden Kapitel beschriebenen Experimente bestätigen diese Annahme (▹12.5): werden explizite Zuordnungen unabhängig von der Frequenz beibehalten, verschlechtert sich die Präzision nicht wesentlich, aber die Abdeckung steigt deutlich. Dabei eignet sich, je nach Datenbasis und Präferenz für den Trade-Off zwischen Präzision und Abdeckung, eine Mindestfrequenz von zwei oder drei als Cutoff-Wert.

In der Implementation durch `ExportRdf` wird der Cutoff durch die beiden folgenden Parameter gesteuert:

`-cutoff N` verlangt, dass eine Termzuordnung mindestens eine Frequenz von *N* haben muss, um beim Export berücksichtigt zu werden.

`-force-cutoff` erzwingt die Anwendung des Cutoff auch dann, wenn die Termzuweisung explizit war.

Auf diese Weise lässt sich der Qualitätsanspruch gegen die gewünschte Abdeckung abwägen. Der Effekt der verschiedenen Cutoff-Modi wird in ▹12.5 evaluiert.

10.5. Topic Maps

Als Alternative zu RDF als Basis der Repräsentation des Thesaurus bieten sich *Topic Maps* an [Redmann und Thomas (2007), Garshol (2004), ISO:13250 (2003)]. Topic Maps sind ein ISO-Standard für

[4] Im Gegensatz zum *Recall*, der sich auf das theoretisch zu erreichende Maximum bezieht, bezieht sich die Abdeckung (*Coverage*) auf die Anzahl ohne Cutoff.

die Darstellung von Wissensorganisationssystemen, beziehungsweise für die Repräsentation von Konzepten (*Topics*) und ihren Beziehungen (*Associations*) und Bezeichnungen (*Names*). Sie haben unter anderem die folgenden Eigenschaften, die für die Repräsentation der WikiWord-Daten nützlich wären:

Scopes beschränken die Gültigkeit von Assoziationen und Namen auf bestimmte Kontexte — über diesen Mechanismus lassen sich gut sprachspezifische Informationen integrieren.

Assoziationen unterstützen eine beliebige Anzahl von *Akteuren*, es ist also leicht, mehrstellige Relationen zu repräsentieren.

Reifikation von Assoziationen wird direkt unterstützt und erlaubt damit eine einfache Integration von Metadaten, zum Beispiel darüber, aus welcher Quelle eine Assoziation stammt oder nach welcher Regel sie gebildet wurde.

Indicators vermeiden die *Identitätskrise* von URIs, da sie von *Locators* verschieden sind und es so erlauben, dieselbe URI zu verwenden, um auf ein Dokument oder den Gegenstand eines Dokuments zu verweisen, ohne Verwirrung zu stiften.

Andererseits ist die Semantik von Beziehungen in Topic Maps weitgehend unspezifiziert und es gibt weniger standardisierte Vokabulare, die die Weiternutzung der Daten erleichtern würden.

Eine Abbildung der WikiWord-Daten auf Topic Maps und ein Vergleich des resultierenden Modells mit dem RDF-Modell, das WikiWord generiert, konnte im Rahmen dieser Arbeit nicht untersucht werden und verbleibt als Gegenstand künftiger Forschung.

Teil IV.

Evaluation

11. Statistischer Überblick

Dieses Kapitel bietet einen Überblick über die mit mit WikiWord gewonnenen Daten. Dabei werden die Eingabedaten betrachtet, die Anzahl der extrahierten Elemente und Verknüpfungen, sowie die Verteilungen einiger Eigenschaften.

11.1. Datenbasis und Import

Zur Erprobung von WikiWord und um eine Datenbasis für die in diesem Kapitel beschriebenen Experimente zu schaffen, wurden Dumps von Wikipedia-Projekten in unterschiedlichen Sprachen importiert und ausgewertet. Dabei wurden Projekte gewählt, die aus verschiedenen Bereichen des Größenspektrums stammen. Allerdings wurde auf „exotische“ Sprachen verzichtet, da es dem Autor an Möglichkeiten mangelt, Ergebnisse für solche Sprachen in irgendeiner Form zu überprüfen. Konkret wurden die folgenden Dumps ausgewertet:

- Der Dump der englischsprachigen Wikipedia (`en`) vom 3. Januar 2008[1].
- Der Dump der deutschsprachigen Wikipedia (`de`) vom 20. März 2008.
- Der Dump der französischsprachigen Wikipedia (`fr`) vom 23. März 2008.
- Der Dump der niederländischsprachigen Wikipedia (`nl`) vom 10. März 2008.
- Der Dump norwegischsprachigen Wikipedia (`no`) vom 11. März 2008.
- Der Dump der Wikipedia in einfachem Englisch (`simple`) vom 16. März 2008.
- Der Dump der plattdeutschen Wikipedia (`nds`) vom 19. März 2008.

Die verwendeten Dumps sind in Anhang F.1 noch einmal genauer angegeben. Das Ergebnis des Imports ist in Form von SQL-Datenbanken verfügbar, siehe ▹F.2.

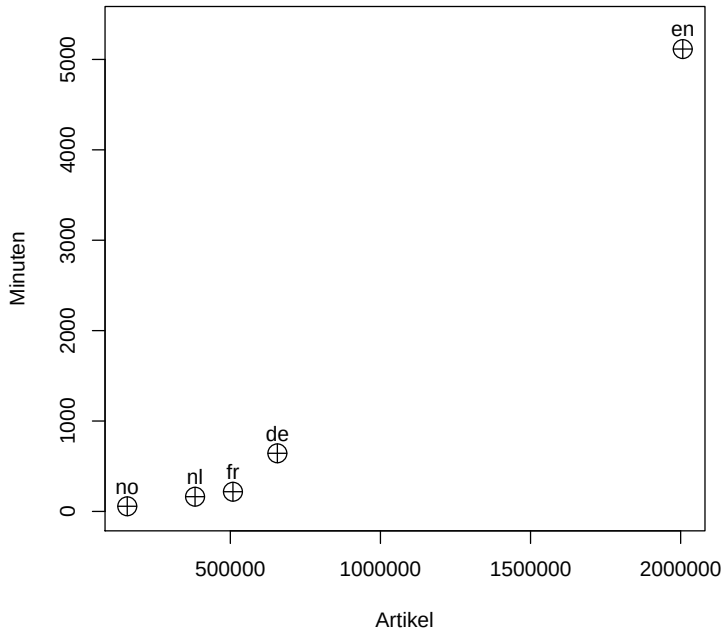
Die Dumps wurden unter Verwendung der verschiedenen Programme der Anwenderschnittstelle von WikiWord (▹B.1) in eine Kollektion mit dem Namen `full` importiert. Konkret wurden zunächst mit `ImportConcepts` die Konzepte aus den Dumps extrahiert (Schritte `analyze` und `process`, siehe ▹9.1), dann mit `BuildStatistics` die statistischen Daten erhoben (`statistics`, siehe ▹9.3) und dann mit `BuildConceptInfo` die Konzept-Informationen für den schnellen Zugriff aufgebaut (`info`, siehe ▹9.4).

[1] Der Dump der englischsprachigen Wikipedia vom 12. März konnte aufgrund eines Fehlers im UTF-8-Decoder der XML-Bibliothek *Xerces* nicht importiert werden (siehe <`https://issues.apache.org/jira/browse/XERCESJ-1257`>). Das Problem kann gegebenenfalls umgangen werden, indem man die Xerces-Bibliothek nicht in den Classpath aufnimmt und so dafür sorgt, dass Javas Standard-Implementation für das Parsen von XML verwendet wird. Das ist allerdings etwas langsamer.

	analyze	process	statistics	info	total
en	508	3 183	1 198	226	5 115
de	81	69	414	80	643
fr	63	59	83	14	218
nl	40	64	45	14	163
no	15	17	17	8	57
simple	4	4	3	2	14
nds	1	1	1	1	5

times-local.tsv, siehe ▸F.5

Tabelle 11.1.: Zeitaufwand für den Import



import-time-x-size.data, siehe ▸F.5

Abb. 11.1: Zeitaufwand für den Import (Minuten)

Der Import wurde auf einer Maschine mit vier Prozessorkernen (zwei Dual-Core AMD Opteron Prozessoren) mit 32 GB Arbeitsspeicher durchgeführt — allerdings wurden zur selben Zeit noch weitere Experimente auf diesem Computer durchgeführt. Sowohl der Prozessor als auch der Arbeitsspeicher waren während des Imports meist voll ausgelastet. Der Zeitaufwand für die einzelnen Schritte ist in Tabelle 11.1 angegeben, die Gesamtzeit ist in Abb. 11.1 dargestellt. Die benötigte Zeit ist aber vermutlich stark beeinflusst von der Zahl und der Art von Tätigkeiten, die gleichzeitig auf demselben Computer durchgeführt wurden. Um klare Aussagen über die Performanz machen zu können, wären Experimente auf einer Maschine notwendig, die keine anderen Aufgaben hat.

Dennoch lassen sich einige Beobachtungen machen: der Zeitaufwand scheint zwischen $O(n)$ und $O(n^2)$ zu liegen, wobei n die Anzahl der Artikel bezeichnet (siehe Tabelle 11.2). Dabei fällt das Verhältnis im unteren Bereich eher linear aus, während die großen Projekte zum Teil deutlich mehr Zeit benötigen. Insbesondere der Import der englischsprachigen Wikipedia dauert sehr lang (etwa dreieinhalb Tage), die deutschsprachige Wikipedia benötigt bei einem Drittel der Größe nur etwa ein Neuntel der Zeit. Das könnte darauf zurückzuführen sein, dass die englischsprachige Wikipedia einen hardware- oder konfigurationsbedingten Schwellwert übersteigt: zum Beispiel ist es möglich, dass für so große Projekte manche Datenbank-Indizes nicht mehr im Arbeitsspeicher gehalten werden können und damit deutlich mehr Festplattenzugriffe nötig werden. Um dies genauer zu untersuchen, wären weitere Experimente, insbesondere mit unterschiedlichen Konfigurationen des MySQL-Servers, notwendig.

11.2. Eigenschaften

	resource	concept	article	link	broader	langlink	meaning
en	4 936 730	6 319 639	2 007 044	58 707 859	6 888 843	3 587 574	12 932 545
de	1 316 554	2 127 028	656703	19 291 643	2 240 720	2 893 177	4 682 065
fr	1 446 357	2 108 017	508741	12 870 575	1 344 173	2 361 487	3 923 011
nl	640222	1 341 691	382550	8201706	667 251	2 595 570	2 187 664
no	267827	626982	156725	3610552	366 293	1 491 655	1 029 257
simple	46 050	196 372	25 217	558 408	46 920	656 398	294 936
nds	14 839	80 473	10 895	225 406	16 473	299 075	102 824

entries-local.tsv, siehe ▸F.5

Tabelle 11.2.: Anzahl von Einträgen in den lokalen Datensätzen

Der vorangegangene Abschnitt beschreibt, welche Daten verarbeitet wurden und wie viel Zeit dafür benötigt wurde. Hier soll nun ein grober Überblick über die Ergebnisse gegeben werden. Die Tabelle 11.2 zeigt, wie viele Einträge welcher Art in den lokalen Datensätzen erzeugt wurden. Die Tabelle 11.3 zeigt einige weitere Eigenschaften, Tabelle 11.4 zeigt relevante Verknüpfungen pro Artikel. Im Einzelnen sind folgende Werte aufgeführt:

`resource` ist die Anzahl der analysierten Seiten in dem Wikipedia-Projekt, also aller Seiten im Hauptnamensraum und im Kategorie-Namensraum. Das entspricht genau den Einträgen in der Tabelle `resource` (▸C.2).

	concepts	articles	aliases	defs	sections	uncat	nolang	navcat
en	6 319 639	2 007 044	2 243 103	1 983 155	337 583	93 140	1 211 836	222 562
de	2 127 028	656703	507878	649 086	113 993	14 073	252 462	31 013
fr	2 108 017	508741	626909	491 256	55 154	6 378	177 745	52 608
nl	1 341 691	382550	179414	377 442	13 958	921	87 004	23 032
no	626982	156725	86893	153 501	5 423	5 635	44 427	16 322
simple	196 372	25 217	10 733	23 897	512	543	591	4 394
nds	80 473	10 895	1 558	9 742	31	3 230	438	746

figures-local.tsv, siehe ▹F.5

Tabelle 11.3.: Weitere Eigenschaften

	links	categories	langlinks	terms
en	29,25	3,43	1,79	2,05
de	29,38	3,41	4,41	2,20
fr	25,30	2,64	4,64	1,86
nl	21,44	1,74	6,78	1,63
no	23,04	2,34	9,52	1,64
simple	22,14	1,86	26,03	1,50
nds	20,69	1,51	27,45	1,28

ref-per-article-local.tsv, siehe ▹F.5

Tabelle 11.4.: Verknüpfungen pro Artikel

concept ist die Gesamtzahl von Konzepten, ohne Weiterleitungen und Begriffsklärungen, aber mit allen Kategorien, Abschnitten und Konzepten, die zwar referenziert werden, aber keine eigene Seite haben. Das entspricht genau den Einträgen in der Tabelle concept (▹C.2).

Dabei ist zu beachten, dass die allermeisten Konzepte keinen eigenen Artikel haben: sie stammen aus „roten“ Wiki-Links, also von Verweisen auf noch nicht existierende Seiten, oder es handelt sich um Konzepte, die aus Verweisen auf Artikelabschnitte entstanden sind, oder um Navigationskategorien (▹9.1). Selbst in großen Projekten, in denen die meisten Wiki-Links in Artikeln „blau“ erscheinen, also auf eine vorhandene Seite zeigen, gibt es zu maximal 32% (en) der insgesamt referenzierten Konzepte einen Artikel. Für kleinere Projekte liegt diese Zahl noch deutlich niedriger, z. B. bei 25% (no) oder gar nur 10% (nds). Diese Zahl kann als Indikator für die „Reife“ eines Wiki-Projektes verstanden werden.

Insgesamt liefert WikiWord Informationen über mehrere Millionen Konzepte, für die englische Sprache allein über sechs Millionen (davon noch etwa zwei Millionen mit zugehörigem Artikel, also mit allen Relationen). Für jedes Konzept sind dabei im Schnitt zwei bis drei Terme bekannt (vergleiche Tabelle 11.4). Zum Vergleich: *WordNet* [Miller (1995), Fellbaum (1998), WordNet] enthält Anfang 2008 etwa 155 000 Wörter für 117 000 Konzepte (*Synsets*).

Dabei gibt es natürlich erhebliche inhaltliche Unterschiede: traditionelle Lexika, Wörterbücher und Wortnetze wie *WordNet* decken vor allem den Alltagswortschatz ab, mit Adjektiven, Verben und zum Teil auch Funktionswörtern; der von WikiWord aufgebaute Thesaurus enthält dagegen die Arten von Konzepten, die von der Wikipedia behandelt werden, nämlich vor allem Nomen: neben allgemeinen Konzepten wie Haus oder Kultur sehr viele *Named Entities*, also Eigennamen von Personen, Orte, Gattungen von Lebewesen, Organisationen, sowie eine große Zahl von Fachwörtern (vergleiche Tabelle

12.3). Insofern stellen die Konzepte, die Wikipedia behandelt, eine nützliche Ergänzung dar zu den Konzepten des Alltagswortschatzes, den traditionelle Wörterbücher bieten (vergleiche [Ruiz-Casado u. a. (2005), Zesch u. a. (2007b), Toral und Munoz (2007)]). Ein eingehender Vergleich des von WikiWord erstellten Thesaurus mit WordNet verbleibt als Gegenstand künftiger Forschung.

`article` ist die Zahl von Artikeln bzw. von Konzepten, die eine eigene Wiki-Seite haben. Das sind diejenigen Einträge in der Tabelle `concept`, die *nicht* den Typ `UNKNOWN` haben. Artikel sind in der Anzahl der Konzepte enthalten.

Diese Anzahl unterschiedet sich deutlich von dem, was MediaWiki als „Artikel“ zählt, und somit auch von der Zahl, die zur Angabe der Größe eines Wiki-Projektes von Wikimedia verwendet wird: MediaWiki zählt solche Seiten als Artikel, die im Hauptnamensraum liegen, keine Weiterleitung sind, aber mindestens einen Wiki-Link enthalten. WikiWord zählt Seiten im Hauptnamensraum, die keine Weiterleitung, keine Begriffsklärung und keine Liste sind, es wird aber nicht verlangt, dass sie einen Wiki-Link enthalten. So ergeben sich zum Teil erhebliche Unterschiede; Insbesondere die französischsprachige Wikipedia, die nach der Zählweise von MediaWiki mit der deutschsprachigen fast gleich auf ist, wirkt hier deutlich kleiner.

Zu beachten ist auch, dass für die norwegischsprachige Wikipedia (`no`) keine projektspezifischen Analyseregeln definiert wurden. Insbesondere die Ressourcenklassifikation schlägt daher fehl, Begriffsklärungsseiten werden nicht als solche erkannt. Das verzerrt die Ergebnisse für dieses Projekt teilweise.

`link` ist die Anzahl von Querverweisen zwischen Konzepten, das heißt die Anzahl von Einträgen in der Tabelle `link` (▹C.2), deren `anchor`-Feld nicht `NULL` ist (also ohne diejenigen Einträge, die *nur* eine Termzuordnung definieren).

Die Anzahl von Wiki-Links pro Artikel (also der durchschnittliche Knotengrad im Hyperlink-Graphen) bewegt sich zwischen 20 (`nds`) und 30 (`en`) und ist grob mit der Größe des Projektes, also der Anzahl der Artikel korreliert. Allerdings ist die deutschsprachige Wikipedia gleichauf mit der englischsprachigen, obwohl sie nur ein Drittel der Artikel hat. Und die norwegischsprachige Wikipedia hat mehr Links pro Seite als die niederländischsprachige, obwohl letztere mehr als doppelt so groß ist. Es wäre zu untersuchen, ob sich die Korrelation nicht vielmehr auf die Länge der Artikel oder das Alter der Seiten bezieht, wie es das Modell des *Preferential Attachment* nahelegt [Barabasi und Albert (1999)]. Die Verteilung der Knotengrade wird in ▹11.4 untersucht.

`broader` ist die Anzahl der Über- bzw. Unterordnungen, also die Anzahl der Einträge in der Tabelle `broader` (▹C.2). Die Anzahl von Kategorisierungen (Unterordnungen) pro Artikel variiert zwischen 1,5 (`nds`) und 3,4 (`en`) und kann als Indikator für die Dimensionalität der Kategoriestruktur interpretiert werden, also für die Anzahl der Kategorisierungskriterien. Diese Zahl korrespondiert nur grob mit der Größe des Projektes, so hat zum Beispiel `simple` mit 1,9 mehr Kategorien pro Artikel als `nl` mit 1,7, obwohl letzteres Projekt etwa fünfzehnmal größer ist. Ein Grund dafür mag sein, dass `simple` seine Kategoriestruktur zu großen Teilen aus der englischsprachigen Wikipedia übernimmt, und dadurch eine für ein so kleines Projekt untypisch hohe Granularität an Kategorien besitzt.

langlink ist die Anzahl der Language-Links, also die Anzahl der Einträge in der Tabelle `langlink` (▹C.2). Die durchschnittliche Anzahl von Language-Links pro Artikel variiert stark zwischen nur 1,8 (`en`) und 27,5 (`nds`) — dies ist ein Indikator für die Granularität der einzelnen Projekte: da die englische Wikipedia (`en`) eine sehr hohe Granularität hat, enthält sie viele Seiten, für die es in keiner oder nur in wenigen anderen Sprachen eine Entsprechung gibt; dagegen gibt es für die meisten Seiten in der plattdeutschen Wikipedia (`nds`) eine Entsprechung in sehr vielen Sprachen.

Dabei ist zu beobachten, dass es zu gewissen Konzepten in so gut wie allen Wikipedias Artikel gibt. Dazu zählen vor allem Jahreszahlen wie 1984 und Kalendertage wie 1. Mai, aber auch die Kontinente sowie essentielle Konzepte des Alltags wie Wasser oder Sonne. Zählt man in verschiedenen Projekten alle Seiten, die mehr als 100 Language-Links haben, ergibt sich jedes Mal eine Zahl um 760.

meaning ist die Anzahl der Termzuordnungen (Paare von Bezeichnung und Bedeutung), also die Anzahl der Einträge in der Tabelle `meaning` (▹C.2). Die von WikiWord vorgenommene Zuordnung von Termen zu Konzepten wird in ▹12.5 näher untersucht.

terms ist die Anzahl der Terme bzw. Termzuordnungen pro Konzept (inklusive der Konzepte ohne Artikel), also die Anzahl von unterschiedlichen Bezeichnungen. Die Anzahl der Terme pro Konzept scheint grob mit der Anzahl der Links pro Artikel zu korrespondieren — da die Terme zu einem großen Teil aus Link-Texten gewonnen werden, und die Anzahl unterschiedlicher Link-Texte mit der Gesamtzahl von Wiki-Links wächst, ist dieser Zusammenhang naheliegend.

aliases ist die Anzahl von Aliasen, die auf die Konzepte angewendet wurden. Das entspricht genau den Einträgen in der Tabelle `alias` (▹C.2). Aliase sind in der Anzahl der Konzepte und Artikel nicht enthalten. Der Anteil von Weiterleitungen an der Gesamtzahl der verarbeiteten Seiten liegt zwischen 10% (`nds`) und 45% (`en`). Die große Zahl von Weiterleitungen in der englischsprachigen Wikipedia rührt unter anderem daher, dass dort Weiterleitungen für Pluralformen und für alternative Groß-/Kleinschreibung und Durchkopplungen üblich sind. Es lässt sich aber generell die Tendenz erkennen, dass größere Projekte einen höheren Anteil von Weiterleitungen haben.

Auch diese Größe mag als Indikator für die „Reife“ eines Wiki-Projektes dienen, allerdings haben sprach- oder projektspezifische Konventionen einen großen Einfluss auf diese Zahl. Ein Beispiel ist die Berücksichtigung des *„Title Case“* in der englischsprachigen Wikipedia: *Principia_mathematica* ist ein Alias für *Principia_Mathematica*, ähnliche Weiterleitungen gibt es für eine große Zahl von Namen von Werken und Organisationen.

defs ist die Anzahl der extrahierten Definitionen. Das sind die Einträge in der Tabelle `definition` (▹C.2). Die Extraktion von Definitionssätzen gelingt für etwa 90% (`nds`) – 99% (`en`) der Artikel. Das schlechte Abschneiden der plattdeutschen Wikipedia (`nds`) ist unter anderem dadurch zu erklären, dass in diesem recht kleinen Projekt Artikel zu Jahreszahlen und Kalendertagen einen großen Anteil ausmachen — solche Artikel haben in der Regel keinen Definitionssatz.

sections ist die Anzahl von Konzepten, die aus Verweisen auf Abschnitte entstanden sind. Bis zu 5% (en, de) der Konzepte entstehen auf diese Weise, wobei der Anteil für kleinere Projekte geringer ist. Das ist vermutlich dadurch zu erklären, dass die Artikel in kleineren Projekten auch kürzer sind und es daher weniger Abschnitte gibt, auf die verwiesen werden kann. Ausserdem beschränken sich junge Projekte meist auf den Aufbau einer Grundstruktur, „rote“ Wiki-Links auf noch nicht vorhandene Artikel kommen dabei sehr häufig vor. In älteren, größeren Projekten ist die Wahrscheinlichkeit höher, dass Autoren einen Abschnitt eines Artikels als Link-Ziel wählen, wenn es keinen eigenen Artikel zu dem betreffenden Thema gibt.

uncat ist die Anzahl unkategorisierter Artikel. Der Anteil von Artikeln ohne Kategorien liegt zwischen 0,2% (nl) und 4,6% (en), mit der Ausnahme von (nds), wo der Anteil bei 30% liegt. Der relative große Anteil unkategorisierter Artikel in der englischsprachigen Wikipedia ist vermutlich darauf zurückzuführen, dass es dort viele sogenannte *„Stubs“* gibt, also sehr kurze „Artikelstummel“. Diese sind per Konvention mit einer Vorlage markiert, die, im Falle der englischsprachigen Wikipedia, auch automatisch eine Kategorisierung vornimmt — diese wird, da sie implizit ist, von Wikiword nicht erfasst. Die sehr hohe Zahl unkategorisierter Artikel in der plattdeutschen Wikipedia dagegen ist wohl dem Umstand geschuldet, dass es sich um ein sehr kleines und recht junges Projekt mit einer geringen Zahl an Mitarbeitern handelt. Sie kann als Indikator dafür verstanden werden, dass dieses Wiki-Projekt noch recht „unreif“ ist.

nolang ist die Anzahl von Artikeln ohne Language-Links. Der Anteil von Artikeln, die keine Language-Links auf entsprechende Artikel in anderen Sprachen haben, variiert sehr stark, nämlich zwischen 2% (simple) und 60% (en). Das ist ein Indikator für die unterschiedliche Granularität der einzelnen Projekte: die englischsprachige Wikipedia hat die mit Abstand größte Zahl an Artikeln und damit die höchste Granularität — naturgemäß enthält sie daher viele Artikel, die keine Entsprechung in anderen Sprachen haben. Der Anteil von 60% entspricht dabei gut dem Umstand, dass die englischsprachige Wikipedia dreimal so groß ist wie die nächstkleinere, nämlich die deutschsprachige — allerdings heißt das nicht, dass ein Drittel der Artikel einen Language-Link auf die deutschsprachige Wikipedia enthalten müssen. Die von den einzelnen Projekten abgedeckten Konzepte überlappen sich ja nicht völlig, so hat auch die deutschsprachige Wikipedia noch 40% Artikel ganz ohne Language-Links. Viele davon werden sich auf Personen, Orte und Organisationen beziehen, die vorrangig von lokaler Bedeutung sind — dies wäre noch näher zu untersuchen.

navcat ist die Anzahl von Navigationskategorien, also von Konzepten, die keinen Artikel haben, denen aber andere Konzepte untergeordnet sind. Navigationskategorien machen zwischen 1% (nds) und 3,5% (en) der Konzepte aus.

11.3. Multilinguale Thesauri

Nachdem alle monolingualen Thesauri aufgebaut waren (▹11.1), wurden auf dieser Basis mittels des Programms BuildThesaurus multilinguale Thesauri aufgebaut (▹9.2): einmal auf Basis

	copy	prepare	merge	process	statistics	info	total
compl.	98	38	467	525	462	265	1 855
top5	121	33	207	176	550	303	1 391
top3	155	28	122	361	446	264	1 377

times-global.tsv, siehe ▸F.5

Tabelle 11.5.: Zeitaufwand für das Zusammenführen

	concept	article	link	broader	langlink	orig.article
compl.	11 835 474	2 783 147	75 650 558	11 565 395	5 086 084	3 747 875
top5	11 589 229	2 777 635	75 192 652	11 500 968	5 042 949	3 711 763
top3	9 979 450	2 597 254	68 623 989	10 469 641	4 725 484	3 172 488

entries-global.tsv, siehe ▸F.5

Tabelle 11.6.: Anzahl von Einträgen in den globalen Datensätzen

aller Sprachen (compl.), einmal aus den größten drei, also Englisch, Deutsch und Französisch (top3), und einmal aus den größten fünf, also Englisch, Deutsch, Französisch, Niederländisch und Norwegisch (top5). Der Aufbau ist in vier Schritte aufgeteilt: copy, prepare, merge und process.

Der Zeitaufwand für das Zusammenführen der Konzepte fällt mit etwa einem Drittel der Zeit, die für den Import der einzelnen Dumps benötigt wurde, relativ gering aus (Tabelle 11.5). Auch scheint er kaum von der Anzahl der zu betrachtenden Sprachen abzuhängen, sondern vor allem von der Gesamtzahl der Artikel: die drei multilingualen Thesauri mit sieben, fünf und drei Sprachen brauchen jeweils etwas weniger als einen Tag. Darüber hinaus lassen sich kaum Aussagen treffen, da die Werte stark fluktuieren und davon auszugehen ist, dass sie vor allem durch die Systemlast zur Zeit der Verarbeitung beeinflusst sind. Zum Beispiel werden im Schritt copy nur Daten aus den lokalen Datensätzen in den globalen Datensatz kopiert, um dann verarbeitet zu werden. Der Zeitaufwand sollte also nur davon abhängig sein, wie viel zu kopieren ist. Die Tabelle 11.5 zeigt aber das Gegenteil: das Kopieren von Daten aus drei Datensätzen (top3) dauert deutlich länger als das Kopieren aus sieben Datensätzen (compl.). Das lässt sich nur durch eine deutlich schlechtere Performanz bzw. höhere Last des Gesamtsystems zu dem fraglichen Zeitpunkt erklären.

Die Tabelle 11.6 zeigt, wie viele Einträge welcher Art in den globalen Datensätzen erzeugt wurden, Tabelle 11.7 zeigt die Anzahl von Verknüpfungen zwischen den Konzepten bzw. Artikeln.

Die dargestellten Eigenschaften entsprechen den in ▸11.2 beschriebenen, mit dem Zusatz des orig.article-Feldes:

	links	categories	langlinks
compl.	27,18	4,16	1,83
top5	27,07	4,14	1,82
top3	26,42	4,03	1,82

re-per-article-global.tsv, siehe ▸F.5

Tabelle 11.7.: Verknüpfungen pro Artikel

orig.article ist die Anzahl von lokalen Artikeln, die in einem globalen Datensatz berücksichtigt sind. Das entspricht genau den Einträgen in der Tabelle origin (▹C.2),die auf ein Konzept verweisen, dessen Typ nicht UNKNOWN ist.

Es handelt sich also um die Anzahl der Artikel vor dem Zusammenfassen äquivalenter Konzepte. Vergleicht man diese Zahl mit der Zahl der Artikel, die nach dem Zusammenfassen übrig bleiben, ergibt sich ein Maß dafür, wie effektiv der Zusammenführungsalgorithmus arbeitet: für die globalen Datensätze compl. und top5 ergibt sich eine Reduktion um 25%, für den Datensatz top3 um 20%. Wenn man bedenkt, dass gut die Hälfte der Artikel gar keine Language-Links hat, also auch mit nichts zusammengeführt werden können, scheint dieses Ergebnis recht gut. In der Tat verbleiben nur für weniger als 10% der Konzepte direkte Language-Links im Feld langref der Tabelle relation (▹C.2), also potentielle Äquivalenzen, die nicht verwendet wurden (so genannte *Konflikte*). Insgesamt sind 15% – 20% der verbleibenden Artikel aus mehr als einem lokalen Konzept zusammengesetzt — in Anbetracht der Tatsache, dass über 60% nur in der englischsprachigen Wikipedia existieren, ist das ein ermutigendes Ergebnis.

Insgesamt liefert WikiWord Übersetzungen für etwa eine halbe Million Konzepte, eine viertel Million davon mit mehr als zwei Sprachen und immer noch eine achtel Million mit mehr als drei. Für jedes Konzept sind dabei im Schnitt zwei bis drei Terme pro Sprache bekannt. Zum Vergleich: das *PONS Großwörterbuch Englisch*[2] hat in der aktuellen Ausgabe 390 000 Stichwörter und 530 000 Übersetzungen. Auch hier ist natürlich, wie schon im Vergleich mit WordNet, zu berücksichtigen, dass ein traditionelles Übersetzungswörterbuch ganz andere Konzepte abdeckt, als die, die WikiWord aus der Wikipedia extrahiert.

In ▹9.2 ist beschrieben, wie jeweils (höchstens) ein Konzept aus jedem monolingualen Thesaurus (also aus jeder Sprache) zu einem sprachübergreifenden Konzept zusammengeführt wird. Ein globales Konzept in einem multilingualen Thesaurus ist also eine Gruppierung von äquivalenten lokalen Konzepten, wobei in jeder Gruppe höchstens ein Repräsentant jeder Sprache vertreten ist.

Eine direkte Evaluation des Algorithmus, der in ▹9.2 verwendet wird, um solche Gruppen von Konzepten zu finden, gestaltet sich schwierig: aufgrund der unterschiedlichen Granularität der Projekte gibt es oft mehrere Möglichkeiten, Konzepte zusammenzufassen, und es ist kaum möglich, anzugeben, welche Gruppierung optimal wäre. Unmittelbar gezählt werden können nur Fälle, in denen sich offensichtlich ein Konzept in der Gruppe befindet, das dort nicht hineingehört und umgekehrt, wenn in der Gruppe ein Konzept fehlt. Letzteres kommt ab und zu vor (in einer Stichprobe von 250 Konzepten insgesamt 6 mal, siehe Anhang F.4), der Grund ist in der Regel ein fehlender Language-Link in einem der betreffenden Artikel.

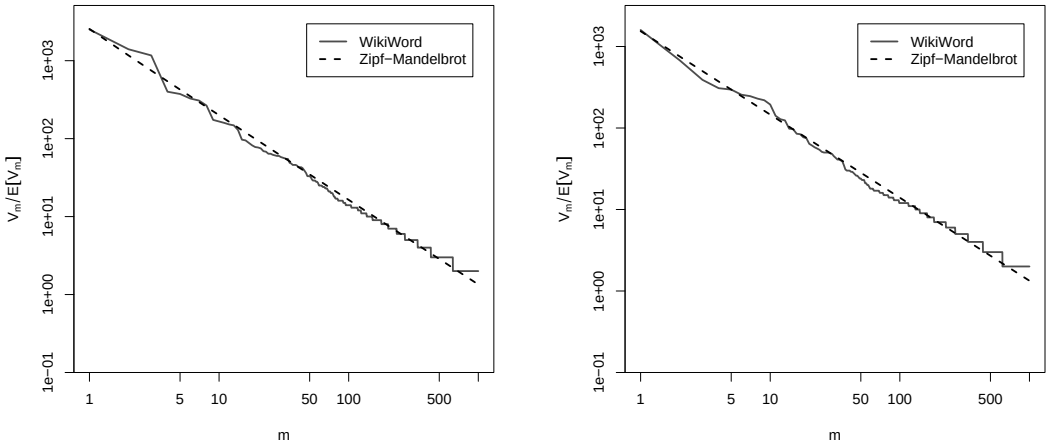
Der Fall, dass ein „falsches“ Konzept aufgenommen wurde, kommt dagegen kaum vor (in der Stichproben gar nicht, siehe Anhang F.4). Im direkten Vergleich zwischen multilingualen Thesauri auf unterschiedlicher Datenbasis lassen sich aber die Auswirkungen der unterschiedlichen Granularität beobachten: die „Trennschärfe“ der impliziten Ähnlichkeitsbeziehung, die durch die Language-Links ausgedrückt wird, ist bei kleinen Projekten deutlich schlechter (vergleiche ▹12.3). Sie fassen häufig mehr Konzepte in einem Artikel zusammen. Werden sie für den Aufbau des multilingualen Thesaurus verwendet, so besteht die Gefahr, dass Konzepte, die nicht völlig äquivalent sind, zu einem sprachübergreifenden Konzept zusammengeführt werden, weil sie

[2] ISBN: 978-3-12-517192-3, <http://www.pons.de/produkte/3-12-517192-X/>

eben über Language-Links miteinander verbunden sind. So wurde zum Beispiel SIMPLE:HEIGHT mit EN:ELEVATION zusammengefasst, obwohl SIMPLE:ELEVATION korrekt gewesen wäre.

Zusätzlich zu dem Vergleich von Thesauri auf der Basis von drei, fünf und sieben Sprachen wäre es interessant, alternative Algorithmen für das Finden und Zusammenfassen von „äquivalenten“ Konzepten zu vergleichen. Eine alternative Methode, die die Language-Links der Seiten als Featurevektoren interpretiert, wurde in ▸9.2.2 bereits kurz beschrieben. Als Baseline könnte desweiteren ein naiver Algorithmus dienen, der die englischsprachige Wikipedia als Ausgangspunkt nimmt und jedes Konzept aus diesem Projekt mit allen Konzepten aus anderen Sprachen vereinigt, mit denen es über Language-Links verknüpft ist. Ein Vergleich dieser unterschiedlichen Methoden konnte im Rahmen dieser Arbeit nicht umgesetzt werden und verbleibt als Gegenstand zukünftiger Forschung.

11.4. Verteilung von Termfrequenzen



(a) en.wikipedia.org (b) de.wikipedia.org

`dewiki-terms-zipf.data`, siehe ▸F.5 bzw. `enwiki-terms-zipf.data`, siehe ▸F.5

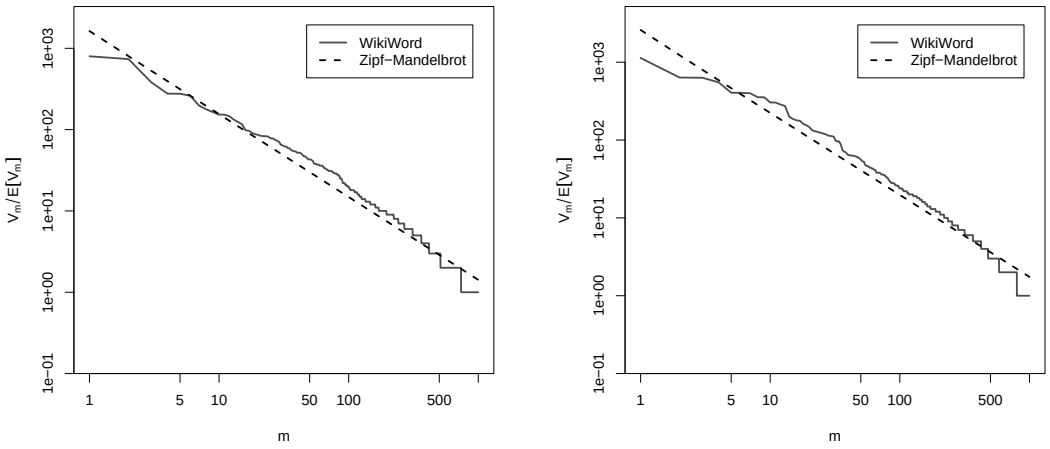
Abb. 11.2: Verteilung der Termfrequenzen

In diesem Abschnitt wird untersucht, wie die von WikiWord gefundenen Terme bezüglich ihrer Häufigkeit (Frequenz) verteilt sind. Es wird erwartet, dass diese Verteilung derjenigen ähnelt, die sich für die Wortfrequenzen in natürlichen Texten ergibt. Die Verteilung von Wortfrequenzen folgt typischerweise dem *Zipfschen Gesetz* [ZIPF (1935)], und Abb. 11.2 zeigt, dass das in der Tat auch für die von WikiWord gefundenen Terme gilt[3]. Die beobachtete Verteilung passt sogar noch besser zu der Vorhersage, als das häufig für Wortfrequenzen aus Volltext der Fall ist: Die Verteilung von

[3]Die Statistiken wurden mit dem *ZipfR*-Paket [ZIPFR] für *GNU R* [GNU-R] ausgewertet und geplottet, für die Regressionsanalyse wurde ein *Zipf-Mandelbrot-Modell* verwendet [EVERT (2004)].

Stoppwörtern (typischerweise etwa die obersten 100 Ränge) folgen häufig nicht der Vorhersage des Zipfschen Gesetzes (was durch die Verfeinerung von Mandelbrot teilweise ausgeglichen wird). Die Menge von Termen, die WikiWord liefert, enthält aber keine Stoppwörter (oder wenn, dann nur als „Fehler“ und daher mit niedriger Frequenz): die Terme stammen vor allem aus Seitentiteln und aus Wiki-Links (▹8.9) und sind daher in der Regel stark sinntragend [Kraft und Zien (2004), Eiron und Mccurley (2003)]. Ihre Verteilung folgt daher sehr genau den Voraussagen des Zipfschen Gesetzes.

11.5. Verteilung von Knotengraden



(a) en.wikipedia.org (b) de.wikipedia.org

`enwiki-degree-zipf.data`, *siehe* ▹*F.5* *bzw.* `dewiki-degree-zipf.data`, *siehe* ▹*F.5*

Abb. 11.3: Verteilung der Knotengrade

Die graphentheoretischen Eigenschaften der Wikipedia als Hyperlink-Netzwerk wurden schon verschiedentlich untersucht [Voss (2005b), Capocci u. a. (2006), Bellomi und Bonato (2005b)]. Ein wichtiges, aber kaum überraschendes Ergebnis dieser Untersuchungen war, dass es sich bei dem Hyperlink-Netzwerk in Wikipedia um einen *skalenfreien Small-Word-Graphen* handelt [Capocci u. a. (2006)], so wie das bei den meisten Hypertext-Korpora der Fall ist [Barabasi und Albert (1999), Steyvers und Tenenbaum (2005)]. Die Skalenfreiheit ergibt sich dabei aus dem „natürlichen“ Wachstum des Netzwerkes, das dem Gesetz des *Preferential Attachment* folgt.

Der Nachweis dieser Eigenschaften soll hier nicht wiederholt werden. Lediglich die Skalenfreiheit des Hyperlink-Netzwerk wird in Abb. 11.3 demonstriert: hier ist zu erkennen, dass die Verteilung der Knotengrade dem *Power Law* folgt[4].

[4] Auch hier wurde mittels *ZipfR* eine Regressionsanalyse mit dem *Zipf-Mandelbrot-Modell* verwendet.

11.6. Auswertung des Exports

Datensatz	WikiWord	SKOS
en	120 300 556	60 720 983
de	41 965 891	20 798 761
fr	37 141 654	16 840 521
nl	23 332 076	10 146 600
no	11 199 935	4 692 075
simple	3 010 005	1 204 964
nds	1 299 067	463 783
top3	232 616 737	138 375 761
top5	269 441 309	159 703 396
comp.	274 464 220	162 442 796

`export-size.tsv`, *siehe* ▸F.5

Tabelle 11.8.: Anzahl von RDF-Tripeln

Der Export von WikiWord-Daten nach RDF bzw. SKOS, wie er in Kapitel ▸10 beschrieben ist, kann für sich genommen kaum evaluiert werden. Hier wären Integrationstests notwendig, die die Kompatibilität und Nützlichkeit dieser Daten gezielt in Bezug auf andere Systeme wie *DBpedia* [Auer u. a. (2007), Auer und Lehmann (2007), DBpedia], *OmegaWiki* (ehemals *WiktionaryZ*) [van Mulligen u. a. (2006), OmegaWiki] oder *Wortschatz* [Quasthoff (1998), Wortschatz] untersuchen. Solche Untersuchungen würden über den Rahmen dieser Arbeit hinausgehen und verbleiben als Gegenstand künftiger Forschung.

Um die Exportfunktion zu testen und einen Eindruck von der zu erwartenden Datenmenge zu erhalten, wurden die aus den Wikipedias extrahierten Daten unter Verwendung verschiedener Optionen als RDF exportiert (für Zugang zu den resultierenden Daten, siehe ▸F.3).

Die Tabelle 11.8 zeigt die Anzahl von *RDF-Statements* (*Tripeln*), die zur Beschreibung der mono- und multilingualen Thesauri benötigt werden. Bei Nutzung des WikiWord-Vokabulars ergibt sich ein Durchschnitt von etwa 23 Statements pro Konzept, bei einer Beschränkung auf SKOS sind es mit etwa 14 deutlich weniger. Die Gesamtgröße eines multilingualen Thesaurus mit den oben angegebenen sieben Sprachen, in exportierter Form, unter Verwendung des WikiWord-Vokabulars und mit *bzip2* komprimiert, beläuft sich auf 1,3 Gigabyte. Zum Vergleich: die zugrunde liegenden Wikipedia-Dumps für die entsprechenden sieben Projekte haben zusammen eine Größe von knapp 6 Gigabyte, ebenfalls unter Verwendung von *bzip2*.

12. Akkuratheit der Konzepte

Dieses Kapitel untersucht die Eigenschaften und Beziehungen von Konzepten. Dazu werden aus zwei Thesauri, dem deutschsprachigen und dem englischsprachigen, jeweils zwei Stichproben entnommen: eine mit zufälligen Konzepten aus der Gesamtmenge der Konzepte (en_{all} mit 102 Konzepten bzw. de_{all} mit 101), und eine mit zufälligen Konzepten aus den „Top 10%“, den am häufigsten verwendeten zehn Prozent der Konzepte (en_{top} bzw. de_{top}, jeweils mit 51 Konzepten) unter Verwendung des `in_rank`-Wertes. Dabei ist zu bedenken, dass die Stichprobe über alle Konzepte auch viele Konzepte aus den 70% enthält, die gar keinen Artikel haben — einige der hier untersuchten Eigenschaften treten bei solchen artikellosen Konzepten gar nicht auf.

Es wird erwartet, dass die Stichprobe über die „Top 10%“ gebräuchliche, allgemeine Konzepte enthält und dass die betreffenden Artikel von guter Qualität sind, das heißt, den Richtlinien der Wikipedia und der Syntax von MediaWiki folgen. Insbesondere ist zu erwarten, dass sich die 70% der Konzepte ohne Artikel am Ende der Rangliste nach `in_rank` befinden (meist werden sie nur ein- oder höchstens zweimal direkt referenziert, zum Teil, z. B. bei Navigationskategorien, gar nicht) — das bedeutet, dass die meisteverwendeten 10% der *Konzepte* größtenteils in die meistverwendeten 33% der *Artikel* fallen, da die Artikel mehr oder weniger die oberen 30% der Konzepte ausmachen.

Die Evaluation der Konzepte ist in `ConceptSurvey` implementiert und basiert auf den folgenden Schritten:

1. In der Tabelle `degree` wird für das Feld `in_rank` das gewünschte Intervall bestimmt, also $[1 \ldots n]$ für die Stichprobe über alle Konzepte und $[1 \ldots n/10]$ für die Stichprobe aus den Top 10%.

2. Eine Zahl aus diesem Intervall wird zufällig gewählt und das Konzept, das diese Zahl als `in_rank` hat, wird geladen. Dabei werden die Informationen aus `concept_info` verwendet, um effizient Zugang zu allen Eigenschaften und Beziehungen zu erhalten.

3. Für jedes Konzept wird eine Anzahl von Eigenschaften (*Properties*) untersucht, wobei manche Eigenschaften mehr als einen Wert haben können. Die untersuchten Eigenschaften sind:

   `terms`: Die Terme (▹8.9), die dem Konzept über die Bedeutungsrelation zugeordnet sind (aus der Tabelle `meaning` ▹C.2).

   `type`: Der Konzepttyp (▹8.6), der dem Konzept zugeordnet ist (aus der Tabelle `concept` ▹C.2).

   `definition`: Die Definition (▹8.8), die dem Konzept zugeordnet ist (aus der Tabelle `definition` ▹C.2).

   `broader`: Übergeordnete Konzepte (▹9.1), die dem Konzept über die Subsumtionsbeziehung zugeordnet sind (aus der Tabelle `broader` ▹C.2).

   `narrower`: Untergeordnete Konzepte (▹9.1), die dem Konzept über die Subsumtionsbeziehung zugeordnet sind (aus der Tabelle `broader`).

related: Verwandte Konzepte (▹9.1.3), die dem Konzept zugeordnet sind (aus der Tabelle `relation` ▹C.2).

similar: Ähnliche Konzepte (▹9.1.3), die dem Konzept zugeordnet sind (aus der Tabelle `relation`).

Diese Eigenschaften entsprechen denen, die im Datenmodell festgelegt sind (▹D.2, ▹C.1).

4. Jedem Wert jeder Eigenschaft von jedem Konzept wurde nun manuell eine Bewertung zugewiesen, die besagt, ob und inwieweit der gegebene Wert korrekt ist (siehe Anhang F.4 für die Rohdaten der Erhebung). Welche Bewertungen möglich sind, hängt dabei von der betreffenden Eigenschaft ab.

Im Folgenden wird die Qualität der Werte für die einzelnen Eigenschaften analysiert.

12.1. Auswertung der Klassifikation

Bei der Konzeptklassifikation (▹8.6) wird jedem Konzept ein Typ aus einer fest vorgegebenen Menge von Typen wie `PERSON` oder `PLACE` zugewiesen. Sie bietet also eine grobe, korpusunabhängige Aufteilung der Konzepte, die verwendet werden kann, um Konzepte für eine bestimmte Anwendung zu filtern. Zum Beispiel können für die Erstellung eines Wörterbuches Personen ignoriert werden, oder es können für die Erstellung eines *Gazetteer* gezielt Orte verwendet werden. Konzepte, die zu keiner der vorgegebenen Klassen passen, erhalten den Typ `OTHER`, solche, zu denen es keinen eigenen Artikel gibt und eine Klassifikation daher nicht möglich ist, erhalten den Typ `UNKNOWN`.

(a) gesamt

Typ	en	de
UNKNOWN	68%	60%
PLACE	2%	3%
PERSON	8%	10%
ORGANISATION	>1%	>1%
NAME	>1%	>1%
TIME	>1%	>1%
NUMBER	>1%	>1%
LIFEFORM	1%	>1%
OTHER	21%	17%

(b) ohne `UNKNOWN`

Typ	en	de
UNKNOWN		
PLACE	7%	11%
PERSON	24%	31%
ORGANISATION	>1%	1%
NAME	>1%	>1%
TIME	>1%	>1%
NUMBER	>1%	>1%
LIFEFORM	4%	2%
OTHER	65%	54%

Tabelle 12.1.: Häufigkeit der Konzepttypen

Zur Evaluation der Konzeptklassifikation stehen vier Werte als mögliche Beurteilungen zur Verfügung:

UNKNOWN wird immer automatisch für den Typ `UNKNOWN` gewählt — wenn der Typ nicht bestimmt werden kann, weil es zu dem Konzept keinen Artikel gibt, dann ist die Beurteilung gegenstandslos.

(a) gesamt

Typ	en_{all}	en_{top}	de_{all}	de_{top}
UNKNOWN	58%	2%	72%	2%
OK	36%	92%	28%	92%
MISSING	6%	4%	0%	6%
WRONG	0%	2%	0%	0%

(b) ohne UNKNOWN

Typ	en_{all}	en_{top}	de_{all}	de_{top}
UNKNOWN				
OK	86%	92%	100%	92%
MISSING	14%	4%	0%	6%
WRONG	0%	2%	0%	0%

Tabelle `concept_survey`*, siehe* ▹F.4

Tabelle 12.2.: Bewertung der Konzeptklassifikation

OK wird benutzt, wenn der Typ korrekt erkannt wurde.

MISSING gibt an, dass kein konkreter Typ zugewiesen wurde (also der Typ OTHER ist), obwohl das Konzept eindeutig zu einer der vorgegebenen Klassen gehört.

WRONG gibt an, dass der erkannte Typ falsch ist.

Bei der Auswertung zeigt sich zunächst, dass die allermeisten Konzepte korrekt und fast gar keine irrtümlich zugeordnet werden. Allerdings wird der Typ einiger Konzepte, die eigentlich klassifiziert werden sollten, nicht erkannt — daraus wird sichtbar, dass es sich bei der Klassifikation um eine Heuristik handelt, die nach dem *Best-Effort*-Prinzip arbeitet: in der betreffenden Unterklasse von `WikiConfiguration` sind in dem Feld `conceptTypeSensors` „Sensoren“ registriert (▹8.2), die *sichere* Hinweise für die einzelnen Konzepttypen erkennen, zumeist die Verwendung bestimmter Kategorien oder Vorlagen. Die Erkennung basiert also auf *Konventionen* bezüglich der Kategorisierung und Gestaltung von Artikeln über Konzepte einer bestimmten Art — wie effektiv die Heuristiken arbeiten, hängt also davon ab, wie konsistent die betreffenden Konventionen in den Artikeln der Wikipedia befolgt werden, und wie gut sie durch die angegebenen Muster erkannt werden.

Auch prinzipiell ist das Zuweisen der Konzepttypen nicht trivial, weil sie oft nicht eindeutig abzugrenzen sind. Zum Beispiel ist unklar, was als Ort zu klassifizieren ist: Länder und Städte sicher, aber Flüsse, Flughäfen, bestimmte Gebäude? Ähnlich verhält es sich mit Organisationen. Personen sind dagegen relativ leicht abzugrenzen, es kann höchstens passieren, dass eine *fiktive* Person mit aufgenommen wird. Artikel über Zeitabschnitte (Jahre, Jahrzehnte und so weiter) sowie biologische Taxa sind im allgemeinen leicht zu erkennen, da sie in den meisten Wikipedias durchgehend typische Vorlagen verwenden: *Taxoboxen* bei Taxa und Vorlagen zur Kalendernavigation in Artikeln über Zeitabschnitte. Problematisch wird es wieder bei Namen: in manchen Wikipedias dienen Artikel über Vor- oder Nachnamen gleichzeitig als Begriffsklärung oder Liste aller (bekannter) Namensträger. Es kann also sein, dass sie gar nicht erst als Artikel erkannt sondern als Begriffsklärung oder Liste behandelt werden (▹8.5).

12.2. Auswertung der Subsumtion

Die Subsumtionsbeziehung ist ein wichtiger Bestandteil eines Thesaurus: sie ist das strukturierende Element, das es erlaubt, ein Konzept durch Eingrenzung zu finden, also durch Navigation einer Hierarchie von Konzepten, ausgehend von den allgemeineren zu den spezielleren. Außerdem ermöglicht sie bei Bedarf ein automatisches *Fallback* auf allgemeinere Konzepte, zum Beispiel bei einer *Query Expansion*.

Zur Bewertung der Subsumtionsbeziehungen eines Konzeptes, also seiner Verknüpfung mit über- bzw. untergeordneten Konzepten, stehen nur zwei Alternativen zur Verfügung: `RIGHT` und `WRONG`. Es wird also nur festgestellt, ob die Unterordnung eines Konzeptes unter ein anderes berechtigt scheint. Dabei werden die Bewertungen für die Eigenschaften `narrower` und `broader` zusammengefasst.

Typ	en_{all}	en_{top}	de_{all}	de_{top}
`RIGHT` gesamt	99%	99%	96%	98%
`RIGHT` ohne `UNKNOWN`	99%	99%	96%	98%

Tabelle `concept_survey`, *siehe* ▹F.4

Tabelle 12.3.: Bewertung der Konzeptklassifikation

Die Tabelle 12.3 zeigt, dass fast alle Unterordnungen korrekt sind. Fälschliche Unterordnungen basieren zum einen auf falschen Angaben in Wikipedia-Artikeln (die meist daher rühren, dass die Bedeutung der zugeordneten Kategorie missverstanden wurde), zum andern aber auch an falsch zugeordneten Kategorisierungs-Aliasen ▹9.1.2. Zum Beispiel wurden für die deutschsprachige Wikipedia die Zuweisung der Kategorie *Kategorie:US-Amerikaner* interpretiert als Unterordnung unter das Konzept DEUTSCHAMERIKANER, weil die Seite *Deutschamerikaner* fälschlich als Hauptartikel der Kategorie *Kategorie:US-Amerikaner* erkannt wurde. Das liegt daran, dass es keinen „richtigen“ Hauptartikel *US-Amerikaner* gibt, und *Deutschamerikaner* die einzige Seite der Kategorie war, die als Hauptartikel infrage kam. Dieser eine Fehler hat, da es viele Personen und darunter viele Amerikaner in der Stichprobe gibt, sichtbare Auswirkungen auf das Ergebnis der Evaluation.

Im allgemeinen liefern die Kategorisierungs-Aliase allerdings sehr gute Ergebnisse: etwa 99% sind korrekt.

Die Subsumtionsrelation wurde hier allerdings nur auf *lokale* Plausibilität untersucht. Interpretiert man sie gemäß ihrer Semantik als transitiv, so können unerwünschte Ergebnisse auftreten: Durch Querverbindungen entstehen unter Umständen sehr lange Pfade, die dann als „abwegige“ Subsumtionen interpretiert werden müssten (siehe dazu auch [HAMMWÖHNER (2007)]).

WikiWord übernimmt die Subsumtionsrelation direkt so, wie sie in der Wikipedia angegeben sind. Ein „Säubern“ dieser Daten, die über das einfache Entfernen von Zyklen (▹E.5) hinausgeht, verbleibt als Gegenstand künftiger Forschung (vergleiche [PONZETTO UND STRUBE (2007B), HAMMWÖHNER (2007B)]). In diesem Rahmen könnte auch untersucht werden, ob sich durch geeignete statistische oder heuristische Verfahren aus der unspezifischen Subsumtionsrelation spezifische semantische Beziehungen extrahieren lassen, ob es also möglich ist, zu erkennen, ob es

sich bei einer Unterordnung um eine *Instanzbeziehung* (*is-a*), *Abstraktionsbeziehung* (*subtype*), eine *Aggregationsbeziehung* (*part-of*) oder eine andere Beziehung handelt (vergleiche [Ponzetto und Strube (2007b)]).

12.3. Auswertung der Ähnlichkeit

Die Zuordnung semantisch ähnlicher Konzepte, wie sie beim Import der Wikipedia-Daten vorgenommen wird (▸9.1.3), basiert auf gemeinsamen Language-Links. Wenn zwei Artikel denselben Artikel in einer anderen Sprache als Äquivalent haben, dann werden sie als „ähnlich“ angesehen und entsprechend im Feld `langmatch` in der Tabelle `relation` markiert. Dieses Vorgehen ist allerdings stark von der Granularität der anderen Wikipedias abhängig: werden bei der Bestimmung der Ähnlichkeit alle Projekte berücksichtigt, dann ist ihre „Trennschärfe“ beschränkt durch die Granularität der *kleinsten*, also am wenigsten granularen Wikipedia. Ein Beispiel: in der deutschsprachigen Wikipedia werden die Konzepte für die Jahre 1980, 1981, 1982 usw. bis 1989 alle als „ähnlich“ angesehen, weil sie alle als Übersetzungsäquivalent als:1980er haben, also das Konzept in der alemannischen Wikipedia, das die achtziger Jahre insgesamt behandelt. Dieses Problem tritt sehr häufig auf, so zum Beispiel auch für Metalle und Eisen, Tag und Sonntag und viele mehr. Solche Paare als „ähnlich“ zu bezeichnen ist nicht völlig falsch, aber kann, je nach Kontext, doch zu weit gefasst sein. Dieses Problem wirkt sich entsprechend auch auf die Ergebnisse der Vereinigung zu einem multilingualen Thesaurus aus.

`langmatch`	*en*	*de*	*fr*
>0	77108	30982	42146
>1	14962	6320	15432
>2	9550	4072	12212
>3	6226	3076	10962
>4	5010	2370	6840
>5	4060	1744	6492
>6	3366	1400	6104

Tabelle `concept_survey`*, siehe* ▸*F.4*

Tabelle 12.4.: Paare von ähnlichen Konzepten

Um dieses Problem zu umgehen, kann verlangt werden, dass zwei Konzepte, um als ähnlich angesehen zu werden, in mehr als einem Language-Link übereinstimmen müssen — die Anzahl der Übereinstimmungen ist in dem Feld `langmatch` in der Tabelle `relation` gespeichert und kann so direkt zum Filtern der Relation verwendet werden. Die Tabelle 12.4 zeigt, wie viele Paare dann jeweils übrig bleiben (wobei interessant ist, zu sehen, dass die französischsprachige Wikipedia deutlich mehr Language-Links hat als die deutschsprachige, obwohl sie etwas kleiner ist). Die stärkste Filterung ist der Schritt von mindestens einer Übereinstimmung zu mindestens zwei: Dabei entfallen viele fehlerhafte Zuordnungen sowie „Ausreißer“, die durch Verweise in sehr kleine Projekte mit geringer Granularität entstehen. Wird ein noch höherer Grad an Übereinstimmung gefordert, verbessert sich die Qualität der Ähnlichkeit sukzessive — so lässt sich der gewünschte *Trade-Off* zwischen Qualität und Abdeckung steuern. Für den Export nach RDF ist dies durch die Option

`--min-langmatch` möglich. Dabei ist allerdings zu bedenken, dass, wenn man mindestens drei übereinstimmende Language-Links verlangt, nur Konzepte überhaupt als ähnlich erkannt werden können, die es noch in mindestens drei weiteren Wikipedias gibt. Wegen der großen Unterschiede in Umfang und Abdeckung der Wikipedias ist das eine recht starke Einschränkung.

Eine Alternative wäre es, Language-Links auf sehr kleine Projekte von der Betrachtung auszuschließen. Vermutlich ließen sich dadurch ebenfalls eine Vielzahl von „unscharfen“ Ähnlichkeiten vermeiden, die entstehen, wenn das Projekt, das Ziel eines Language-Links ist, eine zu geringe Granularität hat. Es wird erwartet, dass dabei weniger Verknüpfungen verloren gingen, als das der Fall ist, wenn man eine Übereinstimmung in mindestens zwei oder mehr Language-Links verlangt. Diese Möglichkeit wurde hier nicht weiter verfolgt und verbleibt als Gegenstand künftiger Forschung.

12.4. Auswertung der Verwandtheit

Die Zuordnung semantisch verwandter Konzepte, wie sie beim Import der Wikipedia-Daten vorgenommen wird (▸9.1.3), ist schwer zu evaluieren, da die Semantik dieser Beziehungen nicht klar definiert ist — das heißt, es ist nicht einfach, zu bestimmen, welche Zuordnung „falsch“ wäre. Die Stärke und Art der subjektiv empfundenen Verwandtheit der Konzeptpaare variiert aber stark. Hier ein paar Beispiele:

Erdmasse ist verwandt mit Planet, Gravitationskonstante und Sonnenmasse.

Earl_Browder (amerikanischer Kommunist) ist verwandt mit 1891 (Geburtsjahr), 1973 (Sterbejahr), Kommunistische_Partei_der_USA (Vorsitz), Präsidentschaftswahl_in_den_Vereinigten_Staaten_1936 (Kandidatur), William_Z._Foster (Vorgänger).

Luke_Donald (englischer Golfer) ist verwandt mit Northwestern_University (Absolvent), World_Cup_(Golf,_Männer) (Sieger), Hemel_Hempstead (Geburtsort), Paul_Casey_(Golfer) (Partner), Omega_Europea_Masters (Sieger).

Verweildauer (im Versicherungsjargon) ist verwandt mit Krankenhaus, Diagnosis_Related_Groups.

Die Qualität der Verknüpfungen über ihre Frequenz zu approximieren, wie das bei der Ähnlichkeit möglich war (▸12.3), führt hier nicht zum Erfolg: die Verwandtheit ergibt sich ja daraus, ob sich zwei Artikel gegenseitig per Wiki-Link referenzieren. Wie oft sie das tun, ist dabei kein Indikator für die Stärke der Verbindung — per Konvention sollte die Anzahl der Verknüpfungen in der Regel sogar immer nur 1 sein: [WP:V] besagt, dass in einem Artikel normalerweise nur einmal auf einen bestimmten anderen Artikel verwiesen werden soll.

Aussichtsreicher ist es, den Typ der Konzepte zu betrachten (▸8.6): Personen zum Beispiel sind oft „verwandt“ mit ihrem Geburts- und Sterbejahr, und eventuell auch mit ihrem Geburtsort — der Grund ist, dass die Artikel über die Jahre Listen von Personen enthalten, die in den betreffenden

Jahren geboren wurden und gestorben sind. Ähnlich wird in Artikeln über Städte häufig angegeben, welche Persönlichkeiten dort geboren wurden oder gewirkt haben. Diese Verknüpfungen sind aber, je nach Nutzung, wenig hilfreich als Indikator für semantische Verwandtschaft. So bietet es sich an, die Verwandtschaftsbeziehung nach Typ zu filtern: zum Beispiel können aus der Verwandtschaft von Personen (Typ `PERSON`) Zeiten entfernt werden (Konzepte vom Typ `TIME`) und eventuell auch Orte (Konzepte vom Typ `PLACE`). Oder man verlangt, dass die Verwandtschaften eines Konzeptes immer vom selben Typ sein müssen, wie das Konzept selbst (wobei `OTHER` auch zulässig sein kann): so findet man zu einer Person weitere Personen, zu einem Ort weitere Orte, zu einem Jahr weitere Jahre und so weiter.

Die Qualität der Verwandtheit ist also schwer festzustellen, der Typ Konzepte bietet einen besseren Anhaltspunkt für die Filterung passend zum Anwendungsfall. Noch besser wäre es natürlich, die konkrete Semantik der Beziehung zwischen den Konzepten zu kennen (etwa *geboren-in* oder *hauptstadt-von*). Um solche Relationen aus Wikipedia-Artikeln zu extrahieren, gibt es im wesentlichen zwei Möglichkeiten: entweder man analysiert die Parameter von bestimmten Vorlagen (*Infoboxen*), um strukturierte Datensätze zu erhalten [WP:IB, Auer u. a. (2007), Auer und Lehmann (2007)]. Oder man analysiert den Satz, in dem die Verknüpfung in den jeweiligen Artikeln auftritt, mit einfachem *Pattern Matching*, oder mit aufwändigeren Methoden der automatischen Sprachverarbeitung [Nguyen u. a. (2007), Nguyen und Ishizuka (2007), Wu und Weld (2007)]. Diese Möglichkeiten der Datenextraktion sind nicht Gegenstand dieser Arbeit und damit nicht Aufgabe von WikiWord — sie sind vielmehr komplementär zu den hier vorgestellten Methoden: ihre Ergebnisse können die von WikiWord ergänzen.

Des Weiteren ist anzumerken, dass die hier betrachtete Verwandtheitsbeziehung relativ grob ist: sie betrachtet nur *explizite direkte* Verwandtheit, die sich vorab für alle Konzepte in einem Thesaurus bestimmen lässt. Sie verbindet also solche Konzepte, die „sehr“ verwandt sind: eine graduelle Abstufung oder ein Vergleich der „Verwandtheit“ ist nicht möglich. In Kapitel ▸14 werden Techniken betrachtet, mit denen ein Wert für die semantische Verwandtheit eines gegebenen Paars von Konzepten oder Termen berechnet werden kann.

12.5. Auswertung der Termzuordnung

Die Bedeutungsrelation, also die Zuordnung von Termen (Bezeichnungen) zu Konzepten (Bedeutungen), ist ein Kernstück eines konzeptorientierten Thesaurus: sie vermittelt zwischen reinem Text und inhaltlichen Beziehungen. Daher ist es besonders wichtig, dass diese Relation korrekt ist, dass heißt, dass sie keine fehlerhaften Zuordnungen liefert. Dieser Abschnitt untersucht die Qualität der Bedeutungsrelation auf der Basis der oben beschriebenen Stichproben.

Die Tabellen 12.5 und 12.6 zeigen die Bewertung der Terme, die den einzelnen Konzepten zugeordnet sind — die Erhebung kann im Einzelnen in Anhang F.4 nachvollzogen werden. Für jeden Term, der einem Konzept zugeordnet ist, wurde eine der folgenden Bewertungen abgegeben, die die Zeilen der Tabellen bilden:

`GOOD`: der Term ist eine korrekte Bezeichnung für das Konzept.

en.wikipedia	$\%t_{all}$	$\%n_{all}$	$\%t_{top}$	$\%n_{top}$
GOOD	87,1	95,9	76,4	91,4
DIRTY	3,0	0,8	2,2	0,3
VARIANT	1,0	0,2	3,8	4,2
DERIVED	1,0	0,4	4,1	1,2
BROADER	3,0	0,6	4,1	1,6
NARROWER	4,0	1,9	6,9	1,0
BAD	1,0	0,2	2,5	0,2

Tabelle concept_survey, *siehe* ▸F.4

Tabelle 12.5.: Bewertung der Termzuordnungen für die englischsprachige Wikipedia

de.wikipedia	$\%t_{all}$	$\%n_{all}$	$\%t_{top}$	$\%n_{top}$
GOOD	87,2	95,8	49,5	83,3
DIRTY	0,0	0,0	0,3	0,0
VARIANT	1,2	0,3	14,8	9,1
DERIVED	9,3	3,3	10,3	3,4
BROADER	2,3	0,7	7,7	1,3
NARROWER	0,0	0,0	15,4	2,5
BAD	0,0	0,0	1,9	0,2

Tabelle concept_survey, *siehe* ▸F.4

Tabelle 12.6.: Bewertung der Termzuordnungen für die deutschsprachige Wikipedia

VARIANT: der Term ist eine Form einer korrekten Bezeichnung für das Konzept, also zum Beispiel ein Plural oder Genitiv, oder das Wort ist groß geschrieben, weil es am Satzanfang steht.

DERIVED: der Term ist von dem Konzept abgeleitet, bezeichnet es aber nicht direkt: zum Beispiel „*Physiker*“ für PHYSIK oder „*deutsch*“ für DEUTSCHLAND.

DIRTY: der Term ist nicht ganz richtig verwendet, zum Beispiel ist die Großschreibung falsch, oder die Verwendung von Bindestrichen oder Apostrophen ist nicht ganz korrekt.

BROADER: der Term bezeichnet eigentlich etwas, das dem Konzept übergeordnet ist — dies tritt relativ häufig auf, und zwar vor allem immer dann, wem ein Term *totem pro parte* (oder in ähnlicher Weise *synekdotisch*) verwendet wird. Beispiele sind „*Amerika*“ für USA oder „*griechisch*“ für GRIECHISCHE MYTOLOGIE. Zum Teil handelt es sich dabei um ungenaue, aber umgangssprachlich gebräuchliche Bezeichnungen, zum Teil aber auch um Terme, die nur in einem ganz bestimmten Kontext auf das gegebene Konzept verweisen: „*Vergrößerung*“ kann zum Beispiel für HEPATOMEGALIE stehen, wenn der Kontext sich mit Krankeiten der Leber beschäftigt. Diese Bedeutungszuweisung ist dann zwar stark kontextabhängig, aber dennoch valid.

NARROWER: der Term bezeichnet etwas, das dem Konzept untergeordnet ist — dies tritt vor allem dann auf, wenn es für das konkretere Konzept keinen eigenen Artikel in der Wikipedia gibt. Sehr häufig zum Beispiel verweist der Name eines Stadteils auf den Artikel über die

ganze Stadt. Dieses Vorgehen ist zwar im Rahmen der Wikipedia pragmatisch, führt aber für WikiWord zu fehlerhaften Bedeutungszuweisungen — besser ist es, wenn in einem solchen Fall auf einen *Abschnitt* des entsprechenden Artikels verwiesen wird. Dann erzeugt WikiWord für den Abschnitt ein eigenes Konzept (▷9.1.1).

Zu beachten ist auch, dass Terme in der Wikipedia fast nie aus rethorischen Gründen auf ein allgemeineres Konzept verweisen — das heißt, sie werden nicht *pars pro toto* verwendet, bzw. wird, wenn diese Stilfigur verwendet wird, kein Wiki-Link gesetzt.

BAD: der Term bezeichnet das Konzept nicht, die Verwendung als Link-Text ist falsch oder zumindest irreführend. Zum Teil handelt es sich hier um Tippfehler oder eine fehlerhafte Wahl des Link-Zieles, zum Teil aber auch um „navigatorische" Link-Texte, wie „*das gleichnamige Buch*" oder „*im folgenden Jahr*". Diese sind für die Bedeutungsrelation unbrauchbar.

Die Übergänge zwischen den einzelnen Bewertungen sind bisweilen fließend. Einige Beispiele:

- Die Bezeichnung „*Amerika*" für das Konzept USA kann als Kurzform des Namens gesehen werden (`GOOD`) oder als Synekdoche, also eine Verwendung *totem pro parte* (`BROADER`).
- Das Adjektiv „*physikalisch*" für das Konzept Physik kann als Wortform (`VARIANT`) oder als abgeleitetes Wort (`DERIVED`) betrachtet werden.
- Die Schreibweise „*Lodz*" für Łódź kann als valide deutsche Transkription (`GOOD`) oder als typographischer Fehler (`DIRTY`) angesehen werden.

Aus diesem Grund wird die Unterscheidung im Folgenden auf „gute" und „schlechte" Zuordnungen reduziert, wobei nur die Bewertungen `NARROW` und `BAD` als „schlecht" betrachtet werden:

Die Interpretation von `BROADER` als „gut" und `NARROWER` als „schlecht" ist dadurch begründet, dass, wie oben schon bemerkt, die Bewertung `BROADER` häufig aus einer Verwendung des Terms als Synekdoche stammt, während die Bewertung `NARROW` meist von „Verlegenheitsverknüpfungen" herrührt, die entstehen, wenn kein Artikel zu dem konkreten Konzept existiert, und deshalb ein Wiki-Link auf ein übergeordnetes Thema gesetzt wird. Aus Sicht von WikiWord wäre es in solchen Fällen vorzuziehen, entweder einen Link auf eine noch nicht existierende Seite zu setzen (was unter Umständen durchaus erwünscht ist [WP:RL, WP:V]) oder auf einen konkreten Abschnitt in einem übergeordneten Artikel zu verweisen. Aus diesem Grund wird hier `BROADER` noch als „gut", aber `NARROW` als „schlecht" bewertet.

Die Bewertungen `VARIANT` und `DERIVED` sind aus der Perspektive des *Information Retrieval* sicherlich als valid anzusehen, da sich im Kontext einer Dokumentensuche oder Indexierung der entsprechenden Text korrekt dem fraglichen Konzept zuordnen lässt. Dasselbe gilt auch für die Bewertung `DIRTY`, da für die Effektivität der Suche ja nicht entscheidend ist, ob eine Bezeichnung „korrekt" ist, sondern nur, wie sie pragmatisch verwendet wird.

Die Spalten der Tabellen 12.5 und 12.6 teilen sich in zwei Gruppen: die ersten beiden Spalten geben die Werte für die Stichprobe über den Gesamtkorpus an ($_{all}$), die letzten beiden Spalten

die Werte für die Stichprobe aus den 10% meistgenutzer Konzepte ($_{top}$). Die jeweils erste Spalte gibt direkt den Anteil von Termen wieder, auf die die der Zeile entsprechende Bewertung zutrifft (*%t*), die jeweils zweite Spalte ist nach Termfrequenz gewichtet, das heißt, sie gibt den Anteil der *Verwendungen* von Termen an, auf die die Bewertung zutrifft (*%n*).

Vergleicht man die Bewertungen innerhalb einer Spalte, so ist zunächst offensichtlich, dass die allermeisten Terme als `GOOD` markiert sind, also als völlig korrekte Bezeichungen für das Konzept. Der Anteil liegt bei etwa 50%–87% der Terme, aber etwa 83%–97% der Verwendungen — was besagt, dass die „guten“, primären Bezeichnungen für ein Konzept häufiger verwendet werden als irgendwelche Varianten, abgeleitete oder gar falsche Bezeichnungen. Der Anteil der „guten“ Bezeichnungen ist ausserdem bei der Stichprobe aus den meistgenutzten 10% kleiner als bei der Stichprobe über den Gesamtkorpus — der Grund dafür ist vermutlich, dass abgeleitete Formen oder alternative Bezeichnungen bevorzugt für gebräuchliche Konzepte verwendet werden, während ungebräuchliche Konzepte meist über ihren primären Namen referenziert werden.

Bei einem Vergleich zwischen den Ergebnissen für die englischsprachige und die deutschsprachige Wikipedia fällt auf, dass in der deutschsprachigen Wikipedia deutlich mehr Terme als `VARIANT` (Wortform) oder `DERIVED` (Abgeleitet) markiert sind. Das ist vermutlich durch die reichere Morphologie der deutschen Sprache sowie die Verwendung von Komposita zu erklären. Diese Vermutung durch Untersuchungen an weiteren Sprachen zu erhärten verbleibt als Gegenstand künftiger Forschung. Eine weitere Auffälligkeit ist, dass in der englischsprachigen Wikipedia mehr Einträge als `DIRTY` markiert sind — das ist insbesondere auf fehlende Apostrophe bei Genitiv-Formen sowie auf fälschliche Kleinschreibung bei Namen und Abkürzungen zurückzuführen.

	LINK	DISAMBIG	SORTKEY	REDIRECT	TITLE
GOOD	96,5	76,5	100,0	57,7	99,0
VARIANT	0,2	–	–	3,8	–
DERIVED	0,8	–	–	11,5	–
DIRTY	0,2	–	–	15,4	–
BROADER	1,1	11,8	–	–	1,0
NARROWER	1,1	–	–	11,5	–
BAD	0,2	11,8	–	–	–

`eval-rules-en.tsv`, siehe ▹F.5

Tabelle 12.7.: Evaluation der Termzuordnung, nach Extraktionsregel (englischsprachige Wikipedia, all-Stichprobe)

Um nun den Anteil der „schlechten“ Termzuordnungen zu reduzieren, können die Zuordnungen gefiltert werden (*Cutoff*, siehe ▹10.4). Dabei kommen die beiden folgenden Kriterien in Betracht: die Häufigkeit, mit der die Zuordnung beobachtet wurde (vergleiche Tabelle 12.8) sowie die Regeln, über die die Zuordnung extrahiert wurde (vergleiche Tabelle 12.7), wie sie in den Feldern `freq` bzw. `rule` in der Tabelle `meaning` (▹C.2) gespeichert sind. Die Bedeutungsrelation wird dabei also um Zuordnungen beschnitten, die zu *selten* und/oder zu *schwach* sind. Die Erwartung

	>=1	>=2	>=3	>=4
GOOD	84,8	94,5	94,8	97,5
VARIANT	1,1	0,0	0,0	0,0
DERIVED	2,7	1,8	1,7	0,0
DIRTY	2,2	0,9	0,0	0,0
BROADER	5,4	0,0	0,0	0,0
NARROWER	2,2	2,7	3,4	2,5
BAD	1,6	0,0	0,0	0,0

eval-freq-en.tsv, siehe ▸F.5

Tabelle 12.8.: Evaluation der Termzuordnung, nach Frequenz (englischsprachige Wikipedia, all-Stichprobe)

	$\%tc_{all}$	$\%t_{all}$	$\%nc_{all}$	$\%n_{all}$	$\%tc_{top}$	$\%t_{top}$	$\%nc_{top}$	$\%n_{top}$
Alle	100,0	91,3	100,0	96,7	100,0	84,3	100,0	97,3
Min. 2×od. Exp.	82,6	91,4	96,0	97,0	79,9	86,6	98,2	97,7
Min. 3×od. Exp.	80,4	91,2	94,9	96,9	75,2	87,9	97,3	97,9
Min. 2×	59,8	94,5	90,7	97,8	50,3	90,0	95,5	98,1
Min. 3×	31,5	94,8	77,5	98,4	38,7	94,3	93,4	98,6
Nur Explizite	77,2	91,5	87,4	97,1	65,7	88,5	77,8	98,7

eval-cutoff-en.tsv, siehe ▸F.5

Tabelle 12.9.: Auswirkung verschiedener Cutoff-Modi, für die englischsprachige Wikipedia

	$\%tc_{all}$	$\%t_{all}$	$\%nc_{all}$	$\%n_{all}$	$\%tc_{top}$	$\%t_{top}$	$\%nc_{top}$	$\%n_{top}$
Alle	100,0	94,4	100,0	98,1	100,0	72,0	100,0	93,8
Min. 2×od. Exp.:	81,4	96,5	94,7	98,8	61,4	80,1	96,0	95,2
Min. 3×od. Exp.	78,5	97,8	93,1	99,5	54,3	80,5	94,6	95,5
Min. 2×	59,3	98,1	88,4	99,3	48,2	80,7	94,7	95,5
Min. 3×	26,0	100,0	69,3	100,0	35,4	86,4	92,1	96,3
Nur Explizite	76,3	97,8	90,1	99,5	46,3	79,2	89,2	95,7

eval-cutoff-de.tsv, siehe ▸F.5

Tabelle 12.10.: Auswirkung verschiedener Cutoff-Modi, für die deutschsprachige Wikipedia

ist, dass sich dadurch der Anteil der „guten“ Termzuordnung (also die *Precision*) vergößern lässt, wobei in Kauf genommen wird, dass sich die Abdeckung (*Coverage*) verschlechtert[1].

Die Tabellen 12.9 und 12.10 zeigen die Auswirkungen unterschiedlicher Cutoff-Methoden auf die Stichproben aus dem englischsprachigen bzw. deutschsprachigen Thesaurus. Sie geben die Abdeckung und den Anteil als „gut“ bewerteter Termzuordnungen wieder, jeweils für einen bestimmten Cutoff-Modus und eine bestimmte Stichprobe. Die ersten vier Spalten beziehen sich auf die Stichprobe über alle Konzepte ($_{all}$), die letzten vier auf die Stichprobe aus den oberen 10% der am häufigsten verwendeten Konzepte ($_{top}$). Für beide Stichproben beziehen sich die ersten beiden Spalten jeweils auf den Anteil an unterschiedlichen Termen (t) und die beiden verbleibenden auf den Anteil bezüglich aller Verwendungen der Terme, also skaliert nach Termfrequenz (n). So ergeben sich vier Paare von Spalten, in jedem dieser Paare gibt die erste Spalte (tc bzw. nc) die Abdeckung an, also den Anteil von Termzuweisungen, die nach dem Cutoff verblieben sind, und die zweite Spalte (t bzw. n) gibt den Anteil der als „gut“ bewerteten Terme an (also alle mit den Bewertungen `GOOD`, `DIRTY`, `DERIVED`, `VARIANT` und `BROADER`).

Die Spalten der Tabelle geben die Ergebnisse für unterschiedliche Cutoff-Modi an, grob sortiert nach zunehmender „Strenge“ der Kriterien. Im Einzelnen:

Alle: Werte ohne Cutoff, also für alle gefundenen Termzuordnungen.

Min. 2×od. Exp.: Die Zuordnung muss mindestens zweimal vorkommen, oder sie muss mindestens einmal explizit vorgenommen worden sein, also nicht ausschließlich aus einer Verwendung des Termes als Link-Text folgen.

Min. 3×od. Exp.: Die Zuordnung muss mindestens dreimal vorkommen, oder sie muss explizit vorgenommen worden sein.

Min. 2×: Die Zuordnung muss mindestens zweimal vorkommen, egal ob sie explizit vorgenommen wurde oder nicht.

Min. 3×: Die Zuordnung muss mindestens dreimal vorkommen.

Nur Explizite: Die Zuordnung muss mindestens einmal explizit vorgenommen worden sein, also nicht ausschließlich aus einer Verwendung des Termes als Link-Text folgen.

Dabei ist es im Allgemeinen so, dass sowohl die Vorgabe einer Mindestfrequenz als auch das Verlangen einer expliziten Zuordnung die Abdeckung stark reduziert, um die Präzision um einige Prozentpunkte zu erhöhen. Dabei sind die Auswirkungen auf die skalierten Werte, also die Anteile an der Verwendung der Terme, deutlich geringer als an den absoluten Werten. Das ist konsistent mit der vorher beschriebenen Beobachtung, dass „gute“ Terme für ein Konzept weit häufiger verwendet werden als „schlechte“; auch ist der Zusammenhang natürlich darin begründet, dass es ja gerade die wenig verwendeten Termzuordnungen sind, die ausgefiltert werden.

[1] Die Abdeckung bezieht sich auf die Anzahl von Zuordnungen bzw. Termen ohne Cutoff; der *Recall* würde sich auf die Anzahl *aller* Bezeichnungen für die gegebenen Konzepte beziehen, diese ist jedoch unbekannt, und selbst mit Hilfe eines „Gold-Standards“ auf der Basis von Standard-Wörterbüchern nur schlecht anzunähern.

Wird kein Cutoff angewendet, ergibt sich (bei, per Definition, 100% Abdeckung) die schlechteste Präzision, die aber für den Anteil der Termverwendungen immer noch durchweg bei 97% liegt. Bezogen auf unterschiedliche Terme liegt die Präzision immer noch zwischen 84% und 94% (mit der Ausnahme von 72% für die Top10%-Stichprobe aus der deutschsprachigen Wikipedia, die vor allem auf das oben erwähnten Problem mit Stadtteilen zurückzuführen ist). Das zeigt, dass die Bedeutungsrelation von WikiWord schon recht hohe Qualität hat. Etwas überraschend ist, dass eine Beschränkung auf explizite Termzuweisungen, die mit großen Einbußen in der Abdeckung verbunden ist, in keinem Fall besser ist als alle anderen Möglichkeiten des Cutoffs, und zumeist sogar deutlich hinter anderen Möglichkeiten zurückbleibt. Das ist insbesondere auch deshalb interessant, weil dieser Modus den Ergebnissen entspricht, die zu erwarten wären, wenn der Link-Text gar nicht als Quelle von Termzuordnungen berücksichtigt würde — wie es die meisten anderen Ansätze zur Extraktion eines Thesaurus aus der Wikipedia tun. Die Verwendung des Link-Textes bringt also nicht nur einen quantitativen, sondern im Allgemeinen auch einen qualitativen Vorteil — zum Teil sogar ohne Filterung, spätestens aber nach Anwendung geeigneter Cutoff-Regeln. Das bestätigt die Ergebnisse von [Craswell u. a. (2001), Eiron und Mccurley (2003), Kraft und Zien (2004)].

Die Anwendung eines „naiven“ Cutoffs, der eine minimale Anzahl von Beobachtungen einer Termzuordnung verlangt, liefert eine Verbesserung der Präzision um ein bis zwei Prozentpunkte auf über 99% (in einer Stichprobe sogar auf 100% für eine Mindestfrequenz von 3), allerdings auf Kosten der Abdeckung, insbesondere in Bezug auf die Anzahl der bekannten unterschiedlichen Terme: sie sinkt für eine Mindestfrequenz von 2 auf 50% und für eine Mindestfrequenz von 3 auf circa 30% — das entspricht grob der Erwartung, dass, nach dem Zipfschen Gesetz [Zipf (1935)], die Häfte der Zuordnungen nur einmal vorkommt, und drei Viertel nur höchstens zweimal.

Die Kombination der Kriterien *Mindestfrequenz* und *explizite Zuordnung* ist als Kompromiss zwischen Abdeckung und Präzision gedacht: wird eine explizite Zuweisung nur für Terme verlangt, die selten vorkommen, so ergibt sich ein deutlich geringerer Verlust an Abdeckung, während der Gewinn an Präzision fast gleich bleibt: Die Verschlechterung der Präzision beträgt etwa 0,5 Prozentpunkt bezüglich der Termverwendungen und bis zu 5 Prozentpunkte bezüglich unterschiedlicher Terme. Dem steht eine Verbesserung der Abeckung um circa 5 Prozentpunkte bezüglich der Termverwendungen und bis zu 50 Prozentpunkte bezüglich der Anzahl unterschiedlicher Terme gegenüber.

Über die beiden Cutoff-Kriterien lässt sich der Trade-Off zwischen Abdeckung und Präzision steuern, insbesondere auch in Bezug darauf, ob in der gegebenen Anwendung die Präzision in Bezug auf die zu erwartende natürliche Häufigkeit der Verwendung relevant ist, oder ob die Wahrscheinlichkeit der Nutzung jedes Terms in etwa gleich ist: ist die Präzision wichtig und die Terme treten in ihrer natürlichen (Zipfschen) Verteilung auf, so empfielt sich ein „strikter“ Cutoff. Ist jedoch die Abdeckung wichtig, sollte ein kombinierter Cutoff verwendet werden, insbesondere dann, wenn die Terme gleichverteilt auftreten, da sonst die Abdeckung zu stark reduziert würde.

Es wäre interessant, weiter zu untersuchen, wie sich unterschiedliche Definitionen für explizite Zuweisung auswirken, also welche Extraktionsregeln unter den Cutoff fallen. In obigem Experiment wurde die Regel `TERM_FROM_LINK` als implizit angesehen, die anderen als explizit. Aufgrund der hohen Fehlerraten der Regeln `TERM_FROM_DISAMBIG` sowie `TERM_FROM_REDIRECT` (vergleiche Tabelle 12.7) scheint es sinnvoll, verschiedene Kombinationen von Extraktionsregeln für den Cutoff näher zu betrachten.

12.6. Auswertung der Glossen

Die Extraktion von Definitionssätzen (*Glossen*, ▹8.8) aus den Artikeln wurde in Bezug darauf evaluiert, ob sie in der Tat einen Definitionssatz liefert. Dabei wurde nicht die *faktische Richtigkeit* bewertet (die der Autor in vielen Fällen gar nicht beurteilen kann), sondern lediglich, ob es sich um einen wohlgeformten, vollständigen Definitionssatz handelt, der eine nicht-triviale Aussage über das Konzept macht. Im einzelnen waren die folgenden Bewertungen möglich:

UNKNOWN: Zu dem betreffenden Konzept gibt es keinen Artikel, daher konnte keine Definition extrahiert werden, die Bewertung ist also gegenstandslos.

OK: Der Definitionssatz wurde korrekt extrahiert und macht eine nicht-triviale Aussage über das Konzept.

MISSING: Es wurde kein Definitionssatz extrahiert. Das bedeutet entweder, dass der Artikel nicht der Standard-Form für Wikipedia-Artikel folgt [WP:ARTIKEL, WP:WSIGA] und nicht mit einem Definitionssatz beginnt, oder, dass die Muster und Heuristiken zum Auffinden des ersten Satzes fehlgeschlagen sind.

TRIVIAL: Der Definitionssatz wurde korrekt extrahiert, enthält aber keine sinnvolle Aussage. Relativ häufig kommen Sätze wie der folgende vor: „*Weltoffenheit ist ein Begriff aus der philosophischen Anthropologie.*“ (▹8.8).

TRUNCATED: Der Definitionssatz wurde nur teilweise extrahiert, ist abgeschnitten oder sonstwie verstümmelt. Der Grund ist meist eine fehlerhafte Erkennung des Satzendes (etwa fälschlich nach einer unbekannten Abkürzung), Verwirrung durch verschachtelte Strukturen wie Klammersätze oder wörtliche Rede, oder aber die Verwendung von unbekannten Vorlagen für die Formatierung von Text.

RUNAWAY: Der Definitionssatz wurde korrekt extrahiert, aber es wurde noch weiterer Text mitgeliefert. Meistens wird noch ein zweiter Satz angehängt, weil der erste mit einer Abkürzung wie „*etc.*“ endete und daher das Satzende nicht als solches erkannt wurde.

CLUTTERED: Der Definitionssatz wurde korrekt extrahiert, enthält aber Überbleibsel von Wiki-Markup oder ähnlichem. Das ist meist auf eine fehlerhafte Verwendung von Wiki-Markup zurückzuführen, oder aber auf Fehler in den Mustern, die zur Bereinigung des Textes verwendet werden (▹8.3).

WRONG: Es wurde etwas extrahiert, was kein Definitionssatz für das betreffende Konzept ist. Entweder der erste Satz ist kein Definitionssatz, weil der Artikel nicht der Standard-Form für Wikipedia-Artikel folgt ([WP:WSIGA, WP:ARTIKEL]), oder die Muster und Heuristiken zum Auffinden des ersten Satzes sind fehlgeschlagen.

Die Tabelle 12.11 zeigt, dass die Definitionssätze in den allermeisten Fällen korrekt extrahiert werden (`OK`) oder nur kleinere Probleme aufweisen (`RUNAWAY`, `CLUTTERED`).

(a) gesamt

Typ	en_{all}	en_{top}	de_{all}	de_{top}
UNKNOWN	58%	2%	72%	2%
OK	40%	86%	28%	88%
CLUTTERED	–	2%	–	–
RUNAWAY	–	2%	–	2%
TRUNCATED	2%	4%	–	4%
TRIVIAL	–	–	–	4%
MISSING	–	2%	–	–
WRONG	–	2%	–	–

(b) ohne UNKNOWN

Typ	en_{all}	en_{top}	de_{all}	de_{top}
UNKNOWN				
OK	95%	88%	100%	90%
CLUTTERED	–	2%	–	–
RUNAWAY	–	2%	–	2%
TRUNCATED	5%	4%	–	4%
TRIVIAL	–	–	–	4%
MISSING	–	2%	–	–
WRONG	–	2%	–	–

Tabelle `concept_survey`, *siehe* ▹F.4

Tabelle 12.11.: Evaluation der Glossen

Die Fälle, in denen der Satz abgeschnitten wurde (`TRUNCATED`) sind meist auf eine zu strenge Heuristik zurückzuführen, nämlich die, dass ein einzelner Zeilenumbruch einen Satz beendet — das ist gemäß der Wiki-Syntax nicht notwendig (▹A) und kann in `LanguageConfiguration.sentencePattern` angepasst werden. Das würde aber vermutlich zu einer Vermehrung der `RUNAWAY`-Fälle führen.

Gegen triviale Glossen (`TRIVIAL`), die der Form nach, aber nicht dem Inhalt nach eine Definition darstellen, ist mit algorithmischen Mitteln wenig zu machen. Es wäre höchstens möglich, immer den ganzen ersten Absatz als Glosse zu nehmen, das würde aber in den meisten Fällen Text mit einbeziehen, der nicht zur Definition gehört, und laut Anforderung (▹5.1) hier nicht erwünscht ist (vergleiche ▹8.8).

Wenn gar keine Glosse gefunden wird (`MISSING`), liegt das meist daran, dass auch keine vorhanden ist — in vielen Wikis ist das zum Beispiel für Jahresartikel der Fall, die im Wesentlichen als Listen von Ereignissen strukturiert sind. Es fällt auch schwer, sich für eine Jahreszahl eine sinnvolle Definition vorzustellen.

Wird ein falscher Satz als Glosse gefunden (`WRONG`), so liegt das oft daran, dass vor dem Definitionssatz eine Bemerkung steht — einleitende Hinweise (*Tags*) in Artikeln kommen recht oft vor, zum Beispiel also Warnung vor schlechter Qualität des Artikels, oder als Hinweis auf eine Begriffsklärungsseite. Aber normalerweise sind solche *tags* Vorlagen, oder sie sind zumindest mit `<center>` oder `<div>` formatiert und können damit leicht erkannt und herausgefiltert werden. In einigen Fällen jedoch ist das nicht der Fall: in der englischsprachigen Wikipedia zum Beispiel stehen Verweise auf alternative Bedeutungen häufig direkt vor dem eigentlichen Artikelinhalt, und sind nur dadurch abgesetzt, dass sie eingerückt sind. Wird diese Einrückung vergessen, was durchaus vorkommt, so findet der Algorithmus diesen vorangestellten Kommentar als ersten Satz, der dann fälschlich als Glosse interpretiert wird. Es ist zu hoffen, dass auch die englischsprachige Wikipedia zur Nutzung von Vorlagen für diesen Zweck übergeht, so dass dieses Problem in Zukunft vermieden wird.

13. Analyse von Begriffsklärungsseiten

Datensatz	OK	NOT	MISSED	PART	BAD	Precision	Recall	F-Maß
en	282	100	16	8	5	0,96	0,95	0,95
de	202	59	30	16	5	0,91	0,87	0,89

disambig-eval.tsv, siehe ▸F.5, Tabelle resource_survey*, siehe ▸F.4*

Tabelle 13.1.: Evaluation der Analyse von Begriffsklärungsseiten

Der in ▸8.10 angegebene Algorithmus zur Analyse von Begriffsklärungsseiten wurde auf einer Stichprobe von jeweils 50 Begriffsklärungsseiten aus zwei Wikipedias (en und de) evaluiert, das Ergebnis ist in Tabelle 13.1 aufgeführt. Ziel ist es, auf der Begriffsklärungsseite diejenigen Wiki-Links zu finden, die auf Bedeutungen der fraglichen Bezeichnung verweisen. Für die Evaluation wurden nun alle Wiki-Links auf jeder Seite der Stichproben manuell dahingehend bewertet, ob sie auf eine der Bedeutungen des Wortes zeigen, für das die Begriffsklärungsseite angelegt wurde (siehe Anhang F.4). Das Ergebnis der manuellen Beurteilung wird dann mit der automatischen Einordnung verglichen:

OK: der Link verweist auf eine der Bedeutungen des Wortes, für das die Begriffsklärungsseite angelegt wurde und er wurde auch als Bedeutungs-Link erkannt (*True Positive*).

NOT: der Link verweist nicht auf eine der möglichen Bedeutungen und wurde auch nicht als Bedeutungs-Link identifiziert. (*True Negative*)

MISSED: der Link verweist auf eine der Bedeutungen, wurde aber nicht als Bedeutungs-Link erkannt (*False Negative*).

BAD: der Link verweist nicht auf eine der Bedeutungen, wurde aber als Bedeutungs-Link erkannt (*False Positive*).

PART: der Link verweist auf etwas, das der eigentlichen Bedeutung übergeordnet ist, wurde aber als Bedeutungs-Link erkannt (auch *False Positive*).

Diese Evaluation kann allerdings aufgrund der geringen Stichprobengröße nur einen ungefähren Eindruck von der Leistungsfähigkeit des verwendeten Algorithmus bieten. Bei der Fehleranalyse war erneut ein Problem zu beobachten, das schon bei den Termzuordnungen über den Link-Text aufgetreten ist: es kommt (insbesondere bei Ortsteilen) häufig vor, dass ein Link auf einen übergeordneten Artikel gesetzt wird, wenn es keinen eigenen Artikel für das Konzept gibt (Bewertung PART in Tabelle 13.1; vergleiche auch Tabelle die Spalte DISAMBIG in Tabelle 12.7). So enthält die Begriffsklärungsseite für *Hüttenberg* zum Beispiel die Zeile „* *einen Ortsteil von [[Brobergen]] im Landkreis Stade in Niedersachsen*“, wodurch Brobergen als Bedeutung von „*Hüttenberg*“ erkannt wird; wie man dem Text entnehmen kann, ist das aber falsch: „*Hüttenberg*“ ist nur eine Bezeichnung für einen *Teil von* Brobergen. Für WikiWord wäre es wünschenswert, wenn an solchen Stellen in der Begriffsklärung ein Wiki-Link auf einen noch nicht

existierenden Artikel über den Ortsteil stünde — allerdings ist das aus der pragmatischen Sicht der Wikipedia nur dann wünschenswert, wenn der Ortsteil hinreichend „interessant“ ist, dass sich ein eigener Artikel darüber „lohnt“. Eine Alternative ist es, auf einen Abschnitt in dem übergeordneten Artikel zu verweisen, wenn es diesen gibt, in diesem Fall etwa BROBERGEN#HÜTTENBERG.

14. Verwandtheit und Disambiguierung

Zwei wichtige Aufgaben im Kontext der Automatischen Sprachverarbeitung und des Information Retrieval sind die *Disambiguierung*, also die *Bedeutungszuweisung* für Wörter abhängig von dem Kontext in dem sie auftreten, sowie die Bestimmung der *semantischen Verwandtheit* von Konzepten. Diese beiden Aufgaben sind an sich sehr unterschiedlich, sie sind jedoch in der Praxis dadurch verknüpft, dass häufig die Verwandtheit von *Termen* bestimmt werden soll; diese Aufgabe kann dadurch gelöst werden, dass den Termen zunächst eine *Bedeutung* (also ein *Konzept*) zugewiesen und dann die Verwandtheit zwischen den Konzepten bestimmt wird. Andersherum ist die Verwandtheit von Konzepten eine nützliche Information bei der Aufgabe der Disambiguierung.

Dieses Kapitel beschreibt, wie die von WikiWord gesammelten Daten zur Bestimmung der Verwandtheit von Termen verwendet werden können. Das Ziel ist, zu zeigen, dass sich die von Wikiword gesammelten Daten grundsätzlich zur Bestimmung von semantischer Verwandtheit von Termen und Konzepten sowie zur automatischen Disambiguierung (Bedeutungszuweisung) eignen. Es soll hier nicht versucht werden, Verfahren für die Bestimmung der Verwandtheit und zur Disambiguierung zu optimieren oder auf ihre Tauglichkeit zu testen — solche Verfahren werden nur insofern untersucht, als sie Aussagen über Eigenschaften der von WikiWord beschafften Daten ermöglichen. Für einen Vergleich verschiedener Methoden zur Bestimmung der semantischen Verwandheit, siehe [Strube und Ponzetto (2006), Gabrilovich und Markovitch (2007), Zesch u. a. (2007b)].

Zunächst werden hierfür Verfahren beschrieben, mit denen die Verwandtheit von Konzepten bestimmt und Termen im Kontext ein Sinn zugewiesen werden kann. Diese Verfahren werden dann verwendet, um die Eignung der von WikiWord gewonnenen Daten zur Lösung der oben genannten Aufgaben zu evaluieren. Die Verfahren selbst sind dabei möglichst einfach gehalten und es wird erwartet, dass durch die Verwendung anspruchsvollerer Verfahren zur Disambiguierung und Bestimmung der Verwandtheit die Ergebnisse noch zu verbessern wären.

14.1. Semantische Verwandtheit zwischen Konzepten

Die semantische Verwandtheit von Konzepten kann auf unterschiedliche Weise bestimmt werden, wobei sich die Ansätze grob in drei Kategorien gliedern lassen:

1. Die Verwandtheit wird aufgrund des *Abstandes* der Konzepte in einem *Graphen* bestimmt, zum Beispiel in einem Hyperlink-Netzwerk oder einer Taxonomie.
2. Die Verwandtheit wird aufgrund der *Ähnlichkeit* einer textuellen *Beschreibung* bestimmt, zum Beispiel auf Basis von Glossen. Die Beschreibungen werden dabei in der Regel als *Featurevektor* repräsentiert und die Verwandtheit als Abstand oder Winkel in einem *Merkmalsraum* angesehen (das entspricht dem *Vector Space Model*, [Salton u. a. (1975)]).
3. Die Verwandtheit wird aufgrund der *Gemeinsamkeiten* in ihrer *Verwendung* bestimmt. Dieser Ansatz kann weiter unterteilt werden in stochastische bzw. informationstheoretische Verfahren, die die Verwandtheit zum Beispiel auf der *Mutual Information* begründen

[Nakayama u. a. (2007b), Ponzetto und Strube (2007b)], und solche, die den Kontext der Verwendung als *Featurevektor* repräsentieren und die Verwandtheit als Abstand oder Winkel in einem *Merkmalsraum* ansehen.

Das WikiWord-Modell beinhaltet bereits eine Beziehung von „Verwandtheit“, die vorab ermittelt wurde und direkt im Modell gespeichert ist (▹C.2, ▹9.1.3, ▹12.4). Diese ist für die Zwecke dieser Evaluation nicht ausreichend, schon weil sie viel zu wenige Paare von Konzepten abdeckt (nämlich nur die explizit verknüpften). Deshalb wird diese *explizite* Verwandtheit nur als einer von mehreren Faktoren bei der Berechnung der *effektiven* Verwandtheit verwendet.

Für die Evaluation von WikiWord wurde ein einfaches Verfahren gewählt, das sich auf Basis der vorliegenden Daten leicht implementieren und effizient anwenden lässt. Die semantische Verwandtheit zweier Konzepte wird als der Wert einer Ähnlichkeitsfunktion definiert, die auf zwei Featurevektoren angewendet wird, die die beiden Konzepte repräsentieren:

$$r_{sem}(c_1, c_2) = sim\,(v(c_1), v(c_2))$$

Als Ähnlichkeitsmaß wird das Kosinus-Maß verwendet, also das normalisierte Skalarprodukt:

$$sim_{cos}(v_1, v_2) = \frac{v_1 \circ v_2}{|v_1| \cdot |v_2|}$$

Zum Vergleich wird zum Teil eine gewichtete Tanimoto-Ähnlichkeit herangezogen. Hierbei werden die Vektoren als *Fuzzy Sets* interpretiert und entsprechend die Größe der Schnittmenge bzw. der Vereinigung als die Summe das paarweise Minimum bzw. Maximum der einzelnen Elemente der Vektoren bestimmt.

$$sim_{tan}(v_1, v_2) = \frac{\sum_i min(v_{1,i}, v_{2,i})}{\sum_i max(v_{1,i}, v_{2,i})}$$

Der Merkmalsraum für die Featurevektoren wird durch die Konzepte des Thesaurus aufgespannt: jedes Konzept definiert eine Dimension. Ein Featurevektor repräsentiert einen Ort in diesem Raum, das heißt, er besteht aus einer Zuordnung von Gewichten zu jedem Konzept. Die Beziehungen, die ein bestimmtes Konzept bezüglich einer bestimmten Relation des Thesaurus hat, weisen ihm einen Ort im Merkmalsraum zu, der durch den Featurevektor dieses Konzeptes bezüglich dieser Relation definiert ist:

$$v_R(c_k) = \begin{pmatrix} w_R(c_k, c_1) \\ w_R(c_k, c_2) \\ \vdots \\ w_R(c_k, c_n) \end{pmatrix}$$

Wobei w_R eine Funktion ist, die das Gewicht der Verknüpfung zweier Konzepte bezüglich der Relation R angibt; das heißt, sie gibt die Gewichte der Kanten in dem von R aufgespannten

Graphen an. In dem hier beschriebenen Experiment werden zwei Varianten dieser Funktion verwendet: für die *ungewichtete* Verwandtheit wird für w_R die charakteristische Funktion von R verwendet, die 1 zurückgibt, wenn $c_n R c_m$ gilt; für die *gewichtete* Verwandtheit ergibt sich w_R aus dem *Relevanzwert* der Verknüpfung, wie er im Modell gespeichert ist. In der Implementation wird er dem Referenzobjekt entnommen, das die entsprechende Beziehung repräsentiert ▹D.2). In der vorliegenden Implementation handelt es sich dabei meist um den IDF-Wert des Konzeptes, das Ziel der Verknüpfung ist (oder, im Fall der Subsumtionsbeziehung, um den inversen LHS-Wert, ▹9.3). Theoretisch könnte die Relevanz aber auch direkt als Kantengewicht modelliert sein.

Der Featurevektor eines Konzeptes ergibt sich dann aus der Summe der Featurevektoren bezüglich der einzelnen Relationen, jeweils skaliert mit einem (*ad hoc* festgelegten) Koeffizienten, der angibt, wie viel Gewicht eine bestimmte Relation in der Berechnung der Verwandtheit haben soll:

$$v(c) = a_{id}v_{id}(c) + a_{br}v_{br}(c) + a_{nr}v_{nr}(c) + a_{in}v_{in}(c) + a_{out}v_{out}(c) + a_{rel}v_{rel}(c) + a_{sim}v_{sim}(c)$$

Die berücksichtigten Relationen aus dem Konzeptmodell (▹4.2, ▹C.1) sind die folgenden:

v_{id} ist der *Identitätsvektor* des Konzeptes, also der Vektor, in dem nur für die Dimension, die das Konzept selbst definiert, eine 1 steht, und ansonsten 0. Die Verwendung dieses Vektors bedeutet, dass ein Konzept immer Teil seines eigenen Kontextes ist. a_{id} ist auf 2 festgelegt, da *direkte* Beziehungen eines Konzeptes ein großes Gewicht bei der Bestimmung der Verwandtheit haben sollen.

v_{br} enthält die übergeordneten Konzepte (▹C.2). a_{br} ist auf 2 festgelegt, da eine gemeinsame Einordnung unter ein allgemeineres Konzept einen wichtigen Hinweis auf Verwandtschaft liefert.

v_{nr} enthält die untergeordneten Konzepte. Der Wert von a_{nr} ist 1, da die gemeinsame Zuordnung eines konkreteren Begriffes kein gutes Indiz für Verwandtheit ist.

v_{in} enthält die Konzepte, die über einen Hyperlink auf das gegebene Konzept verweisen (▹C.2). a_{in} ist auf 2 festgelegt, da gemeinsames Auftreten in einem Artikel ein gutes Indiz für Verwandtschaft ist.

v_{out} enthält die Konzepte, auf die das gegebene Konzept per Hyperlink verweist. Der Wert von a_{out} ist 1, da Gemeinsamkeiten bezüglich referenzierter Artikel kein starkes Indiz für Verwandtheit sind.

v_{rel} enthält die Konzepte, die im vorhinein als „verwandt“ gekennzeichnet wurden — diese Zuweisung basiert insbesondere auf der Relation `bilink`, die Konzepte verbindet, die sich gegenseitig referenzieren (▹C.2, ▹9.1.3). Der Wert von a_{rel} ist -1; zwar ist diese Beziehung ein sehr deutliches Zeichen für Verwandtheit, dasselbe Konzept ist aber bereits durch v_{in} und v_{out} berücksichtigt; so ist es sogar notwendig, den Verwandtheitswert etwas zu dämpfen (siehe Rechenbeispiel unten).

v_{sim} enthält Konzepte, die als „ähnlich“ erkannt wurden, insbesondere auf Grund von Language-Links. a_{sim} ist auf 2 festgelegt, da ähnliche Konzepte natürlich auch stark verwandt sind.

Ein alternative Möglichkeit, v_{sim} zu verwenden, wäre es, die Featurevektoren aller ähnlichen Konzepte zu berechnen und zu dem Featurevektor des ursprünglichen Konzeptes hinzuzuaddieren — diese Möglichkeit wurde hier nicht weiter verfolgt und verbleibt als Gegenstand künftiger Forschung.

Einige Beispiele dafür, wie viele „Punkte“ sich auf diese Weise in typischen Fällen für den Verwandtheitswert sim_{cos} eines Paares von Konzepten ergeben:

- Wenn A ein übergeordnetes Konzept von B ist, dann gilt $A \in v_{br}(B)$ und $B \in v_{nr}(A)$, sowie natürlich immer $A \in v_{id}(A)$ und $B \in v_{id}(B)$. Damit ergibt sich für das Skalarprodukt $v(A) \circ v(B)$ eine Punktzahl von $2 \cdot 2 + 1 \cdot 2 = 6$.
- Wenn A und B ein gemeinsames übergeordnetes Konzept C haben, also bezüglich C *kohyponym* sind, dann gilt $C \in v_{br}(A)$ und $C \in v_{br}(B)$. Damit ergibt sich für $v(A) \circ v(B)$ eine Punktzahl von $2 \cdot 2 = 4$.
- Wenn A und B ein gemeinsames untergeordnetes Konzept C haben, dann gilt $C \in v_{nr}(A)$ und $C \in v_{nr}(B)$. Damit ergibt sich eine Punktzahl von $1 \cdot 1 = 1$.
- Wenn A per Hyperlink auf B verweist, dann gilt $A \in v_{in}(B)$ und $B \in v_{out}(A)$, sowie natürlich immer $A \in v_{id}(A)$ und $B \in v_{id}(B)$. Damit ergibt sich für das Skalarprodukt $v(A) \circ v(B)$ eine Punktzahl von $2 \cdot 2 + 1 \cdot 2 = 6$.
- Wenn A und B beide von einem Konzept C per Hyperlink referenziert werden, also bezüglich C *kookkurrent* sind, dann gilt $C \in v_{in}(A)$ und $C \in v_{in}(B)$. Damit ergibt sich für $v(A) \circ v(B)$ eine Punktzahl von $2 \cdot 2 = 4$.
- Wenn A und B beide per Hyperlink auf ein Konzept C verweisen, dann gilt $C \in v_{out}(A)$ und $C \in v_{out}(B)$. Damit ergibt sich eine Punktzahl von $1 \cdot 1 = 1$.
- Wenn sich A und B gegenseitig per Hyperlink referenzieren, also im vorhinein über die `bilink`-Relation als „verwandt“ markiert wurden, dann gilt $A \in v_{out}(B)$, $B \in v_{out}(A)$, $A \in v_{in}(B)$, $B \in v_{in}(A)$ sowie außerdem $A \in v_{rel}(B)$ und $B \in v_{rel}(A)$. Damit ergibt sich eine Punktzahl von $2 \cdot (2 + 1 - 1) + (2 + 1 - 1) \cdot 2 = 8$.
- Wenn A und B einen Language-Link gemeinsam haben, sie im vorhinein also als „ähnlich“ markiert wurden, dann gilt $A \in v_{sim}(B)$ und $B \in v_{sim}(A)$. Damit ergibt sich eine Punktzahl von $2 \cdot 2 + 2 \cdot 2 = 8$.

Durch die Anpassung der Koeffizienten kann also das Gewicht bestimmt werden, das bestimmten Konstellationen von Konzepten bezüglich der im Datenmodell erfassten Relationen für die Bestimmung der Verwandtheit der Konzepte beigemessen werden soll. Die absoluten Werte spielen dabei allerdings keine Rolle, da bei der Bestimmung des Kosinus die Vektoren ja normiert werden. Nur die Verhältnisse bleiben erhalten.

Die Featurevektoren geben direkte Verknüpfungen in dem Graphen wieder, der durch die Beziehungen zwischen den Konzepten definiert wird. Diese direkten Kanten stellen Verbindungen

von Konzepten zu ihrem Verwendungskontext dar, also zum Beispiel zu Kategorien, denen sie zugeordnet sind, oder Artikel, in denen auf sie verwiesen wird. Betrachtet man die Ähnlichkeit zwischen Featurevektoren, so werden dadurch Pfade der Länge zwei in diesem Graphen abgedeckt — das entspricht gerade den Kookkurrenten der Konzepte, deckt also diejenigen Konzepte ab, die gemeinsam in einer Kategorie enthalten sind oder zusammen in einem Artikel referenziert werden. Der Vergleich von Featurevektoren deckt also *syntagmatische* Beziehungen zwischen den Konzepten auf[1]. Laut [Nakayama u. a. (2007b)] ist eine Suchtiefe von zwei Kanten für die Bestimmung semantischer Verwandtheit ausreichend, und [Strube und Ponzetto (2006)] kommen zu dem Ergebnis, dass eine Beschränkung der Suchtiefe unter Umständen die Ergebnisse sogar verbessern kann, wenngleich dort eine Tiefe von maximal vier Kanten als optimal befunden wird.

Es wäre interessant, (signifikante) Kookkurrenten direkt mit in die Featurevektoren aufzunehmen — damit würden auch Pfade der Länge drei und vier beim Vergleich der Vektoren berücksichtigt und es würden somit auch *paradigmatische* Beziehungen zwischen den Konzepten gefunden [Biemann u. a. (2004)]. Eine Untersuchung dieser Erweiterung des hier vorgeschlagenen Verfahrens geht jedoch über den Rahmen dieser Arbeit hinaus und bietet Gelegenheit für weitere Forschung.

Experiment

	Alle	Nomen
Anzahl	188	113
Korrelation (Cos)	0,31	0,39
Korrelation (Tani)	0,22	0,43
Korrelation (PL+Avg)	0,36	0,43
Korrelation (PL+Best)	0,43	0,49

`concept-relatedness-eval.tsv`, *siehe* ▸F.5

Tabelle 14.1.: Evaluation der Verwandtheit von Konzepten

Die oben beschriebene Bestimmung der semantischen Verwandtheit von Konzepten wurde evaluiert, indem die auf diese Weise berechneten Verwandtheitswerte verglichen wurden mit der menschlichen Beurteilung von Verwandheit. Dazu wurde der Datensatz ZG222[2] [Zesch und Gurevych (2006)] herangezogen: er enthält 222 Wortpaare und zu jedem Paar einen Wert zwischen 0 und 4, der aus den Urteilen von 24 Probanden berechnet wurde.

Jedem Wortpaar in ZG222 wurde nun manuell ein Paar von Konzepten aus der deutschsprachigen Wikipedia zugewiesen — das war für 188 Paare möglich, in den restlichen Fällen gab es in der deutschsprachigen Wikipedia für mindestens eines der Wörter kein passendes Konzept. Das war insbesondere häufig der Fall für Adjektive wie „*angehend*“ oder „*neu*“, Verben wie „*belieben*“ oder „*drängeln*“, aber auch für sehr allgemeine Begriffe wie „*Übersicht*“ oder sehr spezielle wie „*Kursstätte*“. Die Zuordnung von Konzepten zu dem Originaldatensatz ist in Anhang F.4 wiedergegeben.

[1] Bezogen auf eine ganze Wiki-Seite als Kontext, nicht auf einen einzelnen Satz, wie es typischerweise bei der Analyse der Kookkurrenzen von Wörtern der Fall ist.

[2] Verfügbar unter <http://www.ukp.tu-darmstadt.de/data/semRelDatasets>.

Für jedes Paar von Konzepten wurde ein Verwandtheitswert bestimmt, nach dem oben beschriebenen Verfahren, jeweils einmal unter Verwendung des Kosinus-Maßes und einmal der gewichteten Tanimoto-Ähnlichkeit. Die so gewonnenen Verwandheitswerte wurden auf ihre statistische Korrelation mit den in ZG222 ermittelten Werten untersucht. Die Korrelation ist, der Literatur über semantische Verwandtheit in der Wikipedia folgend [Zesch u. a. (2007b), Strube und Ponzetto (2006)], in der Tabelle 14.1 als r-Wert nach Pearson angegeben. Die Korrelation wurde jeweils einmal für alle 188 Konzeptpaare bestimmt, und einmal für alle 113 Konzeptpaare, die zu Paaren von *Substantiven* in ZG222 korrespondieren — es ist nämlich zu erwarten, dass sich die Daten einer Enzyklopädie wie Wikipedia insbesondere zur Untersuchung von Substantiven eignen.

Zum Vergleich sind in der Tabelle zusätzlich einige Ergebnise aus [Zesch u. a. (2007b)] wiedergegeben, nämlich die Werte für PL+Avg und PL+Best. PL basiert auf der *taxonomischen Pfadlänge*, also der Anzahl von Kanten im Kategoriegraph, die die kürzeste Verbindung zwischen zwei Kategorien bilden. Dabei wird aus jeweils einer Kategorie jedes Konzeptes ein Paar gebildet, und der PL-Wert für dieses Paar von Kategorien berechnet. Der Wert PL+Avg ergibt sich aus der durchschnittlichen Distanz für alle möglichen Kategoriepaare, PL+Best verwendet die minimale Distanz, die für irgendein Paar von Kategorien gefunden wurde. Dieses Verfahren ist deutlich aufwändiger als das oben beschriebene, da unter Umständen der gesamte Kategoriegraph betrachtet werden muss, statt nur lokal definierte Featurevektoren zu vergleichen. Zudem werden kurze Pfade, die höchstens zwei Kanten lang sind, von den Featurevekoren mit abgedeckt.

Die Ergebnisse für Substantivpaare zeigen mit 0,43 für die Tanimoto-Ähnlichkeit eine deutliche, wenn auch nicht überragende Korrelation mit dem menschlichen Urteil, wie es in ZG222 wiedergegeben ist — Zesch et. Al. erreichen für Paare von Nomen eine Korrelation von 0,49 mit dem aufwändigeren PL+Best-Maß und 0,43 mit PL+Avg [Zesch u. a. (2007b)].

Das Ergebnis über alle Konzeptpaare ist mit 0,30 für das Kosinus-Maß gerade noch als signifikant anzusehen (Zesch et. Al. erreichen hier 0,43 mit PL+Best und 0,36 mit PL+Avg). Interessant ist, dass das Ergebnis für die Tanimoto-Ähnlichkeit über alle Kozepte schlechter, für die Menge der Substantive aber deutlich besser ist. Diesen Umstand näher zu betrachten sowie die Effekte anderer Ähnlichkeitsmaße zu untersuchen, verbleibt aber als Gegenstand künftiger Forschung.

Ebenfalls auffällig ist, dass die Ergebnisse insgesamt schlechter sind als für den Fall einer automatischen Bedeutungszuweisung, wie weiter unten in ▹14.3 beschrieben. Zwar war zu erwarten, dass diese aufgrund der Methode der Konzeptauswahl durch Maximierung der Verwandheit (▹14.2) insgesamt höhere Verwandheitswerte liefert, jedoch schien es ebenfalls wahrscheinlich, dass die Korrelation mit dem menschlichen Urteil unter falschen Zuordnungen, und damit überhöhten Verwandheitswerten, leiden würde.

Es zeigt sich aber stattdessen, dass bei der manuellen Termzuordnung insgesamt häufig ein zu geringer Verwandheitswert errechnet wurde. Ein möglicher Grund hierfür ist, dass die in ZG222 von Menschen bestimmten Verwandtheitswerte auf einer unbekannten Zordnung von Konzepten basieren: zwar ist in [Zesch und Gurevych (2006)] beschrieben, dass den Probanden als Hilfe für die Bewertung der Verwandheit Glossen und Synonyme der Terme vorgegeben wurden, so dass die zu bewertenden Bedeutungen eindeutig waren. Sogar ein Nachschlagen des Begriffes in der Wikipedia wurde dort angeboten, was impliziert, dass eine Zuordnung von Termen zu Wikipedia-Artikeln bereits vorgenommen wurde. Diese Zuordnung zu kennen wäre für die Evaluation von

WikiWord ausgesprochen hilfreich gewesen, sie ist jedoch in dem bereitgestellten Datensatz nicht wiedergegeben. Daraus ergibt sich ein grundsätzliches Problem mit ZG222: ein direkter Vergleich mit Verwandheitswerten, die für Konzeptpaare in WikiWord ermittelt wurden, beruht immer auf der Annahme, dass die Zuordnung von Konzepten zu Wörtern dort genau gleich erfolgte wie hier. Diese Annahme scheint in Anbetracht der relativ schlechten Korrelation insbesondere für Nicht-Substantive nicht in ausreichendem Maße gültig zu sein.

14.2. Disambiguierung durch Maximierung der Kohärenz

Die Verwandtheitsfunktion für Konzepte, $r_{sem}(c_1, c_2)$, kann nun verwendet werden, um einzelnen Termen aus einer vorgegebenen Menge eine Bedeutung zuzuweisen. Die Idee ist, dass sich die Terme gegenseitig disambiguieren: für jeden Term wird die Menge seiner möglichen Bedeutungen bestimmt und dann für jeden Term diejenige Bedeutung gewählt, die dazu führt, dass die paarweise Verwandtheit der gewählten Terme (die *Kohärenz*) insgesamt maximal wird (vergleiche [Ponzetto und Strube (2007c), Zesch u. a. (2007b)]). Genauer:

Sei $W = \{w_1, w_2, \ldots, w_n\}$ eine Menge von Termen und $s(w) = \{c \mid S(w, c)\}$ die Funktion, die zu einem Term w die Menge der Konzepte (Bedeutungen) liefert, die dem Term über die Bedeutungsrelation S zugeordnet sind (▷9.1.3, ▷C.2). Dann ist $D_k = \{c_{1,k}, c_{2,k}, \ldots, c_{n,k}\}$ eine *Belegung* derart, dass $c_{i,k} \in s(w_i)$ ist, also für jeden Term ein Bedeutung gewählt wurde.

Der *Score* g_r einer Belegung D_k bezüglich der *Kohärenz* ist gegeben durch die normierte Summe der paarweisen Verwandtheit der Konzepte:

$$g_r(D_k) = \frac{2 \cdot \sum_{i,j}^{n} r_{sem}(c_{i,k}, c_{j,k})}{n \cdot (n-1)}$$

Alternativ zur Verwandtheit der beiden Konzepte kann auch betrachtet werden, wie häufig ein Term als Bezeichnung für ein bestimmtes Konzept verwendet wird — diese Frequenz der Termzuordnung ist gegeben durch $f(w_i, c_{i,k})$. Daraus lässt sich ein Wert für die *Popularität* $pop(w_i, c_{i,k})$ dieser Bedeutungszuweisung ableiten, mit der Eigenschaft, dass der Wert zwischen 0 und 1 liegt und mit wachsendem f monoton steigt:

$$pop(w_i, c_{i,k}) = 1 - \frac{1}{\sqrt{f(w_i, c_{i,k})} + 1}$$

Der *Score* einer Belegung bezüglich der *Popularität* der einzelnen Konzepte ergibt sich dann aus dem arithmetischen Mittel:

$$g_f(D_k) = \frac{\sum_{i}^{n} pop(w_i, c_{i,k})}{n}$$

Diese Scores sind beide auf einen Wert zwischen 0 und 1 normiert und lassen sich linear über einen festen Koeffizienten p (den *Popularitäts-Bias* des Scores) kombinieren:

$$g(D_k) = g_f(D_k) \cdot p + g_r(D_k) \cdot (1 - p); 0 \leq p \leq 1$$

Die gesuchte Disambiguierung ist die bezüglich des Scores *beste* Belegung, $D_{max} = D_{k_{max}}$, wobei $k_{max} = \arg\max_k g(D_k)$ ist. Es wird also diejenige Kombination von Bedeutungen für die Terme gewählt, die den besten Score ergibt.

Für die Implementation bedeutet das, dass alle Belegungen, also alle möglichen Kombinationen von Bedeutungszuweisungen, ausprobiert und bewertet werden müssen — das sind n^d Belegungen und $\frac{n^2-n}{2}$ zu vergleichende Paare von Konzepten pro Belegung, wobei n die Anzahl der Terme und d die Anzahl von Bedeutungen pro Term ist. Für jeden Vergleich von zwei Featurevektoren fallen so viele Multiplikationen und Additionen an, wie Einträge in dem Feature-Vektor sind; Diese Größe hängt wesentlich von der Knotengradverteilung im Hyperlink-Graphen ab (▹11.4) und dürfte in der Größenordnung von 100 liegen. Für vier Terme mit jeweils fünf Bedeutungen ergeben sich $4^5 \cdot \frac{4^2-4}{2} = 1024 \cdot 6 = 6144$ Vergleiche von Featurevektoren, also insgesamt etwa 614 400 Operationen. Für sechs Terme wären es schon 11 664 000, dieses Verfahren skaliert also nicht für große Mengen von Termen. Um es auf Fließtext anwenden zu können, würde sich ein *Sliding Window* von insgesamt drei bis fünf Termen anbieten.

Experiment

Zur Evaluation der automatischen Bedeutungszuweisung (*Disambiguierung*) nach dem oben beschriebenen Verfahren wird wieder der Datensatz `ZG222` nach [Zesch und Gurevych (2006)] mit den manuell zugeordneten Konzeptpaaren (▹14.1, ▹F.4) herangezogen. Tabelle 14.2 zeigt, wie sich die Ergebnisse der verschiedenen Möglichkeiten der automatischen Zuordnung zu den manuell zugewiesenen Konzepten verhalten. Die angegebenen Werte beziehen sich auf die 131 Wortpaare bzw. 92 Substantivpaare, für die sowohl manuell als auch automatisch beide Konzepte zugeordnet werden konnten. Die Spalte `voll` gibt an, wieviel Prozent der Konzeptpaare genau den manuell ausgewählten Konzeptpaaren entsprechen, die Spalte `teilw.` gibt an, bei wieviel Prozent mindestens eines der Konzepte übereinstimmt. Die letzte Spalte, `Korr.`, ist die statistische Korrelation (r-Wert nach Pearson) mit den in [Zesch und Gurevych (2006)] ermittelten Verwandtheitswerten.

Wünschenswert wäre eine hohe Übereinstimmung zwischen automatisch zugeordneten Konzepten mit den manuell zugewiesenen und gleichzeitig eine starke Korrelation mit den in `ZG222` vorgegebenen Verwandtheitswerten. Zunächst scheinen sich diese Werte aber gegenläufig zu entwickeln: Wird ausschließlich die Kohärenz betrachtet ($p = 0$), ergibt sich ein relativ guter Korrelationskoeffizient von 0,4 (über alle Konzepte, bei Verwendung des Kosinus-Maßes) aber eine schlechte Übereinstimmung der gewählten Konzepte (37% volle und 75% teilweise Übereinstimmung). Wird dagegen nur die Popularität der Bedeutungen (also die Frequenz der Bedeutungszuweisung) beachtet ($p = 1$), so ergibt sich eine geringe Korrelation von 0,26, aber eine größere Übereinstimmung der gewählten Konzepte (43% voll und 91% teilweise).

(a) Alle

Methode	Pop-Bias (p)	voll	teilw.	Korr.
Kohärenz, Cosinus	0	37.4%	74.8%	0,41
Kohärenz, Cosinus	0,25	40.5%	87.0%	0,41
Kohärenz, Cosinus	1	42.7%	90.8%	0,26
Kohärenz, Tanimoto	0	35.1%	74.0%	0,32
Kohärenz, Tanimoto	0,25	41.2%	87.8%	0,29
Kohärenz, Tanimoto	1	42.7%	90.8%	0,22
PL+Avg				0,36
PL+Best				0,43

(b) Substantive

Methode	Pop-Bias (p)	voll	teilw.	Korr.
Kohärenz, Cosinus	0	39.1%	68.5%	0,38
Kohärenz, Cosinus	0,25	43.5%	85.9%	0,39
Kohärenz, Cosinus	1	47.8%	92.4%	0,39
Kohärenz, Tanimoto	0	37.0%	69.6%	0,26
Kohärenz, Tanimoto	0,25	45.7%	90.2%	0,23
Kohärenz, Tanimoto	1	47.8%	92.4%	0,34
PL+Avg				0,43
PL+Best				0,49

`sense-eval.tsv`, *siehe* ▸*F.5* *bzw.*. `times-global-NN.tsv`, *siehe* ▸*F.5*

Tabelle 14.2.: Evaluation der automatischen Bedeutungszuweisung

Experimentell wurde ein Wert von 0,25 für p ermittelt, der einen guten Kompromiss darzustellen scheint: die Übereinstimmung der gewählten Konzepte bleibt hoch (41% voll und 89% teilweise) und der Korrelationskoeffizient steigt bis auf die Marke von 0,4. Die Verwendung der Tanimoto-Ähnlichkeit erweist sich hier als deutlich unterlegen: bei ähnlichen Werten für die Übereinstimmung der Konzepte ergeben sich deutlich schlechtere Werte für die Korrelation der Verwandtheitswerte. Betrachtet man die Werte nur für Paare von Substantiven, so ergeben sich erwartungsgemäß bessere Werte für die Übereinstimmung der Bedeutungszuordnung, aber überraschenderweise schlechtere Werte für die Korrelation der Verwandtheitswerte — dieser Zusammenhang wird im nächsten Abschnitt ▸14.3 genauer betrachtet.

Der Umstand, dass sich bei deutlich schlechterer Übereinstimmung mit den manuell gewählten Konzepten trotzdem eine gute Korrelation der Verwandtheitswerte erreichen lässt, scheint zunächst verwunderlich. Die Erklärung ist wohl darin zu suchen, dass in den Fällen, in denen das automatische Verfahren ein anderes Konzept gewählt hat als das, was manuell zugeordnet wurde, dieses Konzept selten ein *völlig* anderes ist — vielmehr fällt die automatische Wahl häufig auf ein Konzept, das dem manuell gewählten *verwandt* ist. Die Verwandtheit zwischen den gewünschten und tatsächlich zugeordneten Konzepten zu messen und näher zu untersuchen bietet sich als Gegenstand künftiger Forschung an.

Bei der Betrachtung solcher Fehlzuordnungen ist insbesondere das Phänomen der *Überspezialisierung* zu beobachten: da ja Konzeptpaare so gewählt werden, das die Konzepte eine möglichst große Verwandtheit miteinander aufweisen, kommt es vor, dass zwei sehr spezielle Bedeutungen gewählt werden. Zum Beispiel wurde für das Wortpaar „*Struktur*“ und „*Entwicklung*“ bei ausschließlicher Verwendung des Kohärenzwertes ($p = 0$) das Konzeptpaar Herz#Struktur und Herz#Entwicklung gewählt — diese beiden Konzepte sind natürlich stark verwandt und haben die gesuchte Bezeichnung. Da auch im Original-Datensatz die Verwandtheit von „*Struktur*“ und „*Entwicklung*“ überdurchschnittlich bewertet war, trägt diese Wahl zu einer guten Korrelation der Verwandtheitswerte bei. Die gewählten Konzepte entsprechen aber natürlich nicht denen, die den Termen manuell zugeordnet wurden, nämlich die allgemeinen Konzepte Struktur und Entwicklung. Wird aber die Frequenz der Bedeutungszuordnung, also die Popularität der Bedeutungen, berücksichtigt (und sei es nur mit dem Faktor 0,25), so wählt das automatische Verfahren die korrekten Konzepte — die allerdings als weniger stark (wenn auch immer noch deutlich) verwandt angegeben werden, was der Korrelation der Verwandtheitswerte abträglich ist.

Ein weiteres Problem sind sehr allgemeine Konzepte wie Beschreibung: dem Term „*Beschreibung*“ sind zwar im WikiWord-Datensatz eine Vielzahl von Bedeutungen zugeordnet, diese sind aber alle sehr speziell. Das allgemeine Konzept Beschreibung wird in der Wikipedia nicht behandelt, daher muss die automatische Bedeutungszuweisung hier fehlschlagen.

Zu untersuchen wäre, wie sich verschiedene Cutoff-Verfahren (▸10.4) auf die Ergebnisse der automatischen Disambiguierung auswirken, insbesondere auch in Hinblick auf das Problem der Überspezialisierung.

Desweiteren macht sich auch hier wieder das Problem bemerkbar, das schon in ▸14.1 angesprochen wurde: welche Konzepte (Bedeutungen der Terme) die Evaluatoren in [Zesch und Gurevych (2006)] tatsächlich vorgegeben bekamen, als sie die Verwandtheit beurteilen sollten, ist unbekannt. Deshalb sind die Verwandheitswerte, die für die hier vorgenommene manuelle Zuordnung von

Konzeptpaaren bestimmt wurden, nur bedingt mit denen aus dem Originaldatensatz vergleichbar. Es wäre daher angebracht, weitere Untersuchungen auf einer Datenbasis durchzuführen, die direkt auf die Evaluation von Systemen zur automatischen Disambiguierung zugeschnitten ist, wie zum Beispeil *Senseval* [Mihalcea u. a. (2004)].

14.3. Semantische Verwandtheit zwischen Termen

In der Praxis ist es häufig verlangt, die semantische Verwandtheit von zwei oder mehr *Termen* zu bestimmen — diese Aufgabe lässt sich als Kombination von Disambiguierung und Bestimmung der Verwandtheit von Konzepten verstehen: da Terme keine semantischen, sondern lexikalische Einheiten sind, muss den Termen zunächst eine Bedeutung zugewiesen werden, bevor ihre Verwandtheit bestimmt werden kann. Diese Art, die Verwandtheit zweier Terme zu messen, bietet sich insbesondere bei der Verwendung des oben beschriebenen Verfahrens zur Disambiguierung an, da es den entsprechenden Wert für die Verwandtheit bereits als $g(D_{max})$ liefert.

Experiment

Die Evaluation der Bestimmung der semantischen Verwandtheit von Termen nach dem oben beschriebenen Verfahren entspricht genau der Evaluation für das Disambiguierungsverfahren (▹14.2), nur dass ein anderer Aspekt des Ergebnisses im Vordergrund steht, nämlich die Korrelation der Verwandtheitswerte. Die *Scores* $g(D_{max})$, die bei der Disambiguierung für das beste Konzeptpaar berechnet wurden, werden als Maß für die Verwandtheit der betreffenden Terme interpretiert und mit den in [Zesch und Gurevych (2006)] ermittelten Verwandtheitswerten für die Termpaare in `ZG222` verglichen, indem ihre statistische Korrelation (r-Wert nach Pearson) bestimmt wird.

Das beste Ergebnis wird dabei, wie in Tabelle 14.2 zu sehen ist, mit 0,41 für alle Konzepte unter Verwendung des Kosinus-Maßes erzielt. Das ist interessanterweise besser als das Ergebnis von 0,30, das in ▹14.1 für die manuell gewählten Konzeptepaare erzielt wurde, aber noch etwas schlechter als das Ergebnis von 0,43, das mit dem aufwändigen PL-Verfahren erzielt wurde [Zesch u. a. (2007b)]. Überraschenderweise sind die Ergebnisse für Substantivpaare hier erheblich schlechter, der beste Wert von 0.39 liegt deutlich unter dem Ergebnis von 0,43, das für manuell gewählte Terme erzielt werden konnte, und erst recht unter dem Ergebnis für das PL-Maß mit 0,49. Das ist insbesondere deshalb überraschend, da die Übereinstimmung mit den manuell gewählten Konzepten für Substantivpaare besser ist, und diese manuell gewählten Paare für Substantivpaare die besseren Werte geliefert hatten ▹14.1. Weshalb die Ergebnisse der automatischen Disambiguierung ausgerechnet für Substantivpaare schlechter sind, obwohl das die Wortart ist, über die Wikipedia vorwiegend Informationen bietet, scheint einer näheren Untersuchung zu bedürfen.

In ähnlichen Experimenten zur Disambiguierung und Bestimmung der semantischen Verwandtheit auf Basis von Informationen aus der Wikipedia, mit etwas anderen Verfahren und auf anderen Datensätzen, wurden vergleichbare Korrelationen zwischen 0.32 und 0.57 gegenüber dem menschlichen Urteil erreicht [Strube und Ponzetto (2006), Ponzetto und Strube (2007c)].

Insgesamt zeigt sich, dass die Daten von WikiWord grundsätzlich geeignet sind, als Datenbasis für die Bestimmung der semantischen Verwandtheit von Termen und Konzepten sowie für die automatische Disambiguierung von Wörtern im Kontext zu dienen. Es wird aber auch deutlich, dass einerseits die betrachteten Verfahren weiter verfeinert werden können, andererseits die zur Evaluation verwendete Datenbasis nicht optimal war, da zu den Wortpaaren keine Zuordnung von Konzeptpaaren bereitgestellt wurde und damit unklar blieb, auf welche Konzepte sich die vorgegebenen Verwandtheitswerte bezogen.

Auch wurden Probleme mit den von WikiWord gewonnenen Daten deutlich: einerseits das zu erwartende Fehlen von Informationen zu Adjektiven und Verben, andererseits das Problem der Überspezialisierung bei der Disambiguierung und das Fehlen von Artikeln über allgemeine Konzepte. Hier zeigt sich noch einmal, dass die von WikiWord aus der Wikipedia gewonnenen Daten weniger einen Ersatz für als vielmehr eine Ergänzung zu den Daten aus klassischen Wörterbüchern und Thesauri darstellen.

15. Zusammenfassung

In der vorliegenden Diplomarbeit wurde die Extraktion eines multilingualen Thesaurus aus den Daten der Wikipedia behandelt. Zunächst wurden einige Grundlagen über Wikipedia und Thesauri erörtert (▹1) und der Stand der Forschung auf diesem Gebiet evaluiert (▹2). Dann wurde eine Methode entworfen, mit der die relevanten Informationen aus dem Wiki-Text extrahiert werden können, sowie ein Schema zur Interpretation dieser Informationen zum Aufbau eines Thesaurus entworfen (▹4). Desweiteren wurde ein Prototyp (*WikiWord*) entwickelt, der diese Methode implementiert (▹8 und ▹9) und die gewonnenen Daten in einer geeigneten Struktur in einer relationalen Datenbank speichert (▹C). Für den so entstandenen Thesaurus wurde ein Verfahren entwickelt, mit dem er in ein Austauschformat exportiert werden kann, das auf den Standards RDF und SKOS beruht (▹10). WikiWord umfasst insgesamt über 31 000 Zeilen Programmcode in fast 300 Klassen (▹G).

Der Prototyp wurde verwendet, um die Daten einiger Wikipedia-Projekte zu importieren und aus ihnen einen multilingualen Thesaurus zu erzeugen. Dieser wurden in Hinblick auf seine Validität sowie seine Nützlichkeit für verschiedene Aufgaben aus dem Bereich der automatischen Sprachverarbeitung sowie des Information Retrieval evaluiert (▹IV).

15.1. Besonderheiten

Das WikiWord-System, das in der vorliegenden Arbeit beschrieben ist, implementiert eine „*end-to-end*-Lösung“ für die Aufgabe der automatischen Erstellung eines multilingualen Thesaurus, wobei es auf die Daten der Wikipedia zurückgreift. Dabei werden vor allem Informationen verwendet, die durch die Verknüpfungsstruktur sowie die Kategorisierung und die Verwendung von Vorlagen in den Wiki-Seiten kodiert sind. Eigenschaften, die WikiWord von den in ▹2 beschriebenen Systemen und Verfahren abgrenzt, sind insbesondere:

- Zur Bestimmung der Bezeichnungen für die durch Artikel repräsentierten Konzepte wurde unter anderem der Link-Text von Wiki-Links verwendet, die auf diese Artikel verweisen (siehe ▹8.9). Dabei wurde auch die Frequenz dieser Zuordnungen gespeichert. Die Verwendung des Link-Textes ist zwar in der Literatur durchaus bereits untersucht worde, jedoch in der Regel nicht im Kontext einer Thesaurus-Struktur, sondern nur in Bezug auf Disambiguierungssysteme, insbesondere für die *Named Entity Recognition* (vergleiche ▹2.3 und ▹2.4).

- Zur Bestimmung der Bezeichnungen für Konzepte wurden, neben den gängigen Quellen wie Seitentitel und Weiterleitungen sowie den zum Teil ebenfalls schon untersuchten Begriffsklärungsseiten, zusätzliche Quellen erschlossen, insbesondere die bei der Kategorisierung angegebenen *Sort-Keys* sowie die Verwendung der *Magic Words* `DISPLAYTITLE` und `DEFAULTSORT` (siehe ▹9.1).

- Für die Verarbeitung von Begriffsklärungsseiten wurde ein Algorithmus entwickelt (▹8.10, ▹13), der deutlich über das hinaus geht, was in bisherigen Arbeiten zu dem Thema beschrieben wurde (vergleiche ▹2.3).

- Es wurde ein Verfahren entwickelt, das zu Kategorien einen „Hauptartikel“ bestimmt, und Kategorie und Artikel dann auf dasselbe Konzept im Thesaurus abbildet (▹9.1.2). Dieses Verfahren ist insbesondere sprachunabhängig und basiert nicht auf einer Stammformreduktion. Ein solches Verfahren ist in den einschlägigen Arbeiten bisher nicht beschrieben worden.

- Es wurde auf der Basis verschiedener Merkmale, wie den verwendeten Kategorien und Vorlagen, eine Klassifikation von Seiten und Konzepten vorgenommen (siehe ▹8.5 und ▹8.6). Eine solche Klassifikation von Konzepten wurde zwar zuvor im Rahmen der *Named Entity Recognition* beschrieben, aber nicht mit einem Thesaurus kombiniert.

- Die Nutzung von Language-Links zur Bestimmung der semantischen Ähnlichkeit zwischen Konzepten wurde erstmals untersucht (siehe ▹9.1.3). Überhaupt gibt es zwar eine große Zahl von Arbeiten zur Bestimmung von semantischer *Verwandtheit* unter Verwendung der Wikipedia (vergleiche ▹2.3), semantische *Ähnlichkeit* wird jedoch kaum behandelt.

- Die Kombination der Daten aus mehreren Wikipedias durch Zusammenfassen äquivalenter Konzepte aus den verschiedenen Sprachen, identifiziert aufgrund ihrer Language-Links, wurde ebenfalls zuvor nicht beschrieben. Lediglich [Auer u. a. (2007)] nutzt die Language-Links, um Glossen in mehreren Sprachen anzubieten. Dabei dient allerdings die englische Wikipeda als Basis für die Datenstruktur, während WikiWord alle Sprachen gleichberechtigt behandelt. Auch wird die Strukturinformation aus anderen Wikipedias nicht verwendet. In [Hammwöhner (2007b)] wird zwar der Abgleich von Kategoriestrukturen verschiedener Wikis mit Hilfe der Language-Links vorgeschlagen, eine Vereinigung der Strukturen wird jedoch nicht betrachtet.

- WikiWord berücksichtigt Konzepte, die im Wiki nur als Abschnitte von Artikeln, nicht als eigenständige Artikel, repräsentiert sind (▹9.1.1).

- Außerdem werden Konzepte berücksichtigt, auf die zwar verwiesen wird, die aber keinen eigenen Artikel haben. Obwohl zu solchen Konzepten wenig Informationen vorliegen, können sie z. B. zur Indexierung von Dokumenten dennoch nützlich sein.

- Die von WikiWord erstellte Datenstruktur ist konzeptorientiert und abstrahiert weitgehend von den Strukturen der Wikipedia. Insbesondere werden, im Unterschied zu [Zesch u. a. (2007a)], Kategorien gar nicht direkt und Wiki-Seiten nur als Hilfskonstrukt gespeichert.

- Die Bereitstellung einer Abbildung auf RDF/SKOS, wie in ▹10 beschrieben, erlaubt eine unmittelbare Weiternutzung der erzeugten Daten in bestehenden Systemen zur Verarbeitung von Wissensnetzen. Einen ähnlichen Ansatz verfolgen vor allem [Gregorowicz und Kramer (2006)]. [Auer u. a. (2007)] bietet zwar auch eine solche Abbildung, allerdings in Hinblick auf die Repräsentation von Konzepteigenschaften und semantisch starken Beziehungen zwischen Konzepten, wie sie typischerweise aus Infoboxen extrahiert werden, nicht in

Hinblick auf eine Thesaurus-Struktur. [van Assem u. a. (2006)] untersuchen die Repräsentation von Thesauri in SKOS, beschränken sich aber auf manuell gewartete Thesauri wie MeSH.

- WikiWord verwendet keine automatische Sprachverarbeitung. Textuelle Inhalte als solches wurden bei der Verarbeitung ignoriert, mit Ausnahme der Extraktion des ersten Satzes als Definition (Glosse) des Konzeptes. Die verwendeten Verfahren sind weitgehend sprachunabhängig. Projektspezifische Muster, die Konventionen und Eigenheiten der betreffenden Wikipedias wiederspiegeln, können über einen Satz von Konfigurationsvariablen angepasst werden.
- Es wurde nicht versucht, konkrete semantische Beziehungen zwischen Konzepten oder Eigenschaften und Werte von Artikelgegenständen zu ermitteln. Die Verarbeitung solcher *strukturierten Daten* ist nicht Gegenstand dieser Arbeit.

WikiWord berücksichtigt also eine Vielzahl von Informationsquellen im Wiki-Text, von denen einige bislang völlig oder doch weitgehend übergangen wurden. Dadurch entsteht nicht nur ein Thesaurus, der für gängige Aufgaben wie Indexierung, Disambiguierung und Bestimmung von semantischer Ähnlichkeit und Verwandtheit verwendet werden kann. Die so gewonnenen Daten bieten auch eine reichere Datenbasis, auf die die verschiedenen in der Literatur beschriebenen Verfahren zur Nutzung und Analyse der Wikipedia angewendet werden können.

15.2. Ergebnis und Ausblick

Die Evaluation der mit WikiWord erzeugten Daten zeigt, dass WikiWord eine sehr breite Datenbasis liefert, die den in ▸5 definierten Anforderungen entspricht. Die Qualität der verschiedenen extrahierten Daten ist befriedigend bis gut, und in einigen Fällen wurden Verfahren angegeben, mit denen die Präzision der Daten auf Kosten der Abdeckung verbessert werden kann.

Evaluiert wurden insbesondere die Bedeutungsrelation (in ▸12.5) sowie die Bestimmung der semantischen Verwandtheit von Konzepten und die automatische Bedeutungszuweisung (in ▸14). Die Qualität der Zusammenfassung der Konzepte aus unterschiedlichen Sprachen zum Aufbau eines multilingualen Thesaurus wurde ebenfalls betrachtet und für gut befunden (▸11.3), hier scheinen aber weitergehende Untersuchungen und insbesondere ein Vergleich mit alternativen Verfahren sinnvoll.

Weitere Forschung verdient auch die Frage, inwieweit die teilweise unklaren oder fehlerhaften Kategoriestrukturen, wie sie in einigen Wikipedias vorliegen, durch geschickte Kombination der Informationen aus mehreren Wikis verbessert werden können, wie in [Hammwöhner (2007b)] vorgeschlagen.

Ein Punkt, für den die Evaluation der Praxistauglichkeit noch weitgehend aussteht, ist der Export nach RDF/SKOS (▸10). Zwar bieten diese Standards eine Orientierungshilfe in Bezug darauf, welche Inhalte wichtig sind und in welcher Form sie zu repräsentieren sind [W3C (2005)], Integrationstests mit einem oder mehreren realen Systemen, die SKOS-Daten verarbeiten können, wären jedoch angezeigt. Projekte, die vermutlich von den von WikiWord bereitgestellten Daten

profitieren könnten, und die sich daher für einen Integrationstest anbieten, sind unter anderem DBpedia [Auer u. a. (2007), DBpedia], Wortschatz [Quasthoff (1998), Wortschatz], Freebase [Freebase] sowie das freie Wörterbuchprojekt OmegaWiki [van Mulligen u. a. (2006), OmegaWiki].

Die Frage, inwieweit der von WikiWord erstellte multilinguale Thesaurus als Basis für ein Übersetzungssystem geeignet ist, wäre ebenfalls noch zu untersuchen. Insbesondere eine Evaluation des Nutzens der Informationen aus diesem Thesaurus für ein bestehendes System wäre aufschlussreich.

Die vorliegende Diplomarbeit beschreibt Methoden zur Extraktion von Wissen über Beziehungen von Konzepten zueinander sowie über die Bezeichnungen, die für Konzepte verwendet werden. Es wurde ein funktionsfähiger Prototyp entwickelt und auf die Daten einiger Wikipedias angewendet. Es konnte gezeigt werden, dass die beschriebenen Methoden die gestellten Anforderungen befriedigend erfüllen, und es wurde eine Anzahl von Ansätzen für weiterführende Untersuchungen aufgezeigt. Der Autor verbleibt in der Hoffnung, mit dieser Arbeit einen Beitrag dazu geleistet zu haben, Systeme zur automatischen Sprachverarbeitung und Information Retrieval dadurch zu verbessern, dass sie Zugang zu einer großen Menge detaillierten lexikalisch-semantischen Weltwissens erhalten.

Wissen ist nicht genug. Wir müssen es anwenden.

— J.W. Goethe

Teil V.

Anhang

A. Wiki-Text

Wiki-Text ist die Hypertext-Markup-Sprache, die von Wikis im Allgemeinen und, im Kontext dieser Arbeit, von MediaWiki im Speziellen verwendet wird. Sie dient dazu, Text zu formatieren und zu strukturieren, Verknüpfungen herzustellen oder weitere spezielle Funktionen der Wiki-Software anzusprechen. Dabei soll sie leichter zu lesen und zu schreiben sein als zum Beispiel HTML.

Leider gibt es bislang keine formale Grammatik oder Spezifikation für das Wiki-Markup, das MediaWiki verwendet; einen Überblick bieten [MW:EDIT] und [MW:SPEC]. Die Syntax für Wiki-Text ist, soweit sie für WikiWord relevant ist, fest in `PlainTextAnalyzer` kodiert. Die wichtigsten Markup-Elemente, die von MediaWiki verwendet werden, sind im Folgenden kurz beschrieben.

A.1. HTML

MediaWiki unterstützt neben den speziellen Konstrukten des Wiki-Markups auch HTML-Markup [W3C:HTML (1999)], insbesondere Tags wie **, * *, *<p>* und *
* sind erlaubt, mitsamt den möglichen Attributen, speziell auch *class*, *style*, *id* und *title*. Auch die in HTML definierten Zeichenreferenzen wie *&amul;* sowie die SGML-Syntax für Kommentare, *<!-- ...-->*, werden unterstützt.

Zusätzlich definiert MediaWiki eigene Tags, die derselben HTML- bzw. XML-artigen Syntax folgen, zum Beispiel das Tag *<nowiki>*, das als Escape dient und für den eingeschlossenen Text die Interpretation von Wiki-Markup unterbindet. Diese Nebenwirkung hat übrigens auch *<pre>* für vorformatierten Text. Erweiterungen (*Extensions*) für MediaWiki können beliebige zusätzliche Tags definieren, siehe ▹A.7.

A.2. Einfache Textformatierung

Ein leicht zu lesendes und zu schreibendes Markup für einfache Textformatierungen sind der Hauptgrund für die Existenz von Wiki-Text: die Verwendung von HTML wurde für zu wenig benutzerfreundlich befunden. So definiert MediaWiki, wie die meisten Wikis, spezielles Markup für die Formatierung von Text:

- Eine leere Zeile beginnt einen neuen Absatz (einzelne Zeilenumbrüche haben im Fließtext keine Bedeutung).
- Blöcke von Zeilen, die jeweils mit mindestens einem Leerzeichen beginnen, werden als vorformatierter Text behandelt (ähnlich wie bei *<pre>*, nur dass die Interpretation von Wiki-Markup nicht vollständig unterdrückt wird).

- Zwei Apostrophe markieren kursiven Text: `''kursiv''` wird zu *kursiv*; drei Apostrophe markieren fetten Text: `'''fett'''` wird zu **fett**.
- Kursiv- und Fettdruck lassen sich kombinieren: `'''fett ''kursiv'''''` wird zu **fett *kursiv***.
- Hoch- und Tiefstellung werden mit HTML-Markup realisiert, also mit `<sup>` und `<sub>`. Das gleiche gilt für unterstrichenen und durchgestrichenen Text mit `<u>` und `<s>`.
- Überschriften (Abschnitte) werden mit Gleichheitszeichen markiert: `= Überschrift =` erzeugt, wenn es für sich auf einer eigenen Zeile steht, eine Überschrift erster Ordnung, `== Überschrift ==` eine Überschrift zweiter Ordnung, und so weiter. Das HTML-Äquivalent, `<h1>`, `<h2>` usw. wird ebenfalls akzeptiert. Überschriften dienen automatisch als Anker, also als mögliche Sprungziele, die über Wiki-Links adressiert werden können.
- Einrückung des Textes erfolgt durch vorangestellte Doppelpunkte: `:` ergibt eine Einrückung um einen Schritt, `::` um zwei Schritte `:::` um drei, und so weiter.
- Unnummerierte Listen werden durch einen Asterisk (`*`), nummerierte durch ein Gitterkreuz (`#`) am Zeilenanfang markiert. Listen und Einrückungen können beliebig kombiniert und verschachtelt werden: `:*##` am Zeilenanfang markiert ein Element einer nummerierten Liste, die ein Element einer nummerierten Liste ist, die ein Element einer unnummerierten Liste ist, die eingerückt ist.
- Definitionslisten können mit `;Wort: Definition` erzeugt werden, sind aber in der Praxis eher ungebräuchlich.

A.3. Wiki-Links und Bilder

Die Wiki-Links werden als Name des Zielartikels in doppelten eckigen Klammern geschrieben: `[[Sprache]]` erzeugt einen Hyperlink auf die Seite *Sprache*. Der *Link-Text* (auch *Anker-Text* oder *Label* genannt) kann dabei optional als von dem Namen des Ziels abweichend angegeben werden, z. B. `[[Linguistik|linguistischen]]` erzeugt einen Hyperlink auf die Seite *Linguistik* mit dem sichtbaren Text „*linguistischen*“. Eine weitere Form, den Link-Text anzupassen, ist der so genannte *Link-Trail*: `[[Sprache]]n` erzeugt einen Hyperlink auf den Artikel *Sprache* mit dem sichtbaren Text „*Sprachen*“ [WP:LINKS, WP:V].

MediaWiki erlaubt (genau wie HTML) Verknüpfungen auf „Anker“ [W3C:HTML (1999)], also auf bestimmte Stellen *innerhalb* eines Artikels bzw., in URI-Terminologie, auf Dokument-Fragmente [RFC:3986 (2005)]. In MediaWiki erzeugt jede Überschrift einen Anker, es ist also möglich, Wiki-Links direkt auf einzelne Abschnitte von Artikeln zu setzen: `[[Myanmar#Geschichte]]` würde auf den Abschnitt *Geschichte* im Artikel *Myanmar* verweisen. Für Verweise auf einen Abschnitt im selben Artikel reicht die Angabe des Ankers aus: innerhalb der Seite *Myanmar* wäre also zum Beispiel der Link `[[#Geschichte]]` möglich.

Bei der Angabe des Link-Zieles kann ein Präfix verwendet werden, das durch einen Doppelpunkt von eigentlichen Namen der Zielseite getrennt ist. Ein solcher Präfix kann unterschiedliche Bedeutungen haben:

Der Präfix kann einen **Namensraum** angeben (▹1.2): Der Link *[[Hilfe:Formatieren]]* verweist auf eine Seite im Namensraum *Hilfe*, die sich mit Textformatierung befasst. Links in einige spezielle Namensräume ändern allerdings das Verhalten des Wiki-Links grundlegend: Wiki-Links, die in den Namensraum für **Bilder** zeigen, binden das entsprechende Bild in den Artikel ein, z. B.: *[[Bild:Baum.jpg]]* würde das Bild `Baum.jpg` dort einbinden, wo der Link im Text definiert wurde. Wiki-Links in den **Kategorienamensraum** ordnen die Seite einer Kategorie zu: der Link *[[Kategorie:Semantik]]* auf der Seite *Unsinn* ordnet diese Seite der Kategorie *Semantik* zu sorgt dafür, dass sie auf der Seite *Kategorie:Semantik* aufgeführt wird [WP:KAT]. Der Link würde auch nicht dort angezeigt, wo er im Text definiert ist, sondern in einem besonderen Bereich für Kategorien ganz am Ende der Seite. Die Sonderbehandlung von Wiki-Links in den Bild- und Kategorienamensraum lässt sich umgehen, indem man dem Kategorienamen einen Doppelpunkt voranstellt: *[[:Bild:Baum.jpg]]* und *[[:Kategorie:Semantik]]* erzeugt „normale“ Wiki-Links.

Bei der Kategorisierung einer Seite lässt sich ein sogenannter *Sort-Key* (*Sortierschlüssel*) angeben, der bestimmt, an welcher Stelle eine Seite in der Kategorie gelistet wird: So ist *Albert Einstein* mittels *[[Kategorie:Pazifist|Einstein, Albert]]* so der *Kategorie:Pazifist* zugeordnet, dass er unter „E“, nicht unter „A“ gelistet wird. Um einen *Sort-Key* für alle Kategorien einer Seite gemeinsam anzugeben kann das *Magic Word* *{{DEFAULTSORT:...}}* verwendet werden (siehe unten).

Der Präfix kann auf ein anderes Wiki verweisen, so dass sich ein **Interwiki-Link** ergibt. dazu wird einem Präfix eine Basis-URL zugeordnet, die zusammen mit dem angegebenen Seitentitel einen Hyperlink auf eine Seite in einem anderen Wiki ergibt. Zum Beispiel erzeugt der Link *[[mw:Installation]]* in der Wikipedia einen Hyperlink auf <`http://www.mediawiki.org/wiki/Installation`>, weil dem Präfix „MW“ das URL-Muster „`http://www.mediawiki.org/wiki/$1`“ zugeordnet wurde [WP:INTERWIKI].

Wenn der Präfix für einen Interwiki-Link gleichzeitig als Sprach-Code definiert ist, ändert sich das Verhalten des Links grundsätzlich: er wird als **Language-Link** interpretiert, der auf einen Artikel mit gleichem oder ähnlichem Inhalt in einer anderen Sprache verweist [WP:INTERLANG]. Solche Links werden nicht inline an der Stelle angezeigt, an der sie im Text definiert wurden, sondern sie werden in einem speziellen Teil der Navigationsleiste dargestellt, unter dem Namen der Sprache, die dem Sprachcode entspricht, der als Präfix des Links verwendet wurde. Diese Sprach-Codes folgen im Wesentlichen der ISO 639-1 Norm [ISO:639-1 (2002)] und entsprechen im Fall der Wikipedia den Subdomains, die für die Wikipedia-Projekte in den einzelnen Sprachen verwendet werden [M:SITES]. So würde z. B. der Link *[[en:Language]]* im Artikel *Sprache* in der deutschsprachigen Wikipedia (`de.wikipedia.org`) eine Verknüpfung auf den Artikel *Language* in der englischsprachigen Wikipedia (`en.wikipedia.org`) erzeugen, die im Artikel *Sprache* in der Navigationsleiste unter dem Label „*English*“ angezeigt wird. Auch dieser Mechanismus kann umgangen werden, indem dem Sprachcode ein Doppelpunkt vorangestellt wird: *[[:en:Language]]* funktioniert als „normaler“ Interwiki-Link, der direkt im Artikeltext angezeigt wird.

A.4. Tabellen

Tabellen beginnen mit `{|` und enden mit `|}`, jeweils in einer eigenen Zeile; hinter dem `|` am Tabellenanfang können HTML-Attribute wie `class` oder `style` angegeben werden. Zeilen der Tabelle werden mit `|-` getrennt (ebenfalls in einer eigenen Zeile, das `-` kann beliebig wiederholt werden, `|-------` ist also auch zulässig). Jede Zelle beginnt entweder mit `|` an einem Zeilenanfang, oder mit `||` irgendwo in einer Zeile. Zellen im Tabellenkopf können mit `!` am Zeilenanfang oder `!!` in der Zeile markiert werden. Ein Beispiel:

```
{| class="wikitable"
|-
! header 1
! header 2
! header 3
|-
| row 1, cell 1
| row 1, cell 2
| row 1, cell 3
|-
| row 2, cell 1
| row 2, cell 2
| row 2, cell 3
|}
```

Neben der Wiki-Syntax für Tabellen wird auch die HTML-Syntax unterstützt, mit <*table*>, <*tr*>, <*td*>, und so weiter.

A.5. Vorlagen

MediaWiki unterstützt sogenannte *Vorlagen* (*Templates*) als eine Methode, den Inhalt einer Wiki-Seite in eine andere einzubinden: Dafür wird der Name des Vorlage in doppelten geschweiften Klammern benutzt [WP:VOR]. Vorlagen befinden sich typischerweise (aber nicht notwendigerweise) im Vorlagen-Namensraum.

Beim Einbinden von Vorlagen können sogenannte *Vorlagen-Parameter* verwendet werden: Sie erlauben es, beim Einbinden konkrete Werte anzugeben, die dann für Platzhalter eingesetzt werden, die im Wiki-Text der Vorlage verwendet wurden. Zum Beispiel: Wenn auf der Seite *Vorlage:Hallo* der Wiki-Text `Hallo Welt` steht, kann dieser auf einer anderen Seite eingefügt werden, indem man dort `{{Hallo}}` schreibt. Steht in *Vorlage:Hallo* `Hallo {{{1}}}`, so würde `{Hallo|Mond}}` den Text „*Hallo Mond*“ erzeugen (positionaler Parameter); steht dort `Hallo {{{was}}}`, so würde `{{Hallo|was=Mond}}` den Text „*Hallo Mond*“ erzeugen (benannter Parameter).

A.6. Magic Words

MediaWiki definiert *Magic Words* im Wesentlichen in zwei Varianten: die eine („Variablen“) folgt der Syntax für Vorlagen und bietet Zugriff auf kontext- oder projektspezifische Werte, zum Beispiel den Titel der Seite selbst (`{{PAGENAME}}` bzw. `{{FULLPAGENAME}}`) oder die Anzahl der Seiten im Wiki (`{{NUMBEROFPAGES}}`); die andere („Optionen“) beeinflusst die Verarbeitung der Seite, zum Beispiel schaltet `__NOTOC__` das automatisch erzeugte Inhaltsverzeichnis aus [MW:MAGIC].

Die meisten *Magic Words* werden von WikiWord ignoriert. Einige, die von WikiWord verwendet werden, sind die folgenden:

`#REDIRECT` definiert eine Weiterleitungsseite [WP:WL]. Beginnt die erste nicht-leere Zeile der Seite mit dieser Markierung, so ist die Seite eine Weiterleitungsseite und dient als Alias für eine andere Seite (siehe ▹1.2); Das Ziel des Alias wird als Wiki-Link unmittelbar hinter dem *Magic Word* angegeben; Zum Beispiel könnte die Seite *USA* den Text `#REDIRECT [[Vereinigte Staaten von Amerika]]` enthalten und so als Weiterleitung auf *Vereinigte_Staaten_von_Amerika* dienen.

Neben `#REDIRECT` wird auch `#REDIRECTION` unterstützt, sowie in einigen Wikis auch eine übersetzte Version dieses *Magic Words*, zum Beispiel `#DOORVERWIJZING` in der niederländischsprachigen Wikipedia. Solche Übersetzungen sind in der entsprechenden Implementation von `WikiConfiguration` berücksichtigt (▹8).

`{{DISPLAYTITLE:ldots}}` definiert einen alternative Form des Titels zur Anzeige. MediaWiki lässt solche alternativen Titel nur sehr eingeschränkt zu, zum Beispiel um Titel mit einem Kleinbuchstaben am Anfang zu erlauben, wie zum Beispiel „*iPod*“[1] für die Seite *IPod* oder „*c't*“ für die Seite *C't*. Dies ist notwendig, da MediaWiki für den ersten Buchstaben des Seitennamens Großschreibung erzwingt. Auch dieses *Magic Word* kann lokalisiert werden, zum Beispiel als `{{AFFICHERTITRE:ldots}}` im Französischen.

`{{DEFAULTSORT:...}}` gibt an, welcher *Sort-Key* bei der Einordnung der Seite in Kategorien verwendet werden soll, wenn im Kategorisierungslink kein *Sort-Key* angegeben ist. Alternative Formen sind `{{DEFAULTSORTKEY:...}}` und `{{DEFAULTCATEGORYSORT:...}}`, Lokalisierungen sind zum Beispiel `{{STANDAARDSORTERING:...}}` im Niederländischen und `{{CLEFDETRI:...}}` im Französischen.

Zur Nutzung dieser *Magic Words* siehe ▹8.9.

[1] iPod ist ein eingetragenes Markenzeichen der Apple Computer, Inc.

A.7. Parser-Functions und Extension-Tags

MediaWiki bietet Schnittstellen, über die das Wiki-Text Markup erweitert werden kann. Es gibt im Wesentlichen zwei Arten von solchen Erweiterungen: *Extension-Tags* definieren XML-artige Tag-Strukturen der Form `<mytag>...</mytag>`, *Parser-Functions* definieren Vorlagen-artige Strukturen der Form `{{#mytag:argument}}`.

Solche speziellen Syntax-Erweiterungen werden in der Regel von Erweiterungen verwendet, die dynamische Inhalte zum Beispiel aus einer Datenbank in einem Wiki-Artikel anzeigen.

B. Kommandozeilenschnittstelle

B.1. Anwenderschnittstelle

WikiWord wird über eine klassische Kommandozeilenschnittstelle (*Command Line Interface, CLI*) bedient. Zur Extraktion eines Thesaurus aus Wikipedia-Dumps werden insbesondere die folgenden Programme (*Entry Points*) verwendet.

`ImportConcepts` liest Seiten aus einem Wikipedia-Dump, analysiert den Wiki-Text (▹8), überführt ihn zunächst in das Ressourcenmodell und dann weiter in einen lokalen Datensatz (▹9.1). Dieser Schritt ist der erste, der auf dem Weg zu einem Thesaurus für jeden Wikipedia-Dump ausgeführt werden muss.

`BuildThesaurus` führt die einzelnen lokalen Datensätze zusammen und erzeugt damit einen globalen Datensatz (▹9.2). Dieser Schritt setzt natürlich voraus, dass die entsprechenden Dumps bereits importiert wurden.

`BuildStatistics` berechnet Statistiken auf einem lokalen oder globalen Datensatz (▹9.3). Dabei werden vor allem die Knotengrade im Term-Konzept-Graphen auf ihre Verteilung hin untersucht. Das setzt voraus, dass die betreffenden lokalen Daten bereits importiert wurden bzw. der globale Datensatz bereits erstellt wurde.

`BuildConceptInfo` fasst die Eigenschaften und Relationen jedes Konzeptes in einem lokalen oder globalen Datensatz für einen effizienten Zugriff zusammen (▹9.4). Das setzt voraus, dass die Statistiken für den betreffenden Datensatz bereits erstellt wurden.

`ExportRdf` Erzeugt eine Repräsentation des Thesaurus in RDF/SKOS (▹10.3). Dieser Schritt setzt voraus, dass die betreffenden lokalen Daten bereits importiert wurden bzw. der globale Datensatz bereits erstellt wurde.

Die Parameter und Optionen für den Aufruf der einzelnen Programme werden im Anhang beschrieben (▹B.2). Weitere Möglichkeiten zur Konfiguration bieten die sogenannten *Tweak-Werte*, siehe ▹B.5.

Die Durchführung aller Aufgaben wird über das *Agenda*-System kontrolliert und protokolliert (siehe Anhang B.6). Dadurch ist es einerseits möglich, den genauen Zeitaufwand einzelner Teilaufgaben zu ermitteln sowie andererseits unterbrochene Berechnungen wieder aufzunehmen. Das ist insbesondere deshalb wichtig, weil die Verarbeitung großer Datenmengen, wie zum Beispiel der englischsprachigen Wikipedia, mitunter mehrere Tage dauern kann.

Für das Erstellen eines Thesaurus werden die Programme `BuildStatistics` und `BuildConceptInfo` nicht benötigt — es ist ausreichend, die Wiki-Dumps mit `ImportConcepts` zu importieren, mit `BuildThesaurus` zusammenzufassen und dann mit `ExportRdf` zu exportieren. Die von `BuildStatistics` generierten Statistiken können aber bei der weiteren

Verarbeitung des Thesaurus nützlich sein, zum Beispiel für die Gewichtung bzw. das Ranking von Konzepten. Die Daten, die `BuildConceptInfo` zur Verfügung stellt, werden immer dann benötigt, wenn nicht der Thesaurus als Ganzes, sondern einzelne Konzepte (oder kleine Mengen von Konzepten) verarbeitet werden sollen — insbesondere also bei der Evaluation (▹12).

B.2. Parameter und Optionen

Die möglichen Parameter und Optionen für den Aufruf von WikiWord-Programmen sind:

Kollektion: Name der Kollektion von Datensätzen. Dieser Name wird zur Bestimmung des Präfixes für die zu verwendenden Datenbanktabellen benutzt (siehe ▹C). Wird typischerweise als erstes Glied in einem Paar von *Kollektion*:*Datensatzname* angegeben.

Datensatzname: Name des zu bearbeitenden (lokalen oder globalen) Datensatzes. Dieser Name wird zur Bestimmung des Präfixes für die zu verwendenden Datenbanktabellen benutzt (siehe ▹C). Wird typischerweise als zweites Glied in einem Paar von *Kollektion*:*Datensatzname* angegeben.

Wikiname: Ein *Datensatzname* eines lokalen Datensatzes — in der Regel wird hier ein Sprach-Code nach ISO 639-1 erwartet [ISO:639-1 (2002)].

Thesaurusname: Ein *Datensatzname* eines globalen Datensatzes.

Datenbank-Konfigurationsdatei: eine Datei, die die Zugangsdaten für die Datenbankverbindung enthält (siehe ▹B.4).

Dump-Datei: ein Wikipedia XML-Dump, wie in [MW:XML] beschrieben und auf [WM:DOWNLOAD] verfügbar. Die angegebene Datei wird mit der Klasse `DumpImportDriver` verarbeitet (▹7.1), die ihrerseits die `mwdumper`-Bibliothek von Wikimedia verwendet [MW:DUMPER].

RDF-Datei: ein RDF-Datei für die Ausgabe des Thesaurus ▹10.

Sprachen: eine Liste von Sprach-Codes, separiert durch Kommata („,“).

Aus dem Dateinamen wird automatisch auf den Wiki-Namen geschlossen, und damit auf den zu verwendenden Tabellenpräfix (siehe ▹C): Der Teil des Namens vor dem ersten „-“ und vor dem ersten „.“ wird als Wiki-Name angenommen. Der automatisch erkannte Name kann mit der Option `--dataset` überschrieben werden.

Die angegebene XML-Datei kann mit GZip oder BZip2 komprimiert sein (dann muss der Dateiname auf „`.gz`“ bzw. „`.bz2`“ enden). Beginnt der Dateiname mit „`http://`“ so wird der Dump direkt per HTTP geladen und verarbeitet — das ist vor allem im Testbetrieb nützlich und sollte nicht für große Datenmengen verwendet werden.

`--loglevel` *L*: bestimmt den Log-Level *L*, also die Menge und Granularität von angezeigten Nachrichten. Der Wert muss eine positive ganze Zahl sein; Nachrichten, deren Level größer

oder gleich dem hier angegebenen Level ist, werden angezeigt. Vordefinierte Werte (übernommen aus der Standardklasse `java.util.logging.Level`) sind 0 (All), 400 (Trace), 500 (Debug), 800 (Info), 900 (Warnung) und 1000 (Fehler). Der Standard-Wert ist 800.

`--db` *F*: verweist auf die Datei *F*, die die Spezifikation der Datenbankverbindung enthält (▹B.4). Falls nicht angegeben, wird die Datei an vier Orten gesucht: `/etc/wikiword/db.properties`, `/etc/wikiword.db.properties`, *HOME*`/.wikiword/db.properties` und *HOME*`/.wikiword.db.properties`. Wenn keine solche Datei gefunden werden konnte, kann keine Datenbankverbindung aufgebaut werden.

`--tweaks` *F*: verweist auf eine Datei *F*, aus der zusätzliche Konfigurationsparameter (*Tweaks*) geladen werden sollen (siehe ▹B.5). Falls nicht angegeben, wird die Datei an vier Orten gesucht: `/etc/wikiword/tweaks.properties`, `/etc/wikiword.tweaks.properties`, *HOME*`/.wikiword/tweaks.properties` und *HOME*`/.wikiword.tweaks.properties`. Wenn keine solche Datei gefunden werden konnte, werden Default-Werte verwendet.

`--tweak.`*NNN*`=`*VVV*: setzt die Tweak-Option *NNN* auf den Wert *VVV*. So definierte Optionen überschreiben solche, die aus einer mit `--tweaks` angegebenen Datei geladen wurden.

`--fresh, --continue`: erzwingen den Neustart bzw. das Fortsetzen eines unterbrochenen Vorgangs (▹B.6). Das heißt, sie unterdrücken die Rückfrage, die bei Aufruf des Programms gestellt wird, wenn erkannt wurde, das der vorherige Aufruf nicht korrekt beendet wurde.

B.3. Aufruf der einzelnen Programme

`ImportConcepts` liest Seiten aus einem Wikipedia-Dump, analysiert den Wiki-Text (▹4.1), überführt ihn zunächst in das Ressourcenmodell und dann weiter in einen lokalen Datensatz (▹4.2). Aufruf:

```
> java ImportConcepts [Optionen] <Kollektion:Wikiname> <Dump-Datei>
```

`BuildThesaurus` führt die einzelnen lokalen Datensätze zusammen und erzeugt einen globalen Datensatz (▹4.3). Aufruf:

```
> java BuildThesaurus [Optionen] [--languages Sprachen] <Kollektion:Thesaurusname>
```

`BuildStatistics` berechnet Statistiken auf einem lokalen oder globalen Datensatz (▹9.3). Aufruf:

```
> java BuildStatistics [Optionen] <Kollektion:Datensatzname>
```

`BuildConceptInfo` berechnet zusammengefasste Datensätze für einen effizienten Zugriff auf einen lokalen oder globalen Datensatz (▹9.4). Aufruf:

```
> java BuildStatistics [Optionen] <Kollektion:Datensatzname>
```

ExportRdf Erzeugt (▸10) eine Repräsentation des Thesaurus in SKOS (siehe ▸10). Aufruf:

```
> java ExportRdf [Optionen] <Kollektion:Datensatzname> <RDF-Datei>
```

Dabei unterstützt `ExportRdf` die zusätzliche Option `--skos`, die bewirkt, dass möglichst „reines“ SKOS ausgegeben und auf das WikiWord-Vokabular verzichtet wird. Das kann die Kompatibilität mit bestehenden Systemen verbessern, führt aber dazu, dass weniger Informationen in den Export übernommen werden. Die Option `--assoc` dagegen aktiviert die Ausgabe aller „losen Assoziationen“ zwischen Konzepten, also der `ww:assoc`-Relation, die die Hyperlinks zwischen Wiki-Seiten wiedergibt. Für weitere Parameter siehe ▸10.3.

Um den Aufruf der WikiWord-Programme zu erleichtern, insbesondere in Hinblick auf das Auflisten aller benötigten Bibliotheksdateien (*JAR-Dateien*) in Java-Classpath, steht das *Shell-Skript* `ww.sh` zur Verfügung. Es wird wie folgt aufgerufen:

```
> ww.sh <Klassenname> [<parameter>]
```

Dabei kann der Präfix `de.brightbyte.wikiword.` am Klassennamen weggelassen werden. Das folgende Kommando führt also die Klasse `de.brightbyte.wikiword.builder.ImportConcepts` aus und importiert die Artikel aus der Datei `enwiki-dump.xml` in den Datensatz `full:en`

```
> ww.sh builder.ImportConcepts full:en enwiki-dump.xml
```

Das Verhalten von `ww.sh` kann über Konfigurationsdateien angepasst werden. Die folgenden Arten von Konfigurationsvariablen sind möglich:

- **Optionen für die Java-VM** können in den folgenden Dateien angegeben werden: `/etc/wikiword/vm.options`, `/etc/wikiword.vm.options`, *HOME*`/.wikiword/vm.options` und *HOME*`/vm.options`. Typischerweise wird hier die zu verwendende Menge an Arbeitsspeicher definiert, indem die Java-VM Option `-Xmx` gesetzt wird, zum Beispiel mit `-Xmx8G` auf acht Gigabyte, um ausreichend Speicherplatz für eine Verwaltung aller Konzept-IDs zu haben (vergleiche ▸9.1.3).
- **Optionen für WikiWord** können in den folgenden Dateien angegeben werden: `/etc/wikiword/vm.options`, `/etc/wikiword.vm.options`, *HOME*`/.wikiword/vm.options` und *HOME*`/vm.options`.
- Der **Java Classpath** kann in den folgenden Dateien angegeben werden: `/etc/wikiword/classpath`, `/etc/wikiword.classpath`, *HOME*`/.wikiword/classpath` und *HOME*`/classpath`. Dabei ist darauf zu achten, dass der Pfad in einer einzigen Zeile angegeben wird, keine Leerzeichen enthält und die einzelnen Elemente durch „`:`“ getrennt sind.

Alle Konfigurationsdateien können selbst auch *Shell-Skripte* sein: wenn an den Dateinamen die Endung `.sh` angehängt und die Datei als ausführbar markiert ist, ruft `ww.sh` sie als Skript auf und betrachtet den ausgegebenen Text als die zu verwendende Konfiguration. So könnte zum Beispiel *HOME*`/.wikiword/classpath.sh` ein Skript sein, dass programmatisch einen geeigneten Classpath erzeugt, indem es alle JAR-Dateien in einem bestimmten Verzeichnis auflistet (in einer Zeile, durch „`:`“ getrennt).

B.4. Datenbank-Konfiguration

Die Konfigurationsdatei für den Datenbankzugriff folgt dem Java-Standard für Property-Dateien, wie von der Standard-Java-Klasse `java.util.Properties` definiert. Sie enthält die folgenden Schlüssel-Wert-Paare:

`driver` ist der Klassenname des zu verwendenden JDBC-Treibers; Für WikiWord sollte das ein Treiber für MySQL sein, entwickelt und getestet wurde WikiWord mit *jconnector* (Version 3.1), anzugeben als `com.mysql.jdbc.Driver`.

`url` ist eine JDBC-URL [JDBC], die die zu verwendende Datenbank angibt. Das Format der URL muss zu der angegebenen Treiberklasse passen — für *jconnector* folgt sie der Form `mysql://`*`server/datenbank`*`?characterEncoding=utf8`, wobei der zu verwendende Zeichensatz immer explizit als „`utf8`“ angegeben werden sollte.

`user` ist der Benutzername für die Datenbankverbindung

`password` ist das Passwort für die Datenbankverbindung

Diese Datei muss entweder im Benutzerverzeichnis unter dem Namen `.wikiword.db.properties` abgelegt oder über die Option `--db` explizit angegeben werden.

B.5. Konfiguration (Tweaks)

WikiWord unterstützt diverse Parameter zur Feineinstellung des Import-Prozesses, sogenannte „*Tweaks*“; diese dienen vor allem der Anpassung an eine konkrete Systemumgebung (verfügbarer Arbeitsspeicher, Anzahl von Prozessoren, etc). Eine Beispielkonfiguration mit Erläuterungen zu den einzelnen Konfigurationsparametern ist unter `/WikiWord/WikiWordBuilder/tweaks.properties` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/WikiWordBuilder/tweaks.properties`> verfügbar.

Werden die Tweak-Optionen in einer Konfigurationsdatei gespeichert, so dass sie über den Parameter `--tweaks` geladen werden können, so gilt für diese Datei die Syntax für Java Property-Dateien, wie von der Standard-Klasse `java.util.Properties` definiert, mit einer speziellen Syntax für die Werte: diese müssen die Syntax für Java-Literale des entsprechenden Typs verwenden. Insbesondere müssen String-Werte in Anführungszeichen stehen.

B.6. Ablaufsteuerung (Agenda)

WikiWord unterteilt die Ausführung in rekursiv geschachtelte Schritte, die mittels der Klasse `de.brightbyte.application.Agenda` einzeln protokolliert werden. Dieses Protokoll kann dazu

verwendet werden, einen Import-Prozess, der unterbrochen wurde, später an der fraglichen Stelle fortzusetzen.

Neben dem Vorzug, langwierige Importprozesse nicht komplett neu starten zu müssen, falls sie unerwartet unterbrochen wurden, erlaubt dieses System auch eine detaillierte Protokollierung darüber, welche Schritte wieviel Zeit in Anspruch nehmen.

B.7. Anpassung an einzelne Wikis

WikiWord arbeitet zwar weitgehend Sprachunabhängig, aber es benötigt doch Informationen über die Konfiguration der einzelnen Wikis sowie über Konventionen und Muster, die dort verwendet werden. Diese Wiki-spezifischen Konfigurationen werden im Package `de.brightbyte.wikiword.wikis` definiert (vergleiche ▹8). Insbesondere können dort die folgenden Klassen bzw. Dateien angelegt werden:

`LanguageConfiguration_\textit{x}` dient der Angabe sprachspezifischer Eigenschaften für die Sprache *x*. Hier wird insbesondere das Muster definiert, das dazu dient, gebräuchliche Abkürzungen zu erkennen. Diese Information wird für die Erkennung von Satzenden benötigt, siehe ▹E.2.

`WikiConfiguration_\textit{x}wiki` dient der Angabe projektspezifischer Eigenschaften für `\textit{x}.wikipedia.org`. Die meisten Muster und Sensoren, die zur Analyse des Wiki-Textes (▹8) verwendet werden, werden hier definiert. In ▹8.2 werden die Objekttypen beschrieben, die für die Konfiguration zur Verfügung stehen.

`Namespace_\textit{x}wiki.properties` ist eine Java Property-Datei, die die Namen für die Namensräume in der Sprache *x* angibt. Die lokalisierten Namen der Namensräume sind wichtig, um zum Beispiel Wiki-Links auf Bilder identifizieren zu können (▹A.3). Allerdings werden die Namen der Namensräume auch in jedem Wiki-Dump mit angegeben.

Über diese Klassen bzw. Dateien lässt sich WikiWord an verschiedene Sprachen und Wiki-Projekte anpassen.

C. Datenbank

Zur Speicherung der Datensätze verwendet WikiWord ein relationales Datenbanksystem, konkret MySQL [MYSQL]. Das interne Datenmodell ist daher in Form von Entitäten und ihren Beziehungen definiert, die sich in Datenbanktabellen und -feldern manifestieren. Diese Datenbankstruktur von lokalen und globalen Datensätzen ist in den Abschnitten ▹C.1 und ▹C.2 beschrieben, die Abschnitte ▹C.3 und ▹C.4 befassen sich mit der *Datenzugriffsschicht* (*DAO-Layer*), die die Einzelheiten der Kommunikation mit der Datenbank kapselt.

Datensätze haben in WikiWord Namen und werden in Kollektionen zusammengefasst (▹1.1). Der Name muss innerhalb einer Kollektion eindeutig sein, die Kombination aus dem Namen der Kollektion und dem Namen des Datensatzes (der „*qualifizierte Datensatzname*“) muss innerhalb einer Datenbank einmalig sein — idealerweise dient er als eindeutiger Bezeichner innerhalb der Organisation, die den Thesaurus erstellt (vergleiche ▹10.1). Es gibt zwei Arten von Datensätzen: einerseits *lokale*, die aus einem bestimmten Korpus (also einer bestimmten Wikipedia) extrahiert wurden (siehe ▹9.1) — sie tragen in der Regel den Sprachcode als Namen, der der Sprache des Korpus entspricht; und andererseits *globale*, die äquivalente Konzepte aus den verschiedenen lokalen Datensätzen zu sprachunabhängigen Konzepten verbinden (siehe ▹9.2). Im Allgemeinen gibt es in jeder Kollektion nur einen globalen Datensatz und dieser bezieht sich auf alle lokalen Datensätze der Kollektion.

MySQL betrachtet eine „Datenbank“ als eine Menge von Tabellen mit Namen, die innerhalb der Datenbank eindeutig sind, eine Aufteilung der Datenbank in „Schemas“ wird nicht unterstützt (der Begriff „Schema“ wird im Folgenden als Synonym für „Datenbankstruktur“ verwendet, nicht als Bezeichnung für eine organisatorische Unterteilung von Datenbanken). WikiWord kann alle Datensätze aus allen Kollektionen in einer einzigen Datenbank verwalten, indem dem Namen jeder Datenbanktabelle ein *Präfix* vorangestellt wird, der aus dem Namen der Kollektion und dem Namen des Datensatzes erzeugt wird. Zum Beispiel ergibt sich der Präfix „`full2008_de_`“ für den Datensatz `de` in der Kollektion `full2008`, die Tabelle `concept` für diesen Datensatz würde also den Namen `full2008_de_concept` tragen.

C.1. Datenbankstruktur

Das relationale Datenbank-Schema, das WikiWord benutzt, um die aus der Wikipedia extrahierten Daten zu speichern und zu verarbeiten, repräsentiert Konzepte und Terme sowie Relationen zwischen Konzepten und zwischen Konzepten und Termen, das heißt, es implementiert das *Konzeptmodell*. Zusätzlich zu den primären Strukturen, aus denen der Thesaurus besteht, werden Strukturen für Sekundärdaten und temporäre Daten benötigt.

Zu beachten ist, wie sich die *lokale* von der *globalen* Ausprägung des Datenbank-Schemas unterscheidet: das lokale Schema beschreibt Datensätze, die direkt aus *einer* einzelnen Wikipedia extrahiert wurden (▹4.2), also Daten in einer bestimmten Sprache. Das globale Schema beschreibt den Datensatz, der sich aus dem Zusammenführen der einzelnen lokalen Datensätze ergibt (▹4.3).

An den Tabellen und Feldern im Abschnitt ▹C.2 ist jeweils angegeben, ob sie nur im globalen oder nur im lokalen Schema verwendet werden oder in beiden.

Die Datenbankstrukturen, die WikiWord zur Speicherung der Datensätze verwendet, sind nicht vollständig *normalisiert* — zwar sind die Entitäten und Relationen in einzelnen Tabellen repräsentiert und durch Referenzen verbunden, die Tabellen enthalten aber an einigen Stellen redundante Informationen. Insbesondere sind die Referenzen auf Konzepte an vielen Stellen doppelt realisiert: einmal über numerische ID des Konzeptes und einmal über den Namen. Diese gezielte *Denormalisierung* des Datenbankmodells hat folgende Vorteile:

- Namensbasierte Referenzen erlauben einen schnellen Aufbau des Datenbestandes dadurch, dass Beziehungen zwischen Konzepten gespeichert werden können, wenn nur ihr Name, aber (noch) nicht ihre numerische ID bekannt ist. In diesem Fall würde in der Relation nur ein Verweis auf den Namen des Konzeptes abgelegt, das Feld, das die ID referenziert, bliebe zunächst `NULL` und würde erst in einem Nachbereitungsschritt gesetzt (▹9.1.3). Zwar könnten die Relationen auch gänzlich auf der Basis von Namen arbeiten, so dass keine IDs verwaltet werden müssen — das bedeutet allerdings eine deutlich schlechtere Performanz bei Datenbankoperationen, insbesondere wenn eine große Zahl von Verknüpfungen auf einmal verarbeitet werden muss, wie das bei einem *Join* (*Verbund*) der Fall ist.

  Namensbasierte Referenzen vermeiden also den Aufwand, bei jeder Referenz die ID des entsprechenden Konzeptes durch eine Datenbankabfrage bestimmen und gegebenenfalls einen Eintrag für das Konzept anlegen zu müssen. Andererseits müssen die IDs am Ende in einem zusätzlichen Arbeitsschritt nachgeführt werden. Eine Alternative ist, die Abbildung von Konzeptnamen auf numerische IDs vollständig im Arbeitsspeicher zu verwalten — was allerdings, abhängig von der Anzahl zu speichernder Konzepte, eine große Menge RAM erfordert (siehe ▹9.1.3).

- Ein weiterer Vorteil, Referenzen doppelt für Namen und IDs zu führen, ist, dass auf diese Weise häufig ein *Join* auf die Tabelle `concept` gänzlich vermieden werden kann — nämlich dann, wenn der Konzepteintrag nur benötigt wird, um den menschenlesbaren Namen des Konzeptes zu bestimmen. Einen *Join* zu vermeiden bedeutet einerseits bessere Performanz und andererseits einfachere Abfragen, die leichter zu warten und weniger fehleranfällig sind.

- Auch bei der Fehlersuche und -analyse ist es hilfreich, Beziehungen direkt in einer „lesbaren“ Form, also zwischen Name und Name, betrachten zu können. Das vermeidet zusätzlichen Aufwand und neue Fehlerquellen bei der manuellen Navigation der Datenbank.

Die Verwendung des Namens eines Konzeptes zur Identifikation ist allerdings auf lokale Datensätze beschränkt: für globale Datensätze sind die Namen weniger aussagekräftig und auch zunehmend „unhandlich“, da sie aus der Konkatenation aller Namen von lokalen Konzepten bestehen, die das globale Konzept kombiniert. Auch ist der Nutzen der doppelten Referenzen hier geringer, weil kein schrittweise Aufbau der Daten stattfindet und somit die numerischen IDs aller Konzepte im Vorhinein bekannt sind.

C.2. Datenbanktabellen

Dieser Abschnitt erläutert die Struktur und Funktion der einzelnen Datenbanktabellen.

`concept`

Die Tabelle `concept` repräsentiert die einzelnen Konzepte des Thesaurus. Die Konzepte bilden das Herzstück der Datenstruktur, sie sind die Entitäten, auf die sich sämtliche Relationen beziehen.

Diese Tabelle existiert sowohl in lokalen als auch in globalen Datensätzen. Sie enthält die folgenden Felder:

`id`: Interne ID, automatisch vergeben. Eindeutig innerhalb eines Datensatzes.

`random`: Zufallswert zwischen 0 und 1, um effizient zufällige Konzepte wählen zu können.

`name`: Name des Konzeptes, hergeleitet von dem Namen der betreffenden Wikipedia-Seite. Der Name ist eindeutig innerhalb eines Datensatzes. Er hat *keine* Bedeutung im Thesaurus, kann aber zur Anzeige verwendet werden. In einem lokalen Datensatz ist dieses Feld ein *Schlüssel* der Tabelle `concept` und erlaubt einen effizienten Zugriff auf einzelne Einträge (vergleiche ▹C.1) – für globale Datensätze ist das *nicht* der Fall, dort ist der Name rein informativ.

`type`: Typ des Konzeptes, wie von `ConceptType` definiert. Siehe auch ▹8.6.

`resource` (nur lokal): Referenz auf den Eintrag in der Tabelle `resource` für diejenige Wiki-Seite, die dieses Konzept beschreibt. Die Ressource ist der Ort der autoritativen Definition des Konzeptes.

`language_bits` (nur global): Oder-Verknüpfung der Sprach-Bits der lokalen Konzepte, die das globale Konzept zusammenfasst (▹9.2). Die entsprechenden lokalen Konzepte sind über die Tabelle `origin` verknüpft (▹C.2).

`language_count` (nur global): Anzahl der lokalen Konzepte, die das globale Konzept zusammenfasst. Erlaubt den Aufbau einer Statistik darüber, wie stark lokale Konzepte zusammengefasst werden konnten.

`resource`

Zusätzlich zu den Konzepten werden im lokalen Datensatz auch Einträge für *Ressourcen*, also Wiki-Seiten, angelegt und an verschiedenen Stellen referenziert. So wird für die Konzepte und viele der Relationen angegeben, aus welcher konkreten Seite sie extrahiert wurden (▹9.1). Das macht die Herkunft einzelner Beziehungen nachvollziehbar und hilft so bei der Fehlersuche. Außerdem ist diese Information Voraussetzung für selektive Updates der Informationen aus einzelnen Ressourcen.

Die Tabelle `resource` enthält Einträge für all die Ressourcen, aus denen Informationen extrahiert wurden. Diese Einträge beziehen sich auf einen konkreten Korpus, daher existiert diese Tabelle nur für lokale Datensätze.

`id`: Interne ID, automatisch vergeben. Eindeutig innerhalb eines Datensatzes.

`name`: Name der Ressource, eindeutig innerhalb eines Datensatzes. Entspricht dem Namen der Wiki-Seite.

`type`: Code für den Typ der Ressource, wie von `ResourceType` definiert. Wichtige Typen sind `ARTICLE`, `REDIRECT`, `DISAMBIGUATION` und `CATEGORY`. Siehe auch ▹8.5.

`timestamp`: Zeitstempel der letzten Änderung an der Ressource. Wird direkt aus der Datenquelle (also dem Dump) übernommen und kann für selektive Updates verwendet werden.

`definition`

Die Tabelle `definition` speichert die Definitionssätze (Glossen) für die einzelnen Konzepte, wie sie aus den Wikipedia-Artikeln extrahiert wurden (▹8.8). Definitionen sind natürlich an eine bestimmte Sprache und daher an einen konkreten Korpus gebunden, daher gibt es diese Tabelle nur für lokale Datensätze.

`concept`: Verweis auf den Eintrag in der Tabelle `concept`, für den die Definition gilt.

`definition`: Der Text der Definition. Dieser Text sollte vollständig von Wiki-Markup bereinigt sein.

`broader`

Die Tabelle `broader` repräsentiert die Subsumtionsrelation zwischen Konzepten: sie verbindet also allgemeinere Konzepte mit ihnen untergeordneten, konkreteren, wobei jedes Konzept beliebig viele über- wie auch untergeordnete Konzepte haben kann. Das entspricht der klassischen *Hyperonym*-Beziehung (die NT/BT-Relation nach [ISO:2788 (1986)]), bezieht sich aber auf Konzepte und nicht auf Terme. Die Tabelle enthält die folgenden Felder:

`narrow` (lokal und global) und `narrow_name` (nur lokal): Verweise auf den Eintrag in der Tabelle `concept`, der das konkretere Konzept enthält. `narrow` ist ein Verweis auf das Feld `id`, `narrow_name` auf das Feld `name`. Während des Imports kann `narrow` zunächst NULL sein und würde dann später nachgeführt (siehe ▹C.1).

`broad` (lokal und global) und `broad_name` (nur lokal): Verweise auf den Eintrag in der Tabelle `concept`, der das allgemeinere Konzept enthält. `broad` ist ein Verweis auf das Feld `id`, `broad_name` auf das Feld `name`. Während des Imports kann `broad` zunächst NULL sein und würde dann später nachgeführt.

resource (nur lokal): Verweis auf den Eintrag in der Tabelle `resource` für die Wiki-Seite, aus der diese Verknüpfung extrahiert wurde.

rule (nur lokal): ID der Regel, die verwendet wurde, um diese Verknüpfung zu extrahieren (siehe ▹9.1). Das erlaubt eine Evaluation der Verwendung und Akkuratheit einzelner Regeln (Heuristiken).

langlink

Die Tabelle `langlink` enthält die Language-Links (siehe ▹1.2), die in den einzelnen Wikipedia-Seiten definiert wurden. Sie repräsentieren eine *semantische Ähnlichkeit* oder gar *Äquivalenz* zwischen Konzepten in Datensätzen verschiedener Sprachen. Die Information, die in dieser Verknüpfung enthalten ist, ist essentiell für das Zusammenführen von lokalen Konzepten zu sprachunabhängigen (siehe ▹9.2 und insbesondere ▹9.2.2).

concept und concept_name: Verweise auf den Eintrag in der Tabelle `concept`, von dem die Verknüpfung ausgeht. `concept` ist ein Verweis auf das Feld `id`, `concept_name` auf das Feld `name`.

language: Der Name des lokalen Datensatzes, auf den verwiesen wird (also der Sprachcode der betreffenden Wikipedia). Der betreffende Datensatz muss nicht unbedingt in der vorliegenden Kollektion enthalten sein.

target: Der Name des ähnlichen oder äquivalenten Konzeptes in dem anderen Datensatz (also der Name der Seite in der entsprechenden Wikipedia).

resource: Verweis auf den Eintrag in der Tabelle `resource` für die Wiki-Seite, aus der diese Verknüpfung extrahiert wurde.

link

Die Tabelle `link` repräsentiert Referenzen zwischen Konzepten, wie sie sich aus Hypertext-Verknüpfungen zwischen den Seiten, die die Konzepte beschreiben, ergeben. Sie speichert außerdem den für die Verknüpfung verwendeten Link-Text (*Label*), und stellt somit eine Verbindung zwischen solchen Link-Texten als Termen und Konzepten (nämlich den Zielen des Verweises) her (▹8.7 und ▹9.1). Die Tabelle `link` wird auch dann zum Speichern für solche Zuordnungen von Termen und Konzepten verwendet, wenn diese nicht aus einem Hyperlink resultieren — in diesem Fall gibt es kein Quell-Konzept, das heißt, `anchor` und `anchor_name` sind `NULL`.

anchor (lokal und global) und anchor_name (nur lokal): Verweise auf den Eintrag in der Tabelle `concept`, von dem die Verknüpfung ausgeht. `anchor` ist ein Verweis auf das Feld `id`, `anchor_name` auf das Feld `name`. In Fällen einer bloßen Zuordnung eines Terms zu einem Konzept sind diese Felder NULL.

`target` (lokal und global) und `target_name` (nur lokal): Verweise auf den Eintrag in der Tabelle `concept`, auf den die Verknüpfung zielt. `target` ist ein Verweis auf das Feld `id`, `target_name` auf das Feld `name`. Diese Felder bestimmen also, welches Konzept mit dem in `term_text` gespeicherten Term gemeint ist.

`resource` (nur lokal): Verweis auf den Eintrag in der Tabelle `resource` für die Wiki-Seite, aus der diese Verknüpfung extrahiert wurde.

`term_text`: Term, der dem in `target` referenzierten Konzept zugeordnet wird, zum Beispiel weil er als Link-Text (*Label*) eines Hyperlinks auf dieses Konzept vorkommt.

`rule` (nur lokal): ID der Regel, die verwendet wurde, um diese Verknüpfung zu extrahieren (siehe ▹9.1). Das erlaubt einerseits eine Evaluation der Verwendung und Akkuratheit einzelner Regeln (Heuristiken), andererseits dient es auch der Bestimmung der Verlässlichkeit von Termzuweisungen selbst: Die Werte im Feld `rule` in der Tabelle `meaning`, die für das Cutoff beim Export des Thesaurus verwendet wird (▹10.4), basieren auf den Werten dieses Feldes.

`section`

Die Tabelle `section` ist eine temporäre Ablage für Artikel*abschnitte*, die als Ziel eines Hyperlinks dienen (siehe ▹9.1.1). Für solche Abschnitte wird später jeweils ein eigener Konzepteintrag erstellt, der dann dem Konzept des Artikels, der den Abschnitt enthält, untergeordnet wird (siehe ▹9.1.3). Wird nur in lokalen Datensätzen verwendet.

`resource`: Verweis auf den Eintrag in der Tabelle `resource` für die Wiki-Seite, auf der sich die Verknüpfung befindet, die auf den Abschnitt verweist.

`section_name`: Der volle Name des Abschnitts (in der Form „*Seite#Abschnitt*“), eindeutig innerhalb eines Datensatzes.

`concept_name`: Der Name des Konzeptes, den die Seite, in der der Abschnitt liegt, beschreibt.

`alias`

Die Tabelle `section` speichert alle Aliase, also alle sekundären Namen von Konzepten, wie sie sich z. B. aus *Weiterleitungsseiten* (▹A.6) in der Wikipedia oder anderen Quellen ergeben (zum Beispiel der Zuordnung von *Hauptartikeln* zu Kategorien, ▹9.1.2). Diese Tabelle dient vor allem dazu, später alle Vorkommen des sekundären Namens durch eine Referenz auf das eigentliche Konzept zu ersetzen. Die Ersetzung kann sich auf Verknüpfungen über die `link` und `langlink` Tabellen beziehen (siehe dazu ▹9.1.3 bzw. ▹9.2.1) oder auf Über- bzw. Unterordnung über die Tabelle `broader` (siehe dazu wieder ▹9.1.3). Die Tabelle `alias` wird nur in lokalen Datensätzen verwendet.

scope: Code für die Art von Beziehungen, auf die sich der Alias bezieht (wie in `AliasScope` definiert). Ein Alias gilt entweder für alle Beziehungen oder nur für die Subsumtionsrelation (also die Tabelle `broader`).

source und source_name: Verweise auf den Eintrag in der Tabelle `concept`, der zu ersetzen ist.

target und target_name: Verweise auf den Eintrag in der Tabelle `concept`, durch den der alte Wert ersetzt werden soll.

resource: Verweis auf den Eintrag in der Tabelle `resource` für die Wiki-Seite, auf der der Alias definiert ist.

meaning

Die Tabelle `meaning` assoziiert Terme mit Konzepten — sie repräsentiert die *Bedeutungsrelation* und ist ist damit, neben der Tabelle `broader`, das Herzstück des Thesaurus. Die Daten in dieser Tabelle werden direkt aus der Tabelle `link` gewonnen (siehe ▸9.1.3).

Die Tabelle `meaning` ist sprachspezifisch und wird daher nur für lokale Datensätze verwendet. Soll ein globales Konzept für einen Term in einer bestimmten Sprache gefunden werden, so wird zunächst über die Tabelle `meaning` das lokale Konzept für diese Sprache gefunden, und dann das zugehörige globale Konzept über die Tabelle `origin` im globalen Datensatz.

concept und concept_name: Verweise auf den Eintrag in der Tabelle `concept`, der das Konzept, also die Bedeutung, enthält. Dabei ist `concept` ein Verweis auf das Feld `id`, `concept_name` auf das Feld `name`.

term_text: Der Term, der als Bezeichnung für dieses Konzept verwendet wird.

freq: Gibt an, wie oft der Term in `term_text` für das das Konzept in `concept` verwendet wurde. Das kann zum Beispiel für einen Cutoff zur Steigerung der Präzision der Bedeutungszuweisung verwendet werden (siehe ▸10.4).

rule: Gibt den *maximalen* Code aller derjenigen Extraktionsregeln (▸9.1) an, die diesen Term diesem Konzept zuordnen (es ist also Gruppen-Maximum des `rule` Feldes aus der Tabelle `link`). Das ist vor allem nützlich, um festzustellen, ob ein Term einem Konzept *nur* durch Verwendung als Link-Text zugewiesen wurde, oder ob auch eine „stärkere“ Regel Anwendung fand, wie zum Beispiel eine explizite Weiterleitung (▸1.2). Das kann zum Beispiel für einen intelligenten Cutoff zur Steigerung der Präzision der Bedeutungszuweisung verwendet werden (siehe ▸10.4 und ▸12.5).

origin

Die Tabelle `origin` verbindet Konzepte im globalen Datensatz mit Konzepten in den einzelnen lokalen Datensätzen. Sie definiert also, welche lokalen Konzepte zu einem globalen Konzept zusammengefasst wurden und ist damit ein Kernstück des multilingualen Thesaurus: sie dient als

Bindeglied zwischen den einzelnen Sprachen (siehe auch ▹9.2.1 und ▹4.3). Die Tabelle `origin` gibt es nur in globalen Datensätzen.

`global_concept`: Verweis auf den Eintrag in der Tabelle `concept` im globalen Datensatz.

`lang` und `lang_bit`: Diese Felder identifizieren den lokalen Datensatz, indem sich das lokale Konzept befindet, das dem globalen Konzept zugeordnet werden soll. `lang` ist der Sprachcode (und damit der Name des Datensatzes), `lang_bit` ist die der betreffenden Sprache zugeordnete Bit-Maske (siehe ▹9.2).

`local_concept` und `local_concept_name`: Verweise auf einen Eintrag in der Tabelle `concept` in dem lokalen Datensatz, der durch `lang` identifiziert ist. `local_concept` bezieht sich auf das Feld `id`, `local_concept_name` auf das Feld `name`.

`relation`

Die Tabelle `relation` speichert zusätzliche semantische Beziehungen zwischen Konzepten, insbesondere *Verwandtheit* und *Ähnlichkeit*.

`concept1` und `concept2` sind Verweise auf Einträge in der Tabelle `concept` für die beiden Konzepte, die durch die Relation verbunden werden.

`langref` (nur global) gibt an, ob sich zwei Konzepte *ähnlich* sind, weil das eine über einen Language-Link auf das andere verweist. Diese Beziehungen werden beim Aufbau des globalen Datensatzes in der Importphase aus dem Inhalt der Tabelle `langlink` berechnet (siehe ▹9.2.1). Dabei bleibt die Beziehung zunächst asymmetrisch und wird nur als Verweis interpretiert — diese Information wird dann benutzt, um Paare von Konzepten zu finden, die sich gegenseitig über Language-Links referenzieren. Solche Paare sind Kandidaten dafür, in einem sprachübergreifenden Konzept zusammengeführt zu werden (siehe ▹9.2.2). In der Nachbereitungsphase (▹9.2.3) wird diese Beziehung dann für die verbleibenden Paare symmetrisch ergänzt, so dass sie als „Ähnlichkeit“ interpretiert werden kann. Diese Ähnlichkeit besteht zwischen Konzepten, die entweder nur in eine Richtung durch einen Language-Link verknüpft sind, oder die zwar beidseitig verknüpft sind, aber dennoch nicht vereinigt werden konnten, weil sie zueinander in Konflikt stehen (siehe ▹9.2.2). Da diese Relation nur zwischen Konzepten aus unterschiedlichen Sprache bestehen kann, existiert dieses Feld nur im globalen Datensatz.

`langmatch` besagt, ob sich zwei Konzepte *ähnlich* sind, weil sie über einen Language-Link auf dasselbe Konzept in einer anderen Sprache verweisen. Diese Beziehungen werden in der Nachbereitungsphase des Imports aus dem Inhalt der Tabelle `langlink` berechnet (▹9.1.3).

`bilink` gibt an, ob zwei Konzepte *verwandt* sind, weil sie *gegenseitig* über einen Wiki-Link aufeinander verweisen. Diese Information wird in der Nachbereitungsphase des Imports aus der Tabelle `link` berechnet (▹9.1.3).

merge

Die Tabelle `merge` ist ein Hilfsmittel für das Zusammenführen von Konzepten im globalen Datensatz. Sie speichert vorübergehend eine Abbildung von IDs von veralteten Konzepten und ihrem jeweiligen Ersatz. Das funktioniert ähnlich wie bei der Tabelle `alias`: beim Zusammenführen wird ein neues, vereinigtes Konzept aus zwei alten erzeugt, und dann werden für die beiden alten Konzepte Einträge in der Tabelle `merge` angelegt, welche die IDs der alten Konzepte auf die ID des neu erzeugten Konzeptes abbilden. Jede Referenz auf die alten Konzepte wird dann in einem weiteren Schritt durch eine Referenz auf das neue, vereinigte Konzept ersetzt (▹9.2.2).

old: Verweis auf den Eintrag in der Tabelle `concept`, der zu ersetzen ist.

new: Verweis auf den Eintrag in der Tabelle `concept`, durch den die alten Werte ersetzt werden sollen.

concept_description

Die Tabelle `concept_description` ist ein *Property Cache* für die sprachspezifischen Eigenschaften von Konzepten in einem lokalen Datensatz (siehe ▹9.4).

concept: Verweis auf den Eintrag in der Tabelle `concept`, für den die Eigenschaften gelten.

terms: Serialisierte Liste von Termen (mit Frequenzangaben), die für dieses Konzept verwendet werden; abgeleitet aus der Tabelle `meaning`.

concept_info

Die `concept_info` Tabelle ist ein *Property Cache* für Beziehungen zwischen Konzepten (siehe ▹9.4).

concept: Verweise auf den Eintrag in der Tabelle `concept`, für den die Beziehungen gelten.

inlinks: Serialisierte Liste von Konzepten (ID und IDF-Wert, in lokalen Datensätzen auch der Name), die auf dieses Konzept verweisen; abgeleitet aus der Tabelle `link`.

outlinks: Serialisierte Liste von Konzepten (ID und IDF-Wert, in lokalen Datensätzen auch der Name), auf die dieses Konzept verweist; abgeleitet aus der Tabelle `link`.

broader: Serialisierte Liste von Konzepten (ID und inverser LHS-Wert, in lokalen Datensätzen auch der Name), die diesem Konzept übergeordnet sind; abgeleitet aus der Tabelle `broader`.

narrower: Serialisierte Liste von Konzepten (ID und inverser LHS-Wert, in lokalen Datensätzen auch der Name), die diesem Konzept untergeordnet sind; abgeleitet aus der Tabelle `broader`.

similar: Serialisierte Liste von Konzepten (nur ID), die diesem Konzept ähnlich sind; abgeleitet aus der Tabelle `relation` unter Betrachtung des `langmatch` und des `langref` Feldes.

related: Serialisierte Liste von Konzepten (nur ID), die mit diesem Konzept verwandt sind; abgeleitet aus der Tabelle `relation` unter Betrachtung des `bilink` Feldes.

langlinks : Serialisierte Liste von Konzepten (`language` und `target` Feld), auf die dieses Konzept per Language-Link verweist; abgeleitet aus der Tabelle `langlink`.

terms

Die Tabelle `terms` speichert die *Termfrequenzverteilung*, das heißt, sie weist allen Termen im Korpus aufgrund ihrer Verwendungsfrequenz einen Rang zu (▹9.3). Dies ist nützlich als Übersicht über das verwendete Vokabular (▹11).

term: Text des Terms.

freq: Verwendungsfrequenz, abgeleitet aus der Tabelle `meaning`.

rank: Rang, gemäß der Frequenz. Kann auch als ID des Terms verwendet werden.

random: Zufallswert zur effizienten Auswahl eines zufälligen Terms.

degree

Die Tabelle `degree` speichert die *Knotengradverteilungen*, das heißt, sie weist jedem Konzept im Korpus den Rang zu, der sich aus dem *Grad* des Konzeptes als Knoten in dem von der Tabelle `link` aufgespannten Graphen ergibt (siehe auch ▹9.3). Dies ist nützlich für eine Untersuchung der Netzwerkstruktur (▹11) (vergleiche [Barabasi und Albert (1999), Steyvers und Tenenbaum (2005)]).

`concept` und `concept_name`: Verweise auf den Eintrag in der Tabelle `concept`, der das betreffende Konzept enthält. Dabei ist `concept` ist ein Verweis auf das Feld `id`, `concept_name` auf das Feld `name`.

`in_degree` und `in_rank`: Grad und entsprechender Rang bezüglich eingehender Referenzen aus der Tabelle `link`.

`out_degree` und `out_rank`: Grad und entsprechender Rang bezüglich ausgehender Referenzen aus der Tabelle `link`.

`link_degree` und `link_rank`: Grad und entsprechender Rang bezüglich eingehender und ausgehender Referenzen zusammengenommen aus der Tabelle `link` (also Betrachtung als ungerichteter Graph).

`idf` und `idf_rank`: *Inverse Document Frequency* (wobei die Dokumentfrequenz die Zahl eingehender Links ist) und der entsprechende Rang, vergleiche [Salton und McGill (1986)].

`lhs` und `lhs_rank`: *Local Hierarchy Score* (*Zentralität*) und der entsprechende Rang, nach [Muchnik u. a. (2007)] (ohne symmetrische Ergänzung).

stats

Die Tabelle `stats` speichert eine Übersicht über verschiedene statistische Werte, die für den Datensatz erhoben wurden.

block: Name des Wert-Blocks (des *Records*)

name: Name des Wertes (eindeutig innerhalb eines Blocks)

value: Wert

C.3. Zugriffsschicht für die Datenabfrage

Der Zugriff auf die Datenbankstruktur (▹C.2) ist in einer *Data Access Object*-Schicht (DAO, auch bekannt als *Data Accessor*, siehe [Nock (2004), s. 9ff] sowie [JAVA:DAO]) gekapselt. Im Gegensatz zu einem *Object/Relational Mapping* (ORM, siehe [Nock (2004), s. 53ff]), wie es z. B. von [Zesch u. a. (2007a)] verwendet wird, bietet dieser Ansatz zwar geringere Abstraktion von dem verwendeten Datenbanksystem, dafür aber mehr Kontrolle über die Verwendung der Datenbank. Auch erlaubt es die Nutzung von systemspezifischen Funktionen. Das ist insbesondere deshalb wichtig, weil wesentliche Teile der Verarbeitung aus Effizienzgründen direkt in der Datenbank ausgeführt werden: das Datenbanksystem wird von WikiWord nicht nur zur reinen Speicherung von Datensätzen verwendet, es spielt vielmehr eine wichtige Rolle bei der eigentlichen Verarbeitung und Aufbereitung der Daten, siehe insbesondere ▹9.1.3 und ▹9.2.

Die DAO-Schicht ist in verschiedene *Stores* für unterschiedliche Arten von Daten gegliedert, die im Package `de.brightbyte.wikiword.store` definiert und implementiert sind.

WikiWordConceptStore

Das Interface `WikiWordConceptStore`, implementiert durch `DatabaseWikiWordConceptStore`, ist die abstrakte Basis für alle DAO-Klassen, die als Speicher für Konzepte dienen. Ein Objekt dieses Typs repräsentiert also einen konkreten Datensatz.

`WikiWordConceptStore` definiert im Wesentlichen die folgenden Methoden:

`getNumberOfConcepts` gibt an, wie viele Konzepte in diesem Store gespeichert sind.

`listAllConcepts` gibt Referenz-Objekte für alle Konzepte zurück.

`getConcept` gibt ein einzelnes Konzeptobjekt zurück (von dem Typ, der für die konkrete Implementation passend ist).

`pickRandomConcept` und `getRandomConcept` gibt eine Referenz auf ein zufälliges Konzept zurück, bzw. direkt ein zufälliges Konzept-Objekt. Der Bereich, aus dem ausgewählt wird, kann über den Rang der Konzepte beschränkt werden.

LocalConceptStore

Das Interface `LocalConceptStore`, implementiert durch `DatabaseLocalConceptStore`, ist ein *Store* für einen lokalen Datensatz, also einen Store, der konzeptionell Objekte vom Typ `LocalConcept` enthält. Es definiert im Wesentlichen folgende Methoden:

`listMeanings` und `getMeanings` liefert alle Konzepte (Bedeutungen) für einen gegebenen Term (als Referenz bzw. Konzeptobjekt).

`getNumberOfTerms` liefert die Anzahl der unterschiedlichen Terme, die in dem Store mit Konzepten assoziiert sind.

`getAllTerms` liefert alle Terme, die in dem Store verwendet werden.

`pickRandomTerm` liefert einen zufälligen Term. Der Bereich, aus dem ausgewählt wird, kann über den Rang der Terme beschränkt werden.

GlobalConceptStore

Das Interface `GlobalConceptStore`, implementiert durch `DatabaseGlobalConceptStore`, ist ein *Store* für einen globalen Datensatz, also ein Store, der konzeptionell Objekte vom Typ `GlobalConcept` enthält. Es definiert im Wesentlichen folgende Methoden:

`listMeanings` und `getMeanings` liefert alle Konzepte (Bedeutungen) für einen gegebenen Term aus einer gegebenen Sprache (als Referenz bzw. Konzeptobjekt).

`getLocalConcepts` liefert für ein gegebenes globales Konzept alle lokalen Konzepte, die in ihm zusammengefasst sind.

`getLanguages` liefert die Sprachen aller lokalen Datensätze, die von dem globalen Store zusammengefasst werden.

C.4. Zugriffsschicht für die Datenspeicherung und Verarbeitung

Das Speichern und Ändern von Daten in der Datenbank erfolgt über *Builder*-Objekte. Ein *Builder* ist ein *Datenzugriffsobjekt* (DAO), das darauf ausgelegt ist, einen Datensatz aufzubauen oder zu manipulieren: er bietet also eine andere Perspektive auf den Datensatz als ein *Store*. Insbesondere bieten die *Builder*-Klassen eine Schnittstelle zum Einfügen von Daten in einen Datensatz, die nicht nur die Kommunikation mit der Datenbank per SQL kapselt, sondern auch den Zugriff in transparenter Weise weiter optimiert. Zwei wichtige Techniken sind dabei:

- Neue Datensätze werden nicht sofort in die Datenbank geschrieben, sondern zunächst in einem Puffer zwischengespeichert. Ist der Puffer voll, so wird er *im Hintergrund* in die Datenbank geschrieben, während bereits ein neuer Puffer befüllt werden kann. Dieser Mechanismus wurde bereits in ▹7.2 beschrieben. Das erlaubt einen sehr hohen Datendurchsatz beim anfänglichen Import der Daten (▹9.1).

- Manipulationen großer Datenmengen, wie sie in den späteren Phasen der Verarbeitung vorkommen (▹9.1.3, ▹9.2, ▹9.3, etc.), werden automatisch in kleinere Schritte unterteilt (*Chunking*) — das kann einerseits die Verarbeitung insgesamt beschleunigen, weil unter Umständen ein Überlaufen der internen Puffer des Datenbanksystems vermieden wird. Aber vor allem erlaubt es auch das Fortsetzen unterbrochener Import-Vorgänge (▹B.6), ohne dass aufwändig berechnete Daten erneut berechnet werden müssen. Die Größe der „*Chunks*“ kann über den Tweak-Wert `dbstore.updateChunkSize` gesteuert werden, per Default werden jeweils zehntausend Einträge auf einmal verarbeitet.

Die folgenden Klassen bilden die Schnittstelle für den Aufbau der Datensätze:

`LocalConceptStoreBuilder`

Die Klasse `LocalConceptStoreBuilder` wird vor allem von `ConceptImporter` bei der Transformation von Daten aus dem Ressourcenmodell in das WikiWord-Datenmodell (das *Konzeptmodell*) verwendet (▹9.1). Sie enthält im Wesentlichen die folgenden Methoden:

`storeResource` legt einen Ressource-Eintrag an, mit Name, Ressource-Typ und Zeitstempel, und gibt die ID des neuen Ressource-Eintrags zurück (siehe auch ▹C.2, ▹9.1).

`storeConcept` legt einen Konzept-Eintrag an, mit Name und Konzept-Typ, und gibt die ID des neuen Konzept-Eintrags zurück (siehe auch ▹C.2, ▹9.1).

`storeDefinition` speichert einen Definitionstext (Glosse) zu einem gegebenen Konzept (siehe auch ▹C.2).

`storeLink` speichert einen Verweis zwischen zwei Konzepten unter Angabe des Link-Textes, fügt also einen Eintrag in die Bedeutungsrelation ein und definiert eine Referenz zwischen Konzepten (siehe auch ▹C.2).

`storeReference` speichert die Verwendung eines Terms für ein Konzept, fügt also einen Eintrag in die Bedeutungsrelation ein (siehe auch ▹C.2).

`storeSection` speichert die Verwendung eines Artikelabschnitts als Link-Ziel (siehe ▹9.1.1, ▹C.2).

`storeConceptBroader` speichert eine Subsumtion, also eine Über- bzw. Unterordnung eines Konzeptes (siehe auch ▹C.2).

`storeConceptAlias` definiert ein Konzept als Alias für ein anderes. Alias-Konzepte sind eine temporäre Hilfskonstruktion, sie kommen im fertigen Thesaurus nicht vor (siehe ▹9.1.3 und ▹C.2).

`storeLanguageLink` speichert eine Verknüpfung eines Konzeptes mit einem semantisch ähnlichen (oder äquivalenten) Konzept in einem anderen Wikipedia-Projekt (also einer anderen Sprache) (siehe auch ▹C.2).

Die Klasse `LocalConceptStoreBuilder` enthält zusätzlich eine Anzahl von Methoden zur Nachbereitung der Daten (Methoden der Form `finishXXX`). Diese werden im Abschnitt ▹9.1.3 im einzelnen beschrieben.

`GlobalConceptStoreBuilder`

Die Klasse `GlobalConceptStoreBuilder` definiert Methoden zum Zusammenführen von lokalen Datensätzen zu einem globalen Datensatz. Diese Methoden operieren auf den Datensätzen als Ganzes und werden von dem Programm `BuildThesaurus` direkt verwendet ▹9.2.1:

`importConcepts` importiert alle Konzepte aus den lokalen Datensätzen in den globalen Datensatz. Dabei werden die Tabelle `concept` (▹C.2) und die Tabelle `origin` (▹C.2) aufgebaut und die Tabelle `relation` (▹C.2) teilweise initialisiert.

`buildGlobalConcepts` baut aus den importierten Konzepten einen multilingualen Thesaurus auf, indem Konzepte aus verschiedenen lokalen Datensätzen aufgrund ihrer semantischen Ähnlichkeit zusammengefasst werden (Siehe ▹9.2.2).

Die Funktion dieser Methoden wird im Abschnitt ▹9.2.1 im einzelnen erläutert.

D. Datenstrukturen

Dieses Kapitel beschreibt wichtige Datenstrukturen, die von WikiWord zur Repräsentation verschiedener Entitäten verwendet werden.

D.1. `WikiPage`

Die Klasse `WikiTextAnalyzer.WikiPage` repräsentiert eine Seite im Wiki und implementiert damit das Ressourcenmodell von WikiWord, das bereits in ▹4.1 kurz beschrieben wurde. Ihre Eigenschaften spiegeln die Eigenschaften der Seite bzw. des Konzeptes, das die Seite beschreibt, so wie sie von `WikiTextAnalyzer` extrahiert wurden. Dabei werden die einzelnen Eigenschaften verzögert initialisiert, also erst dann berechnet, wenn sie zum ersten Mal verwendet werden. Im Einzelnen hat `WikiTextAnalyzer.WikiPage` folgende Eigenschaften:

`categories`: die Menge aller Wiki-Links mit Kategorisierungssemantik, also der Links mit dem `magic`-Wert `LinkMagic.CATEGORY` (siehe ▹8.7).

`cleanedText`: der Text der Seite nach Anwendung von `WikiTextAnalyzer.stripClutter` (siehe ▹8.3).

`conceptType`: der Typ des Konzeptes, das die Seite beschreibt (siehe ▹8.6)

`disambigLinks`: die Links auf die unterschiedlichen Bedeutungen des zu klärenden Terms, falls die Seite eine Begriffsklärungsseite ist (siehe ▹8.10).

`flatText`: der Text der Seite nach Anwendung von `WikiTextAnalyzer.stripBoxes` auf den `cleanedText` (siehe ▹8.3).

`links`: alle auf der Seite definierten Wiki-Links (siehe ▹8.7).

`name`: der normalisierte Titel der Seite.

`namespace`: die numerische ID des Namensraums der Seite. Die ID ist über ein Objekt vom Typ `Namespace` mit den Namen des Namensraumes assoziiert. Sie ist über `WikiTextAlanyzer.getNamespaceId` verfügbar.

`pageTerms`: die Menge von Termen, die als Bezeichnung für das Konzept des Artikels direkt aus der Seite, insbesondere aus dem Titel, abgeleitet werden können (siehe ▹8.9).

`plainText`: der Text der Seite nach Anwendung von `WikiTextAnalyzer.stripMarkup` auf den `flatText` (siehe ▹8.3).

`redirect`: das Ziel der Weiterleitung, falls die Seite eine Weiterleitung ist. Um den Weiterleitungslink zu finden, wird das Muster verwendet, das in `WikiConfiguration.redirectPattern` definiert ist.

resourceType: der Typ der Seite (siehe ▹8.5)

sections: eine Liste aller Abschnittsüberschriften, wie sie von `WikiTextAnalyzer.extractSections` bestimmt wurde.

templates: alle auf der Seite verwendeten Vorlagen und ihre Parameter (siehe ▹8.4).

text: der vollständige, unveränderte Wiki-Text der Seite, wie in der Wikipedia verwendet.

titleBaseName: der Basisname der Seite, also der Titel ohne Klammerzusatz (siehe ▹8.9).

titleSuffix: der Klammerzusatz des Seitennamens, falls vorhanden (siehe ▹8.9), ohne die Klammern selbst.

D.2. Transferobjekte

Die DAO-Objekte benutzen *Transferobjekte*, um die eigentlichen Daten darzustellen. Sie modellieren die Kernkonzepte von WikiWord, insbesondere Terme und Konzepte, und sind im Package `de.brightbyte.wikiword.model` definiert.

Die Transferobjekte sind als nicht-modifizierbare Dateneinheiten ausgelegt. Die wichtigsten Klassen seien im Folgenden kurz beschrieben:

WikiWordReference: Basisklasse für Referenz-Objekte; Referenz-Objekte enthalten die ID und den Namen des referenzierten Objektes, sowie optional Annotationen zu Frequenz und/oder Relevanz des Objektes. Referenz-Objekte werden vor allem bei der Auflistung von Suchergebnissen verwendet.

TermReference: Referenz-Objekte für Terme. Der Term selbst ist als der Name des Referenz-Objektes gespeichert, die ID ist ungenutzt.

GlobalConceptReference und LocalConceptReference: Referenz-Objekte für lokale bzw. globale Konzepte (also Konzepte aus dem globalen bzw. aus einem lokalen Datensatz).

WikiWordConcept: Basisklasse für Konzeptobjekte. Konzeptobjekte enthalten Listen von Referenzen auf andere Konzeptobjekte, bieten also Zugriff auf die Relationen, die zwischen den Konzepten definiert sind. Das entspricht den Informationen, die in ▹9.4 im *Property Cache* gespeichert werden.

Konzepte haben in diesem Modell insbesondere die folgenden Eigenschaften:

id: Eine interne ID (aus der Tabelle `concept` ▹C.2).

name: Der Name des Konzeptes, abgeleitet aus dem Namen der Wikipedia-Seite(n), die das Konzept definieren (aus der Tabelle `concept`).

terms: Die Terme (▹8.9), die dem Konzept über die Bedeutungsrelation zugeordnet sind (aus der Tabelle `meaning`).

type: Der Konzepttyp (▹8.6), der dem Konzept zugeordnet ist (aus der Tabelle `concept`).

definition: Die Definition (▹8.8), die dem Konzept zugeordnet ist (aus der Tabelle `definition` ▹C.2).

broader: Übergeordnete Konzepte (▹9.1), die dem Konzept über die Subsumtionsbeziehung zugeordnet sind (aus der Tabelle `broader` ▹C.2).

narrower: Untergeordnete Konzepte (▹9.1), die dem Konzept über die Subsumtionsbeziehung zugeordnet sind (aus der Tabelle `broader`).

related: Verwandte Konzepte (▹9.1.3), die dem Konzept zugeordnet sind (aus der Tabelle `relation` ▹C.2).

similar: Ähnliche Konzepte (▹9.1.3), die dem Konzept zugeordnet sind (aus der Tabelle `relation`).

inLinks: Konzepte, die auf dieses Konzept verweisen (aus der Tabelle `link` ▹C.2).

outLinks: Konzepte, auf die dieses Konzept verweist (aus der Tabelle `link`).

`LocalConcept`: Konkrete Klassen für Konzeptobjekte, die lokale Konzepte repräsentieren (also Konzepte aus einem lokalen Datensatz). Zusätzlich zu den durch `WikiWordConcept` definierten Eigenschaften bietet sie direkten Zugriff auf die Definition des Konzeptes (die Glosse) sowie auf die Terme, die für das Konzept verwendet wurden (annotiert mit der Frequenz der Verwendung).

`GlobalConcept`: Konkrete Klassen für Konzeptobjekte, die globale Konzepte repräsentieren (also Konzepte aus dem globalen Datensatz). Zusätzlich zu den durch `WikiWordConcept` definierten Eigenschaften bietet sie auch Zugriff auf die `LocalConcept`-Objekte derjenigen lokalen Konzepte, die in dem globalen Konzept zusammengefasst sind.

E. Algorithmen

E.1. Extraktion von Wiki-Links

Die Methode `WikiTextAnalyzer.extractLinks` extrahiert alle Wiki-Links aus dem gegebenen Text. Der Text sollte bereits mit `stripClutter` bereinigt worden sein. Dafür werden zunächst alle Links in dem Text gesucht; der reguläre Ausdruck, der dafür verwendet wird, ist `\[\[([^\r\n]+?)(\|([^\r\n]*?))?\]\]`, gefolgt von dem Ausdruck, der in `WikiConfig.linkTrail` definiert ist. Der „*Link-Trail*“ legt fest, welche Zeichen als zum Text eines Links gehörig betrachtet werden, wenn sie unmittelbar auf den Link folgen — zum Beispiel wäre im Fall von *`[[Physik]]er`* der Link-Text „*Physiker*“, weil „*er*“ auf das Link-Trail Muster passt. Der Link-Trail ist normalerweise eine Folge solcher Zeichen, die im Inneren eines Wortes der betreffenden Sprache vorkommen können (also „normale“ Buchstaben der Sprache). Per Default sind hier alle Zeichen der Unicode-Kategorien für „Buchstaben“ erlaubt (genauer, Zeichen in den Kategorien *Lu*, *Ll*, *Lt*, *Lm* und *Lo* [UNICODE (2007), s. 139]), also folgender reguläre Ausdruck: `([\pIsL]*)`.

Für jede Fundstelle für diesen regulären Ausdruck wird nun mittels `WikiTextAnalyzer.makeLink` ein Link als Instanz der Klasse `WikiTextAnalyzer.WikiLink` erzeugt. Dabei werden drei der vier von dem Ausdruck erfassten Gruppen verwendet: Gruppe 1 als Link-Ziel, Gruppe 3 als Link-Text (falls angegeben) und Gruppe 4 als Link-Trail.

Das Vorgehen von `WikiTextAnalyzer.makeLink` ist dann folgendes:

```
 1: procedure makeLink(context, target, text, tail)
 2:     target := decodeLinkTarget(target);          ▷ Dekodiere HTML- und URL-kodierte Zeichen
 3:     if target starts with „:“ then               ▷ Entferne Doppelpunkte vom Anfang
 4:         esc :=true;                              ▷ ...und markiere Link als „escaped“, als „nicht-magisch“
 5:     end if
 6:     page := target without leading „:“;          ▷ Seitennamen initialisieren
 7:     section := page portion after first „#“, if any;   ▷ Trenne den Abschnittsnamen...
 8:     page := page portion before first „#„ if any;       ▷ vom eigentlichen Seitennamen
 9:     if section then                              ▷ Normalisiere den Abschnittsnamen
10:         section := decodeSectionName(section);
11:         section := section with all spaces replaced by „_“;
12:     end if
13:     prefix := page portion before first „:“, if any;   ▷ Trenne den Präfix vom Seitennamen
14:     if prefix then
15:         prefix := normalizeTitle(prefix);        ▷ Normalisiere den Präfix
16:         ns := getNamespaceId(prefix);            ▷ Finde Namensraum-ID für Präfix
17:         if ns ≠ Namespace.NONE then              ▷ Falls ein solcher Namensraum existiert...
18:             page := page portion after first „:“;          ▷ Entferne Präfix vom eigentlichen Seitennamen
19:             namespace := ns;                     ▷ Speichere Namensraum-ID für Nutzung im Link-Objekt
20:             if ¬esc then                         ▷ Falls der Link nicht „escaped“ ist...
21:                 if ns = Namespace.IMAGE then     ▷ Falls der Namensraum „Image“ ist...
22:                     magic :=LinkMagic.IMAGE;     ▷ setze magic auf IMAGE
23:                 else if ns = Namespace.CATEGORY then   ▷ Falls der N.r. „Category“ ist...
24:                     magic := LinkMagic.CATEGORY; ▷ setze magic auf CATEGORY
25:                 end if
26:             end if
```

```
27:         else if isInterwikiPrefix(prefix) then                    ▹ Falls der Präfix ein Interwiki-Präfix ist...
28:             page := page portion after first „:“;                 ▹ entferne Präfix vom eigentlichen Seitennamen
29:                                   ▹ Falls der Präfix ein Interlanguage-Präfix ist und der Link nicht „escaped“...
30:             if isInterlanguagePrefix(prefix) ∧ ¬esc then
31:                 magic := LinkMagic.LANGUAGE;                      ▹ setze magic auf LANGUAGE
32:             end if
33:         end if
34:     end if
35:     if ¬text then                                                 ▹ Falls kein Link-Text angegeben wurde...
36:         implied := true;                                          ▹ Merke, dass impliziter Link-Text verwendet wird
37:         if magic = LinkMagic.Category then                        ▹ Falls der Link Kategorisierungssemantik hat...
38:                                                                   ▹ setze den Link-Text (hier: Sortierschlüssel). . .
39:                                                                   ▹ auf den Namen der Seite, die den Link enthält
40:             text := context;
41:         else                                                                                        ▹ sonst...
42:             text := target;                          ▹ setze den Link-Text auf das ursprünglich angegebene Link-Ziel
43:         end if
44:     end if
45:                                                                   ▹ Falls der Link keine Kategorisierungssemantik hat...
46:     if tail ∧ magic ≠ LinkMagic.CATEGORY then
47:         text := concat(text, tail);                               ▹ hänge den Trail an den Link-Text an
48:     end if
49:     text := decodeEntities(text);                                 ▹ Dekodiere HTML-Entities im Link-Text
50:     page := normalizeTitle(page);                                 ▹ Normalisiere das Link-Ziel
51:                                                                   ▹ Gib neues Wiki-Link-Objekt zurück
52:     return new WikiLink(interwiki, namespace, page, section, text, implied, magic);
53: end procedure
```

Dabei verwendet `makeLink` (▹E.1) die folgenden Funktionen:

`WikiTextAnalyzer.decodeLinkTarget` entschlüsselt HTML Entity-Referenzen [W3C:HTML (1999)] und URL-kodierte Zeichen [RFC:3986 (2005)].

`WikiTextAnalyzer.decodeSectionName` entschlüsselt Codes in Abschnittsnamen; MediaWiki verwendet einen speziellen Code, um Sonderzeichen in Abschnittsnamen zu kodieren, die anderweitig nicht den Anforderungen an ein HTML *`id`*-Attribut [W3C:HTML (1999)] genügen würden. Diese können auch beim Angeben eines Abschnittsnamens in einem Link verwendet werden. Der Code entspricht im Wesentlichen der URL-Kodierung, nur dass hier der Punkt („.“) statt dem Prozentzeichen („%“) für die Kodierung verwendet wird.

`WikiTextAnalyzer.isInterwikiPrefix` gibt `true` zurück, falls eine gegebene Zeichenkette ein Interwiki-Präfix ist. Dazu wird die Zeichenkette mit dem regulären Ausdruck `^[a-z]+(-[a-z]+)*$` verglichen — dieses Muster ist weit gefasst und kann dazu führen, dass einige Links fälschlich als Interwiki-Links klassifiziert werden. Allerdings sind Seitennamen, die einen Doppelpunkt nicht als Trennzeichen für den Namensraum, sondern im eigentlichen Namen haben, recht selten.

`WikiTextAnalyzer.isInterlanguagePrefix` gibt `true` zurück, falls eine gegebene Zeichenkette ein Interlanguage-Präfix ist, das heißt also, ob es sich um einen bekannten Sprach-Code handelt. Die Sprach-Codes werden durch die Property-Datei

`Languages.properties` im Package `de.brightbyte.wikiword` definiert. Die dort aufgeführten Codes und Namen sind der entsprechenden Datei in MediaWiki entnommen, nämlich `languages/Names.php`.

`WikiTextAnalyzer.getNamespaceId` gibt die numerische ID für einen gegebenen Namensraum zurück (oder `Namespace.NONE`, wenn kein solcher Namensraum bekannt ist). Die kanonischen Namensräume sind in der Property-Datei `Namespaces.properties` im Package `de.brightbyte.wikiword` definiert. Die für ein konkretes Wiki-Projekt gültigen Namensräume und Namensraumnamen werden dem Abschnitt *<namespaces>* des XML-Dumps entnommen [MW:XML].

`WikiTextAnalyzer.normalizeTitle` normalisiert einen Seitentitel: der erste Buchstabe wird in einen Großbuchstaben umgewandelt, und alle Leerzeichen durch Unterstriche (_) ersetzt.

Die `page`-Eigenschaft des resultierenden `WikiLink`-Objekts wird dann mittels `WikiTextAnalyzer.isBadLinkTarget` geprüft — diese Methode benutzt das in `WikiConfiguration.badLinkPattern` festgelegte Muster, um fehlerhafte Links zu erkennen. Per Default werden vor allem solche Links ausgefiltert, deren Ziel (`WikiTextAnalyzer.WikiLink.getPage()`) „schlechte“ Zeichen enthält, wie z. B. „*{*“ oder „*}*“, „*[*“ oder „*]*“, „*|*“, usw. Auch Link-Ziele, die vor dem ersten Leerzeichen einen Doppelpunkt enthalten, der nicht von einem Leerzeichen gefolgt wird, werden als „schlecht“ betrachtet. Diese Regel fängt Links ab, die einen unbekannten Interwiki-Präfix oder einen Pseudo-Namensraum verwenden. Insbesondere werden so die sogenannten Shortcut-Redirects auf Projektseiten ausgefiltert, wie sie in vielen Wikipedias verwendet werden: *WP:LK* als Abkürzung für *Wikipedia:Löschkandidaten*, usw.

Falls `isBadLinkTarget` also `true` liefert, wird der Link verworfen, ansonsten wird er der Liste hinzugefügt, die am Ende von `extractLinks` zurückgegeben wird.

E.2. Extraktion von Glossen

Per Konvention ist der erste Satz in einem Wikipedia-Artikel eine Definition des Konzeptes, das der Artikel beschreibt. Um also eine Glosse für dieses Konzept zu erhalten, kann einfach dieser erste Satz extrahiert werden.

Um das zu erreichen, wird zunächst mit Hilfe von `WikiTextAnalyzer.extractFirstParagraph` der erste Absatz der Seite bestimmt. Dieser Absatz wird dann über das Muster `LanguageConfiguration.sentencePattern` in „Schnipsel“ (*Chunks*) zerlegt. Einem Puffer wird dann immer ein Schnipsel hinzugefügt, solange das Ende des Puffers auf das Muster in `LanguageConfiguration.sentenceTailGluePattern` passt (oder der Puffer leer ist), oder der Anfang des nächsten Schnipsels zu `LanguageConfiguration.`

`sentenceFollowGluePattern` passt. Der erste Satz gilt als gefunden, wenn keine der beiden Bedingungen mehr zutrifft, oder es keine weiteren Schnipsel gibt. In Pseudocode:

```
 1: procedure EXTRACTFIRSTSENTENCE(text)
 2:     apply sentenceManglers to text;                    ▷ Entferne unerwünschten Text
 3:     s = „“;                                            ▷ Starte mit leerem Satz
 4:     chunk = first chunk in text matching sentencePattern;   ▷ Finde ersten Schnipsel
 5:     while chunk do                                     ▷ Solang noch Schnipsel übrig sind...
 6:         s := append(s, chunk);                         ▷ hänge Schnipsel an den Satz an
 7:         if next thing in text is a linebreak then      ▷ Ein Zeilenumbruch beendet den Satz
 8:             break;
 9:         end if
10:         chunk = next chunk if text matching sentencePattern;   ▷ Finde nächsten Schnipsel
11:         if s matches sentenceTailGlue then
12:             continue;                                  ▷ Satzende verlangt nach Verlängerung
13:         end if
14:         if chunk matches sentenceFollowGlue then
15:             continue;                                  ▷ Nächster Schnipsel verlangt nach Verlängerung
16:         end if
17:         break;                                         ▷ Sonst ist der Satz fertig.
18:     end while
19:     return return s;
20: end procedure
```

Der obige Pseudocode verwendet die folgenden Konfigurationsvariablen:

`LanguageConfiguration.sentenceManglers` dienen der Vorverarbeitung des Textes vor der Satzerkennung. Sie entfernen unerwünschte Elemente, per Default vor allem Textstücke in Klammern, die die Erkennung von Satzenden verkomplizieren würden. Geklammerte Teile des ersten Satzes enthalten meist etymologische Erläuterungen oder die Lebensdaten von Personen, also Informationen, die für eine *Definition* des Konzeptes nicht relevant sind.

`LanguageConfiguration.sentencePattern` ist ein Muster zur vorläufigen Satzerkennung, also zur Zerlegung des Absatzes in „Schnipsel“. Es erkennt potentielle Satzenden, per Default Zeilenenden und Punkte dann, wenn auf sie ein Leerzeichen folgt. Andere Zeichen, die in freiem Text ein Satzende anzeigen können (also Ausrufungszeichen und Fragezeichen) kommen in der Wikipedia in dieser Funktion kaum vor und sollten vor allem niemals den einleitenden Definitionssatz abschließen — daher werden sie hier nicht beachtet.

`LanguageConfiguration.sentenceTailGlue` definiert ein Muster, das „unechte“ Satzenden erkennt: falls das Ende des bisher gefundene Textes auf dieses Muster passt, so ist der Satz vermutlich noch nicht vollständig und es sollte ein weiterer „Schnipsel“ angehängt werden. Diese Muster enthält vor allem Abkürzungen, die mit einem Punkt enden, insbesondere solche, die auch häufig in Definitionssätzen vorkommen, wie z. B. `Dr.` („*Doktor*“) oder `geb.` („*geboren*“).

`LanguageConfiguration.sentenceFollowGlue` definiert ein Muster, das Satzfortsetzungen findet: falls der Anfang des nächsten „Chunks“ auf dieses Muster passt, so sollte er an

den bereits gefundenen Text angehängt werden. Dieses Muster passt per Default auf alle Kleinbuchstaben, also Zeichen in der Unicode-Kategorie *Ll* [UNICODE (2007), s. 139].

Auf diese Weise kann recht zuverlässig der erste Satz im Fließtext eines Wikipedia-Artikels gefunden werden (▹12.6). In einigen Fällen werden aber Satzgrenzen nicht erkannt, es werden also die ersten beiden Sätze zurückgegeben: das ist insbesondere dann der Fall, wenn der erste Satz mit einer Abkürzung endet, zum Beispiel mit „`etc.`“, was aber auf Grund der Natur des Definitionssatzes selten ist.

E.3. Analyse von Begriffsklärungsseiten

Bei der Analyse von Begriffsklärungsseiten ist das Ziel, von allen Wiki-Links auf der Seite diejenigen zu finden, die auf Bedeutungen des zu klärenden Begriffes verweisen. Das vorgeschlagene Verfahren ist darauf ausgerichtet, in einfachen Fällen mit geringem Aufwand schnell ein Resultat zu liefern, in „schwierigen“ Fällen aber dennoch präzise Ergebnisse zu erzielen. Sie arbeitet wie folgt:

```
 1: procedure EXTRACTDISAMBIGLINKS(title, text)                ▹ Finde alle Links zu Bedeutungen
 2:     links := ∅;
 3:     if disambigStripSectionPattern then          ▹ Falls ein bestimmter Abschnitt ignoriert werden soll...
 4:         text =stripSections(text, disambigStripSection);                    ▹ ...entferne ihn.
 5:     end if
 6:     name := determineBaseName(title);                          ▹ Bestimme den zu klärenden Term
 7:     for line matching disambigLinePattern do                       ▹ Für alle relevanten Zeilen...
 8:         link =determineDisambigTarget(name, line);          ▹ ...finde den richtigen Link in der Zeile
 9:         if link then                                    ▹ Falls ein solcher Link gefunden wurde...
10:             add link to links;                              ▹ ...füge ihn zu der Ergebnisliste hinzu
11:         end if
12:     end for
13: end procedure
14: procedure DETERMINEDISAMBIGTARGET(term, line)              ▹ Finde den Bedeutungslink in einer Zeile
15:     for pattern in disambigLinkPatterns do                   ▹ Für alle speziellen Link-Muster...
16:         if line matches pattern then                              ▹ ...prüfe ob das Muster passt
17:             return a link constructed from the match;      ▹ Falls ja, gib den entsprechenden Link zurück
18:         end if
19:     end for
20:                                  ▹ Weiter, falls mittels disambigLinkPatterns kein Link gefunden wurde.
21:     links := extractLinks(term, line);                      ▹ Extrahiere alle Links aus der Zeile
22:     if size(links) = 0 then                              ▹ Falls es keine Links in der Zeile gibt...
23:         return null;                                                  ▹ ...gib null zurück
24:     else if size(links) = 1 then                    ▹ Falls es nur genau einen Link in der Zeile gibt...
25:         return links[0];                                              ▹ ...gib diesen zurück
26:     end if
27:                                          ▹ Alle Links, die „irgendwo“ den Term enthalten
28:     links2 := {link ∈ links|SubStringLinkFilter.matches(name, link)};
29:     if size(links2) > 0 then                                 ▹ Falls es solche Links gibt...
30:         if size(links2) = 1 then                      ▹ falls es genau einen solchen Link gibt...
31:             return links2[0];                                         ▹ ...gib diesen zurück...
32:         end iflinks := links2;                ▹ ansonsten betrachte im weiteren nur noch die gefilterte Liste
```

```
33:     end if
34:                                  ▷ Falls es keine solchen Links gab, wird weiter die ungefilterte Liste von Links verwendet
35:     for link ∈ links do                                                              ▷ Für jeden Link...
36:                                                                                      ▷ ...berechne, wie gut er zu dem Term passt
37:         match[link] := DefaultLinkSimilarityMeasure.measure(term, link);
38:     end for
39:     best := link ∈ links|bestmatch[link];                                            ▷ Link, der am besten zu dem Term passt
40:                                                                                      ▷ Wenn selbst beste Ähnlichkeit zu gering...
41:     if match[best] < minDisambigLinkSimilarity then
42:         return null;                                                                 ▷ ...dann gib null zurück
43:     end if
44:     return best;
45: end procedure
```

Der oben beschriebene Algorithmus verwendet die folgenden Funktionen und Konfigurationsparameter:

`WikiConfiguration.disambigStripSectionPattern` ist ein Muster für die Namen von Abschnitten, die bei der Verarbeitung von Begriffsklärungsseiten ignoriert werden sollen. Insbesondere sind das Abschnitte für Querverweise, also z. B. für die deutschsprachige Wikipedia *`Siehe auch`* und für die englischsprachige *`See also`*. Gegebenenfalls wird die Methode `WikiTextAnalyzer.stripSections` verwendet, um jeden Abschnitt zu entfernen, auf den dieses Muster passt.

`WikiTextAnalyzer.determineBaseName` gibt, wie bereits oben (in ▷8.9) beschrieben, den Titel ohne Klammerzusatz zurück. Im Fall einer Begriffsklärungsseite ist das der zu klärende, mehrdeutige Term.

`WikiConfiguration.disambigLinePattern` ist ein Muster für solche Zeilen auf einer Begriffsklärungsseite, die typischerweise Links auf Bedeutungen des zu klärenden Terms haben. Per Konvention werden Bedeutungen in Listen aufgezählt, per Default werden also Zeilen betrachtet, die mit „*“ oder „#“ beginnen — allerdings sollten Zeilen ignoriert werden, die mit einem Doppelpunkt enden, da diese in der Regel keine Bedeutung auflisten, sondern einen Abschnitt in der Bedeutungsliste bezeichnen. Der reguläre Ausdruck für Disambiguierungszeilen ist per Default `^:*[*#]+(.+[^:\s])\s*$`.

`WikiConfiguration.disambigLinkPatterns` ist ein Muster für Formulierungen, von denen bekannt ist, dass sie, falls sie in einer Begriffsklärungszeile vorkommen, den gesuchten Link auf den Bedeutungsartikel enthalten. Per Default ist kein solches Muster konfiguriert, es könnte aber verwendet werden, um zum Beispiel gängige Formulierungen wie *`siehe [[xyz]]`* auszunutzen, um den richtigen Link in der Zeile zu finden.

`PlainTextAnalyzer.SubStringLinkFilter` ist ein Filter, der auf solche Links passt, die einen bestimmten Term als Substring in ihrem Link-Ziel, Abschnittstext oder Link-Text enthalten. Er benutzt vereinfachte Formen dieser Strings, genau wie `DefaultLinkSimilarityMeasure.measure` (Kleinschreibung, keine Klammerzusätze, etc).

PlainTextAnalyzer.DefaultLinkSimilarityMeasure ist ein Maß für die Ähnlichkeit von Links mit einem gegebenen Term — die Berechnung der Ähnlichkeit ist wie oben beschrieben in der Methode `DefaultLinkSimilarityMeasure.measure` implementiert. In der tatsächlichen Implementation werden Instanzen dieser Klasse über `WikiConfiguration.linkSimilarityMeasureFactory` erzeugt, dort kann auch die konkrete Klasse konfiguriert werden.

WikiConfiguration.minDisambigLinkSimilarity gibt die minimale Ähnlichkeit an, die ein Link zu dem gesuchten Term aufweisen muss, um als Bedeutungslink in Betracht zu kommen. Hat kein Link in der Zeile einen Ähnlichkeitswert, der `minDisambigLinkSimilarity` übersteigt, so wird die Zeile übergangen.

Im Folgenden ist der Algorithmus beschrieben, der von `PlainTextAnalyzer.DefaultLinkSimilarityMeasure` verwendet wird, um die lexikographische Ähnlichkeit eines Links mit einem gegebenen Term zu bestimmen.

```
 1:                                                  ▷ misst zwischen einem Link und einem Term.
 2: procedure DefaultLinkSimilarityMeasure.measure(term, link)
 3:     p := trimAndLower(determineBaseName(link.page));          ▷ Vereinfache das Link-Ziel
 4:     s := trimAndLower(normalizeTitle(link.section));          ▷ Vereinfache den Abschnittsnamen
 5:     t := trimAndLower(normalizeTitle(link.text);              ▷ Vereinfache den Link-Text
 6:     ps := measureTextMatch(term, p);                          ▷ Vergleiche Term mit Link-Ziel
 7:     ss := measureTextMatch(term, s);                          ▷ Vergleiche Term mit Abschnittsnamen
 8:     ts := measureTextMatch(term, t);                          ▷ Vergleiche Term mit Link-Text
 9:     return max(ps, ss, ts);                                   ▷ Gib besten Ähnlichkeitswert zurück
10: end procedure
11:                                  ▷ Misst, wie gut ein Name zu einem Term passt (Wert zwischen 0 und 1).
12: procedure DefaultLinkSimilarityMeasure.measureTextMatch(term, name)
13:     if term = name then                                       ▷ Falls Name und Term identisch sind...
14:         return 1;                                             ▷ ... gib 1 zurück.
15:     end if
16:     subsim := calculateSubstringSimilarity(term, name);       ▷ Zeichenweise Abdeckung
17:     levsim := calculateLevenshteinSimilarity(term, name);     ▷ Levenshtein-Ähnlichkeit
18:     wordsim := calculateWordOverlapSimilarity(term, name);    ▷ Überlappung der Wortmengen
19:     acrosim := calculateAcronymSimilarity(term, name);        ▷ Levenshtein auf Anfangsbuchstaben
20:     return max(subsim, levsim, wordsim, acrosim);             ▷ Gib besten Wert zurück
21: end procedure
```

Dieser Algorithmus verwendet die folgenden Funktionen zur Berechnung von lexikographischer Ähnlichkeit zwischen Zeichenketten (der resultierende Ähnlichkeitswert liegt immer zwischen 0 und 1):

DefaultLinkSimilarityMeasure.calculateSubstringSimilarity berechnet die Ähnlichkeit eines Namens n zu dem gegebenen Term t wie folgt: falls t nicht in n vorkommt, ist die Ähnlichkeit 0. Sonst ist die Ähnlichkeit gegeben durch den Anteil, zu dem der Term den Namen abdeckt:

$$sim_{sub} = \frac{|t|}{|n|}$$

`DefaultLinkSimilarityMeasure.calculateLevenshteinSimilarity` berechnet die Ähnlichkeit eines Namens n zu dem gegebenen Term t auf Basis der Levenshtein-Distanz [Levenshtein (1966)] $lev(n, t)$ als normierter Rest verbleibend auf die maximal mögliche Distanz:

$$sim_{lev} = \frac{max(|n|, |t|) - lev(n, t)}{max(|n|, |t|)}$$

`DefaultLinkSimilarityMeasure.calculateWordOverlapSimilarity` berechnet die Ähnlichkeit eines Namens n zu dem gegebenen Term t als die normierte Überlappung der Wortmengen $w(n)$ und $w(t)$ gemäß der Formel

$$sim_{words} = \frac{|w(n) \cap w(t)|}{max(|w(n)|, |w(t)|)}$$

Zur Zerlegung der Zeichenketten in Wortmengen wird die Methode `PlainTextAnalyzer.extractWords` verwendet, die ihrerseits das in `LanguageConfiguration.wordPattern` festgelegte Muster benutzt. Per Default ist der verwendete reguläre Ausdruck `\p{L}+|\p{Nd}+`, also jede Folge von entweder Buchstaben oder Ziffern, gemäß den Unicode-Kategorien für Buchstaben und Dezimalziffern, nämlich *Lu*, *Ll*, *Lt*, *Lm*, *Lo* und *Nd* [UNICODE (2007), s. 139].

`DefaultLinkSimilarityMeasure.calculateAcronymSimilarity` berechnet die Ähnlichkeit eines Namens n mit dem gegebenen Term t unter der Annahme, dass t ein Akronym für n ist: Sei $acro(n)$ eine Funktion, die die Konkatenation der Anfangsbuchstaben der Wörter aus n liefert (wie durch `LanguageConfiguration.wordPattern` bestimmt), dann ist die Ähnlichkeit gegeben durch:

$$sim_{acro} = sim_{lev}(t, acro(n))$$

Diese Ähnlichkeitsfunktion ist besonders deshalb wichtig, weil Begriffsklärungsseiten häufig verschiedene Bedeutungen von Akronymen aufzählen, der zu klärende Term also ein Akronym ist: z. B. listet die Seite *AA* in der deutschsprachigen Wikipedia unter anderem Links auf die Bedeutungen *Alemannia Aachen*, *American Airlines*, *Anonyme Alkoholiker* und *Arbeitsagentur*. Daher ist es wichtig, dass der Term „*AA*“ als lexikographisch ähnlich zu solchen Namen erkannt wird.

Dieses Verfahren wurde in ▸13 evaluiert.

E.4. Zusammenführen von sprachübergreifenden Konzepten

Nachdem die Ähnlichkeiten von Konzepten wie in ▸9.2.2 beschrieben bestimmt wurden, werden nach folgendem Schema Konzepte aus unterschiedlichen Sprachen zu einem sprachübergreifenden Konzept zusammengefasst:

```
1: procedure clusterGlobalConcepts(concepts, langref)          ▷ Vereinige passende Konzepte
2:                                  ▷ langref enthält Language-Link Referenzen zwischen Konzepten
3:                          ▷ gegenseitige Referenzierung via Language-Link wird als „Ähnlichkeit“ interpretiert
4:     similar := {(x, y)  |  (x, y) ∈ langref ∧ (y, x) ∈ langref};
5:     repeat                                  ▷ das Zusammenführen ist in „Runden“ aufgeteilt
6:         replace := findMergeCandidates(concepts, similar);
7:                          ▷ Neue, vereinigte Konzepte wurden hinzugefügt, die alten wurden aber noch erhalten
8:                          ▷ replace enthält nun Paare von Konzept-IDs, die als Abbildung von den IDs der alten
9:                          ▷ auf die IDs der neuen, vereinigten Konzepte dient. Die neuen, vereinigten Konzepte
10:                                                  ▷ wurden in concepts aufgenommen.
11:        performMerge(replace, similar);          ▷ Vereinigte Konzepte in allen Relationen ersetzen
12:    until replace = ∅
13: end procedure
14: procedure findMergeCandidates(concepts, similar)          ▷ Finde zu vereinigende Konzepte
15:     stop := ∅;                                  ▷ Stoppliste von bereits verarbeiteten Konzepten
16:     replace := ∅;                                  ▷ Liste von ausstehenden Ersetzungen
17:    for x ∈ concepts do
18:        if x ∈ stop then                          ▷ während dieser Runde bereits verarbeitet
19:            continue;
20:        end if
21:        stop.add(x);                                          ▷ als verarbeitet markieren
22:        sim := {y ∈ concepts  |  (x, y) ∈ similar};                    ▷ Ähnliche Konzepte
23:        for y ∈ concepts do
24:            if y ∈ stop then                      ▷ während dieser Runde bereits verarbeitet
25:                continue;
26:            end if
27:            stop.add(y);                                      ▷ als verarbeitet markieren
28:            if (x.language_bits&y.language_bits) > 0) then    ▷ Konflikt, Vereinigen nicht möglich;
29:                similar.remove( (x, y) );
30:                continue;
31:            end if
32:            m := createMergedConcept(x, y);                   ▷ vereinigtes Konzept erzeugen
33:            concepts.add(m);                                  ▷ vereinigtes Konzept hinzufügen
34:            replace.add( (x, m) );                            ▷ zur Ersetzung vormerken
35:            merge.add( (y, m) );                              ▷ zur Ersetzung vormerken
36:        end for
37:    end for
38:    return replace;
39: end procedure
40: procedure performMerge(replace, similar)          ▷ Ersetzt Referenzen von alten Konzepten
41:    Ersetze alle Vorkommen der alten einzelnen Konzepte in replace durch die neuen,
42:    vereinigten, in allen Relationen inklusive similar.
43:    Auch die Tabellen origin und relation werden entsprechend angepasst.
44:    Nach der Ersetzung, lösche die entstandenen Schleifen aus allen Relationen.
45:    Lösche nun die alten Konzepte.
46: end procedure
47: procedure createMergedConcept(x, y)                       ▷ Erzeuge kombiniertes Konzept
48:    z := new Concept();                                    ▷ Erzeuge neues Konzeptobjekt
49:    z.name := x.name + „|“ + y.name;                       ▷ Konkatenation der beiden Namen
50:    z.language_bits := x.language_bits|y.language_bits;    ▷ Bit-OR der Language-Bits
51:    if  x.type = UNKNOWN ∨ x.type = OTHER  then            ▷ Wenn x keinen spezifischen Typ hat...
52:        z.type := y.type;                                  ▷ ... dann verwendet den Typ von y
53:    else if  y.type = UNKNOWN ∨ y.type = OTHER  then       ▷ Wenn y keinen spezifischen Typ hat...
54:        z.type := x.type;                                  ▷ ... dann verwendet den Typ von x
55:    else y.ltype = UNKNOWN ∨ y.type = OTHER                ▷ Wenn beide einen spezifischen Typ haben...
```

```
56:         if x.type ≠ y.type then    ▹ Wenn zwei Konzepte vereinigt werden, die unterschiedliche, spezifische ▹
    Konzepttypen haben, ist das ein Hinweis auf einen Fehler.
57:             log type conflict.
58:         end if
59:         z.type := x.type;                                          ▹ ... Verwendet den Typ von x
60:     end if
61:     return z;
62: end procedure
```

Das Zusammenführen findet also in „Runden“ statt: In jeder Runde werden zunächst alle Konzepte durchlaufen und für jedes nach einem „Partner“ zum Zusammenführen gesucht. Dann werden alle solchen Paare von Partnern zu neuen Konzepten zusammengeführt und sämtliche Referenzen auf die alten Konzepte durch eine Referenz auf das neue, kombinierte Konzept ersetzt. Wird in einer Runde kein zu vereinigendes Paar mehr gefunden, terminiert der Algorithmus.

Dabei ist zu beachten, dass jedes der globalen Konzepte immer eine Menge von sprachspezifischen Konzepten ist, wobei die Größe dieser Menge durch die Anzahl der betrachteten Sprachen begrenzt ist, da ja aus jeder Sprache höchstens ein lokales Konzept zu einem globalen Konzept gehören kann. Andererseits wächst die Größe dieser Mengen bei jeder Runde an: in der ersten Zusammenführungsrunde entstehen Gruppen der Größe zwei, in der zweiten Zusammenführungsrunde entstehen Gruppen der Größe von mindestens drei bis höchstens vier, in der dritten Runde zwischen mindestens vier bis höchstens acht, und so weiter. Damit ist klar, dass die Anzahl der betrachteten Sprachen auch die Anzahl der maximal notwendigen Vereinigungsrunden begrenzt.

Auch die Anzahl der maximal zu betrachtenden Partner für jedes Konzept ist durch die Anzahl der Sprachen beschränkt, da die Partner ja durch gegenseitige Verknüpfung über Language-Links bestimmt werden, pro Sprache und Artikel aber nur ein Language-Link zulässig ist. Ist nun k die Anzahl der Sprachen und n die Anzahl der Konzepte, so ist die Laufzeit des Vereinigungsalgorithmus beschränkt durch $O(k \cdot n)$ — beachtet man den Aufwand der Lookups für die verschiedenen Mengen, ergibt sich $O(k \cdot n \cdot log(n))$.

E.5. Algorithmus zum schnellen Entfernen aller Zyklen aus einem gerichteten Graphen

Die Kategoriestruktur der Wikipedia ist per Konvention als (gerichteter) azyklischer Graph angelegt. In der Praxis kommen durchaus Zyklen vor, die gefunden und entfernt werden müssen, um eine Thesaurus-Relation der Über- bzw. Unterordnung zu erhalten, die den Kriterien für eine *Halbordnung* genügt. Hier wird nun ein Algorithmus vorgestellt, der diese Aufgabe auf einem Graphen mit potentiell mehreren Millionen Knoten effizient bewältigen kann.

Die Grundidee ist, alle möglichen Pfade durch den Graphen zu durchlaufen und dabei auf Zyklen zu untersuchen: Ausgehend von allen „Wurzeln“ (Knoten ohne eingehende Kanten) wird mittels *Tiefensuche* jeder Knoten besucht; liegt der nächste Knoten bereits auf dem aktuellen Pfad, so ist ein „Backlink“, und damit ein Zyklus gefunden.

Zu beachten ist, dass jeder Knoten nur einmal besucht werden muss: ist ein Knoten einmal abgearbeitet (und damit auch alle von diesem Knoten aus erreichbaren Knoten), so ist der Knoten nicht (mehr) Teil eines Zyklus, und es ist auch kein Zyklus (mehr) von ihm aus erreichbar. Wird der Knoten also später beim Durchlaufen eines anderen Pfades noch einmal gefunden, muss er nicht mehr bearbeitet werden: Die Tiefensuche beschränkt sich damit auf einen *Spanning Tree* des Graphen. Damit liegt der Aufwand des Algorithmus bei $O(n)$, wobei n die Anzahl der Knoten ist.

Zunächst wird der Graph so normiert, dass es nur noch eine einzige „Wurzel“ (Knoten ohne eingehende Kanten) gibt: Es wird ein neuer Knoten als „Elternknoten“ aller bisherigen Wurzeln eingefügt; Hat der Graph bislang gar keine Wurzel, so wird ein beliebiger Knoten aus jedem separaten Teilbaum als Kind des künstlichen Wurzelknotens bestimmt.

Ausgehend von dieser Wurzel wird nun eine *Tiefensuche* durchgeführt, wobei der Pfad von der Wurzel zum aktuellen Knoten mitgeführt wird. Bei jedem Schritt wird zunächst geprüft, ob der aktuelle Knoten bereits in der Menge der bereits verarbeiteten Knoten ist: Wenn nicht, so wird der Knoten dieser Menge hinzugefügt und dann die Tiefensuche rekursiv fortgesetzt. Wenn der Knoten aber bereits vorher einmal bearbeitet wurde, so wird geprüft, ob er auch Teil des aktuellen Pfades ist, ob also ein „Backlink“ gefunden wurde. Wenn das so ist, so wird diese Kante entfernt. In jedem Fall aber wird die rekursive Suche abgebrochen, wenn der Knoten schon einmal bearbeitet worden ist.

Der aktuelle Pfad, der von der Wurzel aus durch den Graph genommen wurde, wird bei der Tiefensuche mitgeführt. Ist nun der für den nächsten Schritt vorgesehene Knoten bereits in dem Pfad *zu* diesem Knoten enthalten, so ist ein „Backlink“, und damit ein Zyklus, gefunden. Um den Zyklus zu beheben, wird nun einfach der soeben gefundene „Backlink“ entfernt — auf diese Weise entstehen keine Inseln, alle Knoten auf dem Pfad bleiben weiter von der Wurzel aus erreichbar.

In der Praxis hat sich gezeigt, dass es sich lohnt, den Graphen vor der Anwendung dieses Algorithmus zunächst so zu reduzieren, dass er keine „losen Enden“ mehr hat. Die Idee ist, dass ein „Blatt“ (ein Knoten ohne ausgehende Kanten) auf der Suche nach Zyklen nicht zu betrachtet werden braucht. Solche Knoten können also sofort entfernt werden — und zwar wiederholt, solange, bis keine „Blätter“ mehr vorhanden sind und jeder verbleibende „Ast“ des Graphen durch einen Zyklus abgeschlossen wird.

Algorithm 1 Beseitigen von Zyklen

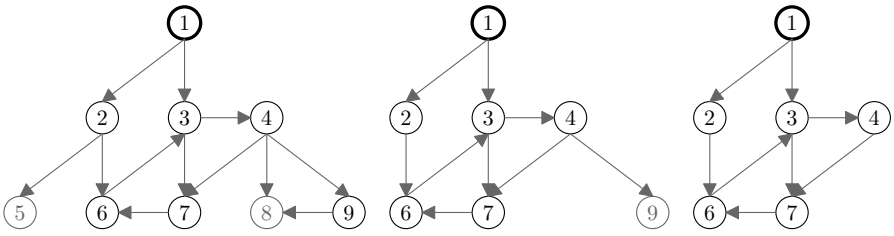
```
 1: G = (V, E);                                  ▷ Ein Graph mit Knoten V und Kanten E
 2: X = ∅;                                       ▷ Die Menge von schon bearbeiteten Knoten
 3: P = ();                                      ▷ Liste, die den aktuellen Pfad speichert
 4: r = makeRoot(G);                             ▷ Künstliche Wurzel einfügen
 5: decycle(E, X, P, r);
 6: procedure DECYCLE(E, X, P, n)
 7:     if n ∈ X then                            ▷ Schon einmal bearbeitet
 8:         if n ∈ P then                        ▷ Liegt auf dem Pfad
 9:             E.remove(P.last, n);             ▷ Zyklus gefunden!
10:         end if
11:     else
12:         X := X ∪ n;                          ▷ Als bearbeitet markieren
13:         P.append(n);                         ▷ Knoten an Pfad anhängen
14:         for m ∈ m : (n, m) ∈ E do            ▷ Für alle Kinder
15:             decycle(E, X, P, m);             ▷ Rekursiv aufrufen
16:         end for
17:         P.remove(n);                         ▷ Knoten von Pfad entfernen
18:     end if
19: end procedure
```

Algorithm 2 Reduktion des Graphen

```
1: procedure PRUNE(G = (V, E))                  ▷ Ein Graph mit Knoten V und Kanten E
2:     while ∃v ∈ V : ∄(v, w) ∈ E do            ▷ Solange es blätter gibt
3:         for v ∈ V : ∄(v, w) ∈ E do            ▷ Für jedes Blatt...
4:             G.remove(v);                     ▷ Entferne es
5:         end for
6:     end while
7: end procedure
```

Realisieren lässt sich dieser Ansatz wie folgt: Solange es Blätter im Graphen gibt, wird die Menge aller Blätter durchlaufen, und jedes Blatt entfernt (Alg. 2). Die äußere Schleife ist notwendig, da ja durch das Entfernen eines Blattes neue Blätter entstehen können. Der Aufwand der inneren Schleife liegt bei n Schritten; die Anzahl von Iterationen der äußeren Schleife ergibt sich aus der Länge des längsten Pfades von einer Wurzel zu einem Blatt — im schlechtesten Fall also ebenfalls n, im Fall eines balancierten Baumes oder eines *Small-World* Graphen, wie der Kategorie-Struktur der Wikipedia [Voss (2006), Voss (2005b)], aber nur $log(n)$.

Beispiel



(a) Ausgangsdaten (b) Reduzieren (c) Weiter

Abb. E.1: Reduzieren des Graphen

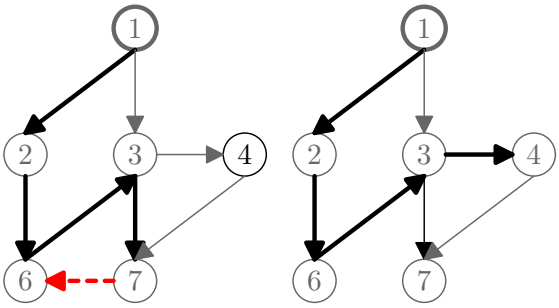
Sei ein Graph mit neun Knoten, $n1 - n9$, gegeben; $n1$ ist der künstlich hinzugefügte Wurzelknoten, er wurde über den „Proto-Wurzeln“ $n2$ und $n3$ eingehängt. Der Graph enthält zwei (sich überlappende) Zyklen: $(n3, n7, n6)$ und $(n3, n4, n7, n6)$.

Zunächst soll der Graph reduziert werden: Im ersten Schritt werden nun alle Blätter markiert ($n5$ und $n8$, siehe Abb. E.1(a)) und entfernt. Im ersten Schritt werden wieder alle Blätter markiert ($n9$, Abb. E.1(b)) und entfernt. Nun gibt es keine Blätter mehr (Abb. E.1(c)) und der eigentliche Algorithmus zum finden von Zyklen wird gestartet.

Ausgehend von der Wurzel $n1$ wird mittels Tiefensuche der Graph durchlaufen. Bei dem Schritt von $n7$ zu $n6$ wird erkannt, dass $n7$ bereits auf dem aktuellen Pfad $(n1, n2, n6, n3, n7)$ liegt: somit ist ein Zyklus gefunden, der zu entfernende „Backlink“ ist die Kante $(n7, n6)$ (Abb. E.2(a)). Die Tiefensuche wird dann fortgesetzt: der nächste interessante Schritt ist der von $n4$ nach $n7$: er wird nicht vollzogen, weil $n7$ als „schon bearbeitet“ markiert ist. Dasselbe gilt für den Schritt von $n1$ nach $n3$. Damit ist der Algorithmus beendet (Abb. E.2(b)): alle Zyklen sind beseitigt, die verwendeten Kanten bilden einen *Spanning Tree* über dem Graphen.

Anwendung

Das hier beschriebene Vorgehen ist also eine Tiefensuche in einem gerichteten Graph, während der Zyklen in diesem Graph gefunden und beseitigt werden. Die naive Tiefensuche wurde optimiert a) durch Beschränkung auf einen *Spanning Tree* und b) durch Reduktion des Graphen durch iteratives Entfernen aller Blätter. Der resultierende Algorithmus ist geeignet, Zyklen aus großen Graphen



(a) Zyklus gefunden (b) Spanning Tree

Abb. E.2: Entfernen von Zyklen

effizient zu entfernen. Das wird im Rahmen dieser Arbeit dazu verwendet, unerwünschte zyklische Verknüpfungen aus der Kategoriestruktur der Wikipedia zu entfernen, um zu einer Halbordnung von über- bzw. untergeordneten Konzepten zu gelangen.

Eine Implementation dieses Algorithmus befindet sich in der Klasse `CycleFinder` im Package `de.brightbyte.wikiword.store.builder` und wird in der Methode `deleteBroaderCycles` in der Klasse `DatabaseWikiWordConceptStoreBuilder` in demselben Package verwendet. Im Experiment ergab sich eine Laufzeit von unter 30 Sekunden für einen Graph mit 65 000 Knoten und 2 Millionen Kanten, wobei etwa 500 Zyklen gefunden wurden[1].

[1] Angaben für die Anwendung auf die deutschsprachige Wikipedia

F. Daten

F.1. Wiki-Dumps

Wikipedia-Dumps

Wikipedia-Dumps [MW:XML], die die Grundlage für die Experimente in ▸IV bilden, stehen auf den Servern der Wikimedia Foundation zur Verfügung [WM:DOWNLOAD]:

- Der Dump der aktuellen Artikelversionen der englischsprachigen Wikipedia (`en`) vom 3. Januar 2008, <`http://download.wikimedia.org/enwiki/20080103/enwiki-20080103-pages-articles.xml.bz2`> (3,2 GB komprimiert, MD5: `c04f8a1474a88ddfa3c6965fc5169208`).
- Der Dump der aktuellen Artikelversionen der deutschsprachigen Wikipedia (`de`) vom 20. März 2008, <`http://download.wikimedia.org/dewiki/20080320/dewiki-20080320-pages-articles.xml.bz2`> (1,1 GB komprimiert, MD5: `28f939fee350eac9c7a98c820fe0140e`).
- Der Dump der aktuellen Artikelversionen der französischsprachigen Wikipedia (`fr`) vom 23. März 2008, <`http://download.wikimedia.org/frwiki/20080323/frwiki-20080323-pages-articles.xml.bz2`> (783,5 MB komprimiert, MD5: `7776cfdd7f237f8522a6606110f155c9`).
- Der Dump der aktuellen Artikelversionen der niederländischsprachigen Wikipedia (`nl`) vom 10. März 2008, <`http://download.wikimedia.org/nlwiki/20080310/nlwiki-20080310-pages-articles.xml.bz2`> (315,0 MB komprimiert, MD5: `2e7e2fb9a3e2d77a68b32d4dc895c9d8`).
- Der Dump der aktuellen Artikelversionen der norwegischsprachigen Wikipedia (`no`) vom 11. März 2008, <`http://download.wikimedia.org/nowiki/20080311/nowiki-20080311-pages-articles.xml.bz2`> (120,9 MB komprimiert, MD5: `f61d04b1505ba97f4f0ee2bc9484105a`).
- Der Dump der aktuellen Artikelversionen der Wikipedia in einfachem Englisch (`simple`) vom 16. März 2008, <`http://download.wikimedia.org/simplewiki/20080316/simplewiki-20080316-pages-articles.xml.bz2`> (22,3 MB komprimiert, MD5: `9f78f72ffc00b73d817df240806fa806`).
- Der Dump der aktuellen Artikelversionen der plattdeutschen Wikipedia (`nds`) vom 19. März 2008, <`http://download.wikimedia.org/ndswiki/20080319/ndswiki-20080319-pages-articles.xml.bz2`> (7,2 MB komprimiert, MD5: `3c3b30ec70c66a8d9bcee1df75bc9427`).

Die Inhalte der Dumps stehen unter der *GNU Free Documentation License* [GFDL], die Liste der jeweiligen Autoren kann den betreffenden Wikipedia-Seiten entnommen werden.

Thematische Auszüge aus der Wikipedia

Als übersichtliche Datenbasis für Experimente wurden fünf thematische Auszüge aus jeweils vier bis sechs Wikipedias erstellt. Ziel war es, die Themen so zu wählen, dass die Abdeckung in den einzelnen Wikipedias in etwa vergleichbar ist. Unter `/wikidumps/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/wikidumps/`> sind Dumps für die folgenden Themen abgelegt:

`animals`: Seiten zum Thema *Haustier* (bzw. *Nutztier*). Die englischsprachige Wikipedia fehlt hier, da die entsprechende Kategorie nicht hinreichend abgegrenzt ist. Die norwegischsprachige fehlt ebenfalls, da dort keine entsprechende Kategorie existiert.

`color`: Seiten zum Thema *Farbe*. Bei diesem Thema ist die Korrelation der Artikel recht gut, die in den verschiedenen Wikipedias behandelten Konzepte entsprechen einander weitgehend.

`logic`: Seiten zum Thema *Logik*. Welche Konzepte zum Thema Logik gehören, differiert stark zwischen den Projekten.

`mountains`: Seiten zum Thema *Berge*. Hier zeigt sich ein starker geografischer Bias: es werden vor allem Berge behandelt, die sich im Sprachgebiet der betreffenden Wikipedia befinden.

`popes`: Seiten zum Thema *Päpste*. Ähnlich wie bei Farben ergibt sich hier eine gute Abgrenzung und starke Korrelation zwischen den Wikis.

Wie bereits oben erwähnt stehen die aus der Wikipedia stammenden Inhalt unter der *GNU Free Documentation License* [GFDL].

F.2. Datenbank-Dumps

SQL-Dumps von Thesauri für thematische Auszüge

Für die Thesauri, die aus den oben genannten thematischen Auszügen (▹F.1) erzeugt wurden, stehen unter `/sqldumps/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/sqldumps/`> SQL-Dumps zur Verfügung:

`animals`: `wikiword-animals.sql.bz2`, 9.4 MB

`color`: `wikiword-color.sql.bz2`, 4.4 MB

`logic`: `wikiword-logic.sql.bz2`, 19.7 MB

`mountains`: `wikiword-mountains.sql.bz2`, 58.2 MB

`popes`: `wikiword-popes.sql.bz2`, 7.9 MB

Die Konventionen für die Tabellennamen sowie die Struktur der einzelnen Tabellen sind in ▹C beschrieben.

SQL-Dumps für die aus der Wikipedia erstellten Thesauri

Dieselben Daten stehen noch einmal unter <http://aspra27.informatik.uni-leipzig.de/~dkinzler/sqldumps/> zur Verfügung, dort ist jedoch auch die Datei `wikiword-full.sql.bz2` enthalten: sie ist 11 GB groß und enthält die Thesaurus-Daten, die aus den Wikipedias gewonnen wurden. Alle Thesaurusdaten stehen berechtigten Benutzern auch in der Datenbank `dkinzlerdb` auf `aspra5.informatik.uni-leipzig.de` zur Verfügung. Die Konventionen für die Tabellennamen sowie die Struktur der einzelnen Tabellen sind in ▹C beschrieben. In der Datenbank befinden sich zum Beispiel die folgenden Tabellen:

`full_de_*` sind die Tabellen, die den monolingualen Thesaurus enthalten, der aus der deutschsprachigen Wikipedia extrahiert wurde (Datensatz `de` in der Kollektion `full`). Die Tabelle `full_de_meaning` zum Beispiel enthält die Bedeutungsrelation dieses Thesaurus, sie assoziiert Konzepte mit ihren Bezeichnungen.

`full_en_*` sind die Tabellen, die den monolingualen Thesaurus enthalten, der aus der englischsprachigen Wikipedia extrahiert wurde (Datensatz `en` in der Kollektion `full`). Die Tabelle `full_en_broader` zum Beispiel enthält die Subsumtionsrelation dieses Thesaurus, sie setzt allgemeinere mit spezielleren Konzepten in Beziehung.

`full_thesaurus_*` sind die Tabellen, die den multilingualen Thesaurus enthalten, der aus allen monolingualen Thesauri in der Kollektion erstellt wurde (Datensatz `thesaurus` in der Kollektion `full`). Die Tabelle `full_thesaurus_relation` zum Beispiel enthält verschiedene assoziative Beziehungen zwischen Konzepten, zum Beispiel Ähnlichkeit (in den Feldern `langmatch` und `langref`) sowie Verwandtheit (im Feld `bilink`).

`full_top3_*` sind die Tabellen, die den multilingualen Thesaurus enthalten, der aus den Thesauri `en`, `de` und `fr` erstellt wurde (Datensatz `top3` in der Kollektion `full`). Die Tabelle `full_top3_degree` zum Beispiel enthält verschiedene Scores und Rangangaben, insbesondere in Bezug auf Knotengrade.

Hinweis: *Diese Daten werden zum Zweck der Begutachtung der Diplomarbeit vorübergehend bereitgestellt. Sie sind unter Umständen nicht von ausserhalb der Universität Leipzig und/oder nicht für längere Zeit zugänglich.*

F.3. RDF-Dumps

Die Thesauri, die zur Erprobung von WikiWord und für die in ▹IV beschriebene Evaluation erzeugt wurden, stehen auch als RDF-Dateien zur Verfügung. Die Repräsentation der Thesauri als RDF ist in ▹10 beschrieben, die Dateien sind im *Turtle*-Format [Backett (2007)] serialisiert und mit *BZip2* komprimiert.

Die Dateinamen geben wieder, mit welchen Optionen für `ExportRdf` die Dumps erzeugt wurden: Dateien mit der Endung `ww.n3.bz2` wurden mit der Default-Einstellung, also unter Verwendung

des WikiWord-Vokabulars und ohne Cutoff, erzeugt. `ww2.n3.bz2` verwendet einen Cutoff-Wert von 2, `skos.n3.bz2` wurde ohne Cutoff erzeugt und verwendet nur das SKOS-Vokabular.

Das RDF-Schema [W3C:RDFS (2000)] für das WikiWord-Vokabular mit der URI <`http://brightbyte.de/vocab/wikiword`> ist unter `/WikiWord/WikiWord/src/main/wikiword.rdf` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/WikiWord/src/main/wikiword.rdf`> verfügbar.

RDF-Dumps von Thesauri für thematische Auszüge

Für die Thesauri, die aus den oben genannten thematischen Auszügen (▹F.1) erzeugt wurden, stehen unter `/rdfdumps/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/rdfdumps/`> RDF-Dumps zur Verfügung.

RDF-Dumps für aus der Wikipedia erstellten Thesauri

Dieselben Daten stehen noch einmal unter <`http://aspra27.informatik.uni-leipzig.de/~dkinzler/rdfdumps/`> zur Verfügung, dort sind jedoch auch die RDF-Dateien für die Thesauri enthalten, die aus den Wikipedias gewonnen wurden. Die Dateinamen dieser Dumps beginnen mit `full_`, gefolgt vom Namen des Korpus bzw. Thesaurus:

- Monolinguale Thesauri: `full_en_`, `full_de_`, `full_fr_`, `full_nl_`, `full_no_`, `full_nds_` und `full_simple_` sind die Präfixe für die Thesauri, die direkt aus den entsprechenden Wikipedias erstellt wurden.
- Multilinguale Thesauri: `full_thesaurus_` ist der Präfix für den Thesaurus, der aus allen oben genannten lokalen Thesauri erstellt wurde; `full_top3_` ist der Präfix für den Thesaurus aus den drei größten Wikipedias (`en`, `de`, `fr`) und `full_top5_` für den Thesaurus aus den fünf „reifen“ Wikipedias aus der obigen Liste(`en`, `de`, `fr`, `nl`, `no`). Die RDF-Dumps der multilingualen Thesauri sind jeweils zwischen 900 und 1 300 Megabyte groß.

Hinweis: *Diese Daten werden zum Zweck der Begutachtung der Diplomarbeit vorübergehend bereitgestellt. Sie sind unter Umständen nicht von ausserhalb der Universität Leipzig und/oder nicht für längere Zeit zugänglich.*

Beispiele für RDF-Daten

Ein Beispiel für die Darstellung eines Konzeptes, nämlich KARPFEN, mit Hilfe von RDF/SKOS und dem WikiWord-Vokabular:

```
@base <http://brightbyte.de/vocab/wikiword/concept/from/http://de.wikipedia.org/wiki/> .
@prefix : <http://brightbyte.de/vocab/wikiword/concept/from/http://de.wikipedia.org/wiki/> .
@prefix dewiki: <http://de.wikipedia.org/wiki/> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
```

```
@prefix ww: <http://brightbyte.de/vocab/wikiword#> .
@prefix wwct: <http://brightbyte.de/vocab/wikiword/concept-type#> .

:Karpfen rdf:type skos:Concept ;
        skos:altLabel "Cyprinus carpio"@de ;
        skos:altLabel "Karpfen"@de ;
        skos:altLabel "Karpfens"@de ;
        skos:altLabel "Lederkarpfen"@de ;
        skos:altLabel "Nacktkarpfen"@de ;
        skos:altLabel "Schuppenkarpfen"@de ;
        skos:altLabel "Spiegelkarpfen"@de ;
        skos:altLabel "Wildkarpfen"@de ;
        skos:altLabel "Zeilkarpfen"@de ;
        skos:definition "Der Karpfen ist eine Fischart aus der Familie der Karpfenfische."@de ;
        skos:definition dewiki:Karpfen ;
        skos:inScheme <http://brightbyte.de/vocab/wikiword/dataset/*/animals:de> ;
        skos:broader :Haustier ;
        skos:broader :Karpfenartige ;
        skos:broader :Speisefisch_und_Meeresfrucht ;
        ww:displayLabel "Karpfen"@de ;
        ww:idfRank 16210 ;
        ww:idfScore 7.77359446736019 ;
        ww:inDegree 8.0 ;
        ww:inRank 772 ;
        ww:lhsRank 16204 ;
        ww:lhsScore 0.293862558415189 ;
        ww:linkDegree 77.0 ;
        ww:linkRank 111 ;
        ww:outDegree 69.0 ;
        ww:outRank 82 ;
        ww:similar :Koi ;
        ww:type wwct:LIFEFORM .
```

F.4. Bewertungsdaten und Statistiken

Zur Evaluation von WikiWord wurden verschiedene Stichproben (*Batches*) gewählt und dann einzelne Eigenschaften und Aspekte der Elemente in den Stichproben manuell bewertet. Diese Bewertungen wurden für die statistische Auswertung in *Survey*-Tabellen gespeichert. Diese stehen in `/sqldumps/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/sqldumps/`> als Tab-separierte Liste (*TSV*) mit Namen der Form `*_survey.tsv` zur Verfügung. Die einzelnen Stichproben und Bewertungen sind im Folgenden beschrieben.

Alle Survey-Tabellen haben die folgenden Felder:

`timestamp`: Zeitpunkt, zu dem die Bewertung vorgenommen wurde.

`batch`: Name der Stichprobe oder des Satzes von Bewertungen.

`seq`: Sequenznummer der Bewertung innerhalb des Satzes.

`property`: Bewertete Eigenschaft.

`value`: Bewerteter Wert, wie von WikiWord vorgegeben.

assessment: Bewertung. Die möglichen Werte sind je nach bewerteter Eigenschaft unterschiedlich.

Die einzelnen Survey-Tabellen können zusätzliche Felder haben, die die Einträge in der Bewertungstabelle mit Einträgen in der Datenstruktur von WikiWord verbinden.

Bewertung der Konzepteigenschaften: concept_survey

Die Tabelle concept_survey dient der Erhebung von Bewertungen von Konzepteigenschaften, wie sie in ▸12 besprochen wurden. Sie hat die folgenden zusätzlichen Felder:

concept: Die ID des bewerteten Konzeptes, als Verweis in die Tabelle concept (▸C.2).

concept_name: Der Name des bewerteten Konzeptes, als Verweis in die Tabelle concept.

In dieser Tabelle befinden sich für lokale Datensätze Bewertungen für die folgenden Eigenschaften:

terms: Bewertung der Termzuordnung, wie in ▸12.5 besprochen. Die möglichen Bewertungen sind dort aufgezählt.

type: Bewertung der Termzuordnung, wie in ▸12.1 besprochen. Die möglichen Bewertungen sind dort aufgezählt.

gloss: Bewertung der Termzuordnung, wie in ▸12.6 besprochen. Die möglichen Bewertungen sind dort aufgezählt.

broader und narrower: Bewertung der Termzuordnung, wie in ▸12.2 besprochen. Die möglichen Bewertungen sind dort aufgezählt.

related und similar: Bewertung der Termzuordnung, wie in ▸12.4 und ▸12.3 besprochen. Die möglichen Bewertungen sind dort aufgezählt.

Konkret sind diese Informationen in den Dateien full_de_concept_survey.tsv (für den deutschsprachigen Thesaurus) und full_en_concept_survey.tsv (für den englischsprachigen Thesaurus) enthalten, die unter der oben genannten URL verfügbar sind.

Für globale Datensätze dagegen wird nur eine Eigenschaft von Konzepten bewertet:

merging: Bewertung der Konzeptzusammenfassung, wie in ▸11.3 besprochen. Die möglichen Bewertungen sind:

- **OPTIMAL:** Bestmögliche Kombination von sprachspezifischen Konzepten.
- **FUZZY:** Bestmögliche Kombination von sprachspezifischen Konzepten, enthält aber Konzepte, die nur „fast“ zu den anderen passen.

LONE: Das globale Konzept besteht nur aus einem einzigen lokalen Konzept, und es gibt auch keine Kandidaten für eine Vereinigung (also keine ähnlichen Konzepte).

LACKING: Die Kombination von sprachspezifischen Konzepten ist korrekt, es wurden aber ähnliche Konzepte übergangen, die mit aufgenommen werden sollten.

BAD: Die Menge von sprachspezifischen Konzepten enthält mindestens ein Konzept, das falsch zugeordnet wurde und nicht zu den anderen passt.

Konkret sind diese Informationen in der Datei `full_thesaurus_concept_survey.tsv` enthalten, die unter der oben genannten URL verfügbar ist.

Bewertung der Analyse von Begriffsklärungsseiten: `resource_survey`

Die Tabelle `concept_survey` dient der Erhebung von Bewertungen von Ressourceneigenschaften, wie sie in ▹13 besprochen wurden. Sie hat das folgende zusätzliche Feld:

`resource_name`: Der Name der bewerteten Wiki-Seite, als Verweis in die Tabelle `resource` (▹C.2).

Für die Evaluation der Analyse von Begriffsklärungsseiten wurde die Eigenschaft `link` bewertet, die angibt, ob ein Wiki-Link auf der Begriffsklärungsseite auf eine der Bedeutungen der Begriffsklärung zeigt und ob dies korrekt erkannt wurde. Die möglichen Bewertungen sind in ▹13 definiert.

Konkret sind diese Informationen in den Dateien `full_de_resource_survey.tsv` (für den deutschsprachigen Thesaurus) und `full_en_resource_survey.tsv` (für den englischsprachigen Thesaurus) enthalten, die unter `/sqldumps/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/sqldumps/`> verfügbar sind.

Bewertung der automatischen Disambiguierung und Bestimmung der semantischen Verwandtheit

Die Tabelle `relatedness_survey` dient der Bewertung von Bedeutungszuweisungen und Verwandtheitswerten, wie sie in ▹14 besprochen wurden. Sie hat die folgenden zusätzlichen Felder:

`term_a` und `term_b` sind die beiden Terme, deren Verwandtheit bestimmt werden soll.

`concept_a` und `concept_b` sind die Konzepte, die den beiden obigen Termen zugeordnet wurden, manuell oder automatisch. Darf auch `NULL` sein, falls ein Konzept nicht gefunden wurde.

is_nn ist 1, falls beide Terme Substantive sind. Paare von Substantiven werden bei der Evaluation (auch) getrennt betrachtet.

found ist 1, falls für beide Terme Konzepte gefunden wurden.

relatedness ist ein Wert für die Verwandtheit der beiden Terme bzw. der beiden Konzepte, nach irgendeinem Bewertungsverfahren, auf irgendeiner Skala.

score ist eine normierte Version des Werte im Feld `relatedness`: $r' := \frac{r-\bar{r}}{/}\sigma_r$. Vom Originalwert wird also der Mittelwert abgezogen und das Ergebnis durch die Standardabweichung geteilt. So ergeben sich Werte, die unabhängig von der Bewertungsskala und einer eventuellen systematischen Verzerrung direkt verglichen werden können, zumindest bezüglich ihres Vorzeichens, das angibt, ob die Verwandtheit zweier Terme bzw. Konzepte für „überdurchschnittlich“ oder „unterdurchschnittlich“ befunden wurde.

Die Evaluation des in ▸14.1 vorgestellten Verfahrens zur automatischen Disambiguierung und Bestimmung der semantischen Verwandtheit wurde auf der Basis eines Vergleichsdatensatzes durchgeführt, der von *Zesch* und *Gurevych* erstellt wurde: Der Datensatz `ZG222`[1] [Zesch und Gurevych (2006)] enthält 222 Wortpaare, die von 24 Probanden in Hinblick auf ihre semantische Verwandtheit auf einer Skala von 0 bis 4 bewertet wurden.

Die folgende Tabelle zeigt, was für Bewertungssätze (*Batches*) auf dieser Basis erstellt wurden:

ZG222 enthält die Originaldaten aus [Zesch und Gurevych (2006)], lediglich das `score`-Feld wurde berechnet. Dieser Satz enthält keine Bedeutungszuweisung, `concept_a` und `concept_b` sind leer.

ZG222-manual enthält manuelle Konzeptzuweisungen zu den Termen in `ZG222`, vorgenommen durch den Autor. Diese Zuweisungen sollten möglichst denen ähnlich sein, die von den Probanden bei der Erstellung von `ZG222` implizit vorgenommen wurden. Ob dies gelungen ist, ist allerdings kaum zu prüfen. Dieser Umstand wurde in ▸14.1 noch näher betrachtet.

ZG222-manual-concepts-* sind Datensätze, die die semantische Verwandtheit der manuell gewählten Konzepte enthalten, berechnet mit unterschiedlichen Parametern (siehe unten).

ZG222-coherence-* sind Datensätze, in denen die Bedeutungszuweisung aufgrund einer Maximierung der Verwandtheit der Konzepte (*Kohärenz*), bestimmt unter Verwendung unterschiedlicher Parameter, durchgeführt wurde (siehe ▸14.2).

ZG222-popularity-* sind Datensätze, in denen immer die meist verwendete Bedeutung für jeden Term verwendet wurde, und dann die Verwandtheit der Konzepte unter Verwendung unterschiedlicher Parameter bestimmt wurde.

Die Parameter, die zur Bestimmung der semantischen Verwandtheit von Konzepten verwendet wurden, sind in den Namen der Datensätze wiedergegeben:

[1] Verfügbar unter <`http://www.ukp.tu-darmstadt.de/data/semRelDatasets`>.

***-cosine** sind Datensätze mit Verwandtheitswerten, die unter Verwendung des Kosinus-Maßes zum Vergleich der Featurevektoren der beiden Konzepte berechnet wurden.

***-tanimoto** sind Datensätze mit Verwandtheitswerten, die unter Verwendung einer gewichteten Tanimoto-Ähnlichkeit (▸14.1) zum Vergleich der Featurevektoren der beiden Konzepte berechnet wurden.

-weighted- sind Datensätze, bei denen die Einträge in den Featurevektoren für die Bestimmung der Verwandtheit nach IDF- bzw. LHS-Werten der Konzepte gewichtet wurden. Die Ergebnisse diese Verfahrens sind durchweg schlechter als die, die mit ungewichteten Featurevektoren erzielt wurden. Daher wurden diese Daten in ▸14.1 nicht weiter erwähnt.

-unweighted- sind Datensätze, bei denen die Einträge in den Featurevektoren für die Bestimmung der Verwandtheit ohne Gewichtung vorgenommen wurden.

-pop=\textit{p}- sind Datensätze, bei denen bei der Bestimmung der Verwandtheit der Konzepte die Frequenz berücksichtigt wurde, mit der der betreffende Term für dieses Konzept verwendet wurde. *p* ist also der *Popularitäts-Bias*, wie in ▸14.2 beschrieben.

Dateien mit verschiedenen Kombinationen dieser Eigenschaften stehen in `/sqldumps/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/sqldumps/`> zur Verfügung.

F.5. Daten für Grafiken und Tabellen

Die Daten, die für die Erstellung von Grafiken und Tabellen in dieser Arbeit (insbesondere im Evaluationsteil) verwendet wurden, stehen unter `/WikiWord/WikiWordThesis/Diplomarbeit/Daten/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/WikiWordThesis/Diplomarbeit/Daten/`> zur Verfügung. Die folgenden Datenformate wurden verwendet:

R: Skriptdatei für GNU-R [GNU-R]. Solche Skripte werden verwendet, um eine Grafik im PDF-Format zu erzeugen, die im Allgemeinen bis auf die Dateiendung denselben Namen trägt. Als Datenbasis wird eine `*.data`-Datei verwendet.

data: Tabellen im TSV-Format (*Tab-Separated Values*), also als Textdateien, wobei Zeilen der Tabelle als Textzeile in der Datei gespeichert sind und die einzelnen Werte durch Tabulatoren (ASCII-Code 0x09) getrennt sind. Texte sind in UTF-8 kodiert.

tsv: Ebenfalls Tabellen im TSV-Format, wie oben beschrieben. Allerdings enthalten diese Dateien zum Teil kein „reines“ TSV: sie können TeX-Markup enthalten sowie Hinweise auf horizontale Trennlinien in der Form von Zeilen, die nur eine Folge von „−−“ enthalten. Solche Dateien werden mit Hilfe des Skriptes `tsv2tex`, das beim Rendern dieses Dokumentes automatisch angesprochen wird, in TeX-Dateien umgewandelt.

G. Quellcode

Dieses Kapitel bietet einen Überblick über den Quellcode von WikiWord. Es wird beschrieben, wie das Programm strukturiert ist, welche Bibliotheken verwendet werden, wo der Quelltext erhältlich ist sowie wie er verwendet werden kann.

G.1. WikiWord Packages

Dieser Abschnitt bietet eine kurze Übersicht über die Java-Packages, aus denen WikiWord besteht. Zunächst seien die Packages für den Kern von WikiWord genannt:

`de.brightbyte.wikiword` definiert Klassen wie z. B. `Corpus` und `Namespace`, die WikiWords Kern-Konzepte modellieren, sowie Hilfsklassen wie `CliApp` (die Basis für die Kommandozeilenschnittstelle von WikiWord, siehe ▹7.1) und `TweakSet` (für Konfigurationsparameter, siehe ▹B.5). Das Package enthält außerdem Property-Dateien mit den vorgegebenen Konzepttypen (`ConceptTypes.properties`), Sprachcodes (`Languages.properties`) und Namensräumen (`Namespaces.properties`).

`de.brightbyte.wikiword.disambig` enthält Klassen zur automatischen Sinnzuweisung an Terme im Kontext (Disambiguierung) sowie die dafür benötigten Klassen zur Bestimmung von semantischer Verwandtheit zwischen Konzepten (▹14).

`de.brightbyte.wikiword.model` enthält Modell-Klassen, die Konzepte und Ressourcen sowie Referenzen auf Terme und Konzepte repräsentieren. Instanzen dieser Klassen dienen als *Transferobjekte* bei Abfrage von Thesaurusdaten aus der Datenbank (DAO-Schicht, siehe ▹C.3).

`de.brightbyte.wikiword.query` enthält die Kommandozeilenschnittstellen und Hilfsklassen für die Abfrage von Thesaurusdaten. Diese sind vor allem für Test und Evaluation von WikiWord nützlich.

`de.brightbyte.wikiword.rdf` enthält die Kommandozeilenschnittstellen und Hilfsklassen für den Datenexport nach RDF/SKOS (siehe ▹10).

`de.brightbyte.wikiword.schema` enthält Hilfsklassen, die die Datenbank-Schemata für die einzelnen Store-Klassen der DAO-Schicht definieren: Jede Schema-Klasse stellt einen Satz von Datenbanktabellen und den zugehörigen Datenfeldern bereit (▹C.2). Sie definiert außerdem die Logik für eine Konsistenzprüfung der Daten sowie für die Erhebung einer einfachen Statistik über die enthaltenen Daten. Diese Klassen werden von den Datenzugriffsobjekten (DAOs) in `de.brightbyte.wikiword.store` und `de.brightbyte.wikiword.store.builder` benutzt.

`de.brightbyte.wikiword.store` enthält die Definition der einzelnen Datenzugriffsobjekte (DAOs) für die Abfrage und Nutzung der Thesaurusdaten (siehe ▹9.1 und ▹C.3).

Dieser Kern von WikiWord umfasst 134 Klassen und über 11 000 Zeilen Programmcode.

Die Logik für den Import von Daten bzw. den Aufbau des Thesaurus ist in den folgenden Packages implementiert:

de.brightbyte.wikiword.analyzer enthält die Klassen, die die Analyse von Wiki-Text implementieren, also zum Aufbau des Ressourcenmodells aus den Rohdaten (siehe ▹8).

de.brightbyte.wikiword.builder enthält die Kommandozeilenschnittstelle für den Aufbau des Thesaurus sowie Hilfsklassen der Import-Architektur (siehe ▹B.1 und ▹7.1)

de.brightbyte.wikiword.store.builder enthält die Definition der einzelnen Datenzugriffsobjekte (DAOs) für den Aufbau und die Konsolidierung der Thesaurusdaten (siehe ▹9.2 und ▹C.4).

de.brightbyte.wikiword.wikis enthält die Konfiguration für die einzelnen Wikis, also Wissen über deren spezielle Einrichtung und Konventionen, sowie sprachspezifisches Wissen. Diese Informationen werden für die Analyse des Wiki-Textes verwendet (▹8).

Die Logik für den Import von Daten umfasst 121 Klassen und über 14 000 Zeilen Programmcode.

Zusätzliche Logik für Experimente mit und für die Evaluation von WikiWord (▹IV) befindet sich in den folgenden Packages:

de.brightbyte.wikiword.eval enthält die Kommandozeilenschnittstelle für die Evaluation von WikiWord.

de.brightbyte.wikiword.schema enthält die Schema-Definition und die DAO-Schicht für die Evaluation, also insbesondere für die Speicherung der in Experimenten erhobenen Daten und deren statistische Auswertung.

Die Funktionen für die Evaluation umfasst 43 Klassen und über 5 000 Zeilen Programmcode.

Der Programmcode von WikiWord steht unter der *GNU General Public License* [GPL] und kann frei genutzt werden (siehe Lizenz, S. 207). Für eine Aufstellung der verwendeten Programmbibliotheken siehe ▹G.2. Für die URLs, unter denen WikiWord verfügbar ist, siehe ▹G.3. Eine kurze Anweisung zum Kompilieren von WikiWord findet sich in ▹G.4.

G.2. Verwendete Bibliotheken

WikiWord verwendet neben den von Java bereitgestellten Standardklassen eine Anzahl von Bibliotheken, die Standard-Aufgaben übernehmen. Diese sind:

JUnit ist ein *Unit-Testing*-Framework für Java (<`http://www.junit.org/`>). Es wird nur zur Kompilierung und Ausführung der Testklassen von WikiWord benötigt. *JUnit* steht unter der *Common Public License* [CPL].

Connector/J ist der offizielle JDBC-Treiber für MySQL (<http://www.mysql.com/products/connector/j/>). Er wird nur zur Laufzeit für den Zugriff auf MySQL-Datenbanken benötigt. *Connector/J* steht unter der *GNU General Public License* [GPL].

MWDumper ist ein Programm bzw. eine Bibliothek zum Lesen und Schreiben von MediaWiki-Dumps [MW:DUMPER, MW:XML]. Er wird für den Import von MediaWiki-Dumps in WikiWord benötigt. *MWDumper* wird von der *Wikimedia Foundation* entwickelt und steht unter der *GNU General Public License* [GPL].

MWDumper benötigt seinerseits einige zusätzliche Bibliotheken:

swing-layout ist eine Erweiterung der Layout-Klassen *Swing*, Javas Bibliothek für grafische Benutzeroberflächen (<https://swing-layout.dev.java.net/>). Diese Bibliothek wird für die Kompilierung von *MWDumper* benötigt und im interaktiven Modus von *MWDumper* eingesetzt. Dieser wird jedoch im Kontext von WikiWord nicht verwendet, so dass diese Bibliothek zur Laufzeit nicht unbedingt benötigt wird. *Swing-layout* steht unter der *GNU Lesser General Public License* [LGPL].

Der *PostgreSQL JDBC Driver* ist der offizielle JDBC-Treiber für PostgreSQL (<http://jdbc.postgresql.org/>). Er wird von *MWDumper* lediglich zur Laufzeit benötigt, um auf *PostgreSQL*-Datenbanken zuzugreifen – was im Kontext von WikiWord nicht notwendig ist. Der *PostgreSQL JDBC Driver* steht unter der *BSD License* [BSDL].

Xerces ist eine Bibliothek für die Verarbeitung von XML (<http://xerces.apache.org/xerces2-j/>). Sie wird von *MWDumper* zur Laufzeit zur schnelleren Verarbeitung der MediaWiki-Dumps verwendet. Fehlt diese Bibliothek, so wird Javas integrierter XML-Parser verwendet, der etwas langsamer ist. *Xerces* wird von der *Apache Software Foundation* entwickelt und steht unter der *Apache Software License* [ASL].

XML-Commons ist eine Hilfsbibliothek zur Verarbeitung von XML (<http://xml.apache.org/commons/>). Wie *Apache Xerces* wird sie von *MWDumper* zur Laufzeit zur Verarbeitung von XML verwendet, wird aber nicht unbedingt benötigt. *XML-Commons* wird von der *Apache Software Foundation* entwickelt und steht unter der *Apache Software License* [ASL].

WikiWord verwendet einige Bibliotheken, die ebenfalls vom Autor entwickelt wurden, aber unabhängig von und zeitlich vor dieser Diplomarbeit. Sie dienen allgemeinen Aufgaben und enthalten keinen Code, der sich auf die Verarbeitung von Wiki-Text oder den Aufbau eines Thesaurus bezieht.

BrightByteUtil enthält diverse Hilfsklassen, unter Anderem zur Arbeit mit Strings, Collections, XML, HTML sowie für die Verarbeitung von dünnbesetzten Matrizen und Vektoren.

BrightByteDB enthält Hilfsklassen für die Arbeit mit relationalen Datenbanken, insbesondere MySQL.

BrightByteRDF enthält ein Framework zur Erzeugung und Verarbeitung von RDF-Dateien.

Diese Bibliotheken stehen unter der *GNU Lesser General Public License* [LGPL]. Der Quellcode sowie verschiedene Archivdateien stehen unter <http://brightbyte.de/TK/DA/> bereit.

Diese Abhängigkeiten sind über die Datei `pom.xml` in jedem Projekt definiert und können so mit Hilfe von *Apache Maven* [MVN] automatisch aufgelöst werden (siehe auch ▸G.4).

G.3. Bündel

Der Quellcode von WikiWord steht an den folgenden Orten zur Verfügung:

- Das Verzeichnis `/WikiWord/WikiWord/` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWord/WikiWord/> enthält die Quellen für den Kern von WikiWord. Diese Dateien befinden sich auch in einem Archiv unter `/WikiWord-0.1-src.tar.gz` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWord-0.1-src.tar.gz>.
- Das Verzeichnis `/WikiWord/WikiWordBuilder/` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWord/WikiWordBuilder/> enthält die Quellen für das System zum Aufbau von Thesauri. Diese Dateien befinden sich auch in einem Archiv unter `/WikiWordBuilder-0.1-src.tar.gz` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWordBuilder-0.1-src.tar.gz>.
- Das Verzeichnis `/WikiWord/WikiWordEval/` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWord/WikiWordEval/> enthält die Quellen für die Logik zur Evaluation von WikiWord. Diese Dateien befinden sich auch in einem Archiv unter `/WikiWordEval-0.1-src.tar.gz` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWordEval-0.1-src.tar.gz>.

Dabei ist zu beachten, dass `WikiWordBuilder` und `WikiWordEval` beide vom Kern `WikiWord`, aber nicht voneinander abhängig sind.

Die drei Komponenten von WikiWord stehen auch in bereits kompilierter Form zur Verfügung:

- Die Datei `/bin/WikiWord-0.1/lib/WikiWord-0.1.jar` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/bin/WikiWord-0.1/lib/WikiWord-0.1.jar> enthält den Kern von WikiWord als Programmbibliothek. Ein Archiv, das alle benötigten Bibliotheken und Hilfsdateien enthält, befindet sich unter `/WikiWord-0.1-bin-dep.tar.gz` auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWord-0.1-bin-dep.tar.gz>.

- Die Datei /bin/WikiWordBuilder-0.1/lib/WikiWordBuilder-0.1.jar auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/bin/WikiWordBuilder-0.1/lib/WikiWordBuilder-0.1.jar> enthält das System zum Aufbau von Thesauri als Programmbibliothek. Ein Archiv, das alle benötigten Bibliotheken und Hilfsdateien enthält, befindet sich unter /WikiWordBuilder-0.1-bin-dep.tar.gz auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWordBuilder-0.1-bin-dep.tar.gz>.
- Die Datei /bin/WikiWordEval-0.1/lib/WikiWordEval-0.1.jar auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/bin/WikiWordEval-0.1/lib/WikiWordEval-0.1.jar> enthält die Logik zur Evaluation von WikiWord als Programmbibliothek. Ein Archiv, das alle benötigten Bibliotheken und Hilfsdateien enthält, befindet sich unter /WikiWordEval-0.1-bin-dep.tar.gz auf der beiliegenden CD oder online unter <http://brightbyte.de/DA/WikiWordEval-0.1-bin-dep.tar.gz>.

G.4. Kompilieren

WikiWord ist in Java implementiert. Als Build-System, um aus dem Quellcode Programme und Bibliotheken zur erzeugen, wurde *Apache Maven* [MVN] verwendet. Dieses System hat insbesondere den Vorzug, dass es automatisch Abhängigkeiten zwischen Programmbibliotheken verwaltet und auflöst. Allerdings ist das nur möglich, wenn alle benötigten Bibliotheken in einem *Repository* verfügbar sind, entweder lokal oder online. Die BrightByte- und WikiWord-Bibliotheken sind in keinem Online-Repository abgelegt und müssen daher explizit lokal installiert werden, um mit Maven genutzt werden zu können. Um eine Komponente von WikiWord (oder ein anderes Maven-Projekt) zu kompilieren, zu testen und lokal zu installieren, dient der folgende Befehl, auszuführen im Basisverzeichnis des Projektes, das auch die Datei pom.xml enthält:

```
mvn install
```

Damit steht die entsprechende Bibliothek (*JAR-Datei*) für die Benutzung durch andere Programme im Maven-Repository bereit. Des Weiteren stehen die folgenden Kommandos zur Verfügung:

clean löscht alle temporären Daten und erzwingt eine vollständige Neukompilierung aller Dateien.

compile kompiliert die Programmdateien.

test führt alle *Uni-Tests* aus.

Um für die einzelnen Module bzw. Bibliotheken Bündel zu erzeugen, eignet sich das folgende Kommando:

```
mvn assembly:assembly
```

Dieser Befehl ist ebenfalls im Basisverzeichnis des betreffenden Projektes auszuführen. Die erzeugten Bündel werden im Verzeichnis `target` abgelegt. Die Bündel mit der Endung `*-bin-dep.tar.gz` enthalten vollständige Installationen mit allen benötigten Bibliotheken und Hilfsdateien.

G.5. TeX-Quellen

Die vorliegende Diplomarbeit wurde mit Hilfe des Textsatzsystems LaTeX geschrieben, der Quelltext ist unter `/WikiWord/WikiWordThesis/Diplomarbeit/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/WikiWordThesis/Diplomarbeit/`> oder als Archiv unter `/WikiWordThesis-tex.tar.gz` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWordThesis-tex.tar.gz`> verfügbar.

Als Build-System für die Erzeugung einer druckfertigen PDF-Datei aus den TeX-Dateien und anderen Quellen wurde *Apache Ant* [ANT] verwendet. Das Build-Skript für Ant, `build.xml`, definiert insbesondere die folgenden Kommandos (*targets*):

`clean` löscht alle temporären Daten und erzwingt eine vollständige Neuerzeugung aller Dateien.

`build` erzeugt aus den Quelldateien eine druckfertige PDF-Datei. Diese wird unter `target/WikiWord.pdf` abgelegt.

Für den Build-Prozess werden diverse externe Programme sowie zusätzliche Module für TeX und MetaPost benötigt. Eine Aufstellung befindet sich in der Datei `BUILDING`, auch zu finden unter `/WikiWord/WikiWordThesis/BUILDING` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/WikiWordThesis/BUILDING`>.

Das Build-Skript wurde für die Verwendung unter Linux entwickelt, für eine Ausführung unter Windows sind vermutlich diverse Anpassungen notwendig. Die Version von LaTeX, die für die Erstellung dieser Arbeit verwendet wurde, ist *LaTeX2e* bzw. *pdfeTeX 3.141592*, wie von *tetex* bereitgestellt. Soll eine andere Implementation verwendet werden, können unter Umständen kleine Anpassungen notwendig sein.

Der Text dieser Diplomarbeit steht unter der *GNU Free Documentation License* [GFDL] und kann frei genutzt werden (siehe Lizenz, S. 207).

Literaturverzeichnis

Hinweis: Es wurden sowohl Print- wie Online-Ressourcen berücksichtigt. Insebsondere ist eine Vielzahl von Wikipedia-Seiten zitiert, die Wikipedia-Konventionen erklären, die für diese Arbeit wichtig sind.

ANT : *Apache Ant*. – URL http://ant.apache.org/

ASL : *Apache License, Version 2.0*. – URL http://www.apache.org/licenses/LICENSE-2.0

van Assem u. a. 2006 Assem, Mark van ; Malaisé, Véronique ; Miles, Alistair ; Schreiber, Guus: A Method to Convert Thesauri to SKOS. In: *The Semantic Web: Research and Applications* (2006), S. 95–109. – URL http://dx.doi.org/10.1007/11762256_10

Auer und Lehmann 2007 Auer, Sören ; Lehmann, Jens: What have Innsbruck and Leipzig in common? Extracting Semantics from Wiki Content. In: *Proc. 4th European Semantic Web Conf* (2007), S. 503–517. – URL http://dx.doi.org/10.1007/978-3-540-72667-8_36

Auer u. a. 2007 Auer, Sören ; Bizer, C. ; Kobilarov, G. ; Lehmann, Jens ; Cyganiak, R. ; Ives, Z.: DBpedia: A Nucleus for a Web of Open Data. In: *6th Int'l Semantic Web Conference, Busan, Korea, Nov* (2007)

Backett 2007 Backett, David: *Turtle – Terse RDF Triple Language*. 2007. – URL http://www.dajobe.org/2004/01/turtle/

Barabasi und Albert 1999 Barabasi, A. L. ; Albert, R.: Emergence of scaling in random networks. In: *Science* 286 (1999), Oktober, Nr. 5439, S. 509–512. – URL http://view.ncbi.nlm.nih.gov/pubmed/10521342. – ISSN 0036-8075

Bellomi und Bonato 2005a Bellomi, F. ; Bonato, R.: Lexical authorities in an encyclopedic corpus: a case study with wikipedia. In: *International Colloquium on 'Word structure and lexical systems: models and applications'*, URL http://www.fran.it/blog/2005/01/lexical-authorities-in-encyclopedic.html, 2005

Bellomi und Bonato 2005b Bellomi, F ; Bonato, R: Network Analysis for Wikipedia. In: *Proceedings of Wikimania 2005* (2005). – URL http://www.fran.it/articles/wikimania_bellomi_bonato.pdf

Berners-Lee 1998 Berners-Lee, Tim: *Cool URIs don't change*. 1998. – URL http://www.w3.org/Provider/Style/URI.html

Biemann u. a. 2004 Biemann, C. ; Bordag, S. ; Heyer, G. ; Quasthoff, U. ; Wolff, C.: Language-independent Methods for Compiling Monolingual Lexical Data. In: *Proceedings of CicLING*, Springer, 2004, S. 215–228

Black 2006 Black, J.: Creating a common ground for URI meaning using socially constructed web sites. In: *Proceedings of Identity, Reference, and the Web Workshop (IMW06)*, URL http://www.ibiblio.org/hhalpin/irw2006/jblack.pdf, 2006

BSDL : *BSD License*. – URL http://www.opensource.org/licenses/bsd-license.php

Buchanan und Feigenbaum 1982 Buchanan, B. G. ; Feigenbaum, E. A.: Forward. In: *DTIC Research Report* (1982), Nr. ADF650104

Bunescu und Pasca 2006 Bunescu, Razvan ; Pasca, Marius: Using Encyclopedic Knowledge for Named Entity Disambiguation. In: *11th Conference of the European Chapter of the Association for Computational Linguistics*, URL http://www.cs.utexas.edu/~ml/publication/paper.cgi?paper=encyc-eacl-06.ps.gz, April 2006, S. 9–16

Capocci u. a. 2006 Capocci, A. ; Servedio, V. D. P. ; Colaiori, F. ; Buriol, L. S. ; Donato, D. ; Leonardi, S. ; Caldarelli, G.: *Preferential attachment in the growth of social networks: the case of Wikipedia*. Februar 2006. – URL http://arxiv.org/abs/physics/0602026

CCS : *Computing Classification System*. – URL http://www.acm.org/class/

Chan 1986 Chan, L. M.: Library of Congress Classification as an Online Retrieval Tool: Potentials and Limitations. In: *Information Technology and Libraries* 5 (1986), Nr. 3, S. 181–92

Coulter u. a. 1998 Coulter, N. ; French, J. ; Glinert, E. ; Horton, T. ; Mead, N. ; Rada, R. ; Ralston, A. ; Rodkin, C. ; Rous, B. ; and, A. T.: Computing Classification System 1998: Current Status and Future Maintenance Report of the CCS Update Committee. In: *Computing Reviews* (1998), S. 1

CPL : *Common Public License - v 1.0*. – URL http://www.opensource.org/licenses/cpl1.0.txt

Craswell u. a. 2001 Craswell, N. ; Hawking, D. ; Robertson, S.: Effective Site Finding using Link Anchor Information. In: *SIGIR'01*, URL http://pigfish.vic.cmis.csiro.au/~nickc//pubs/sigir01.pdf, 2001

Cucerzan 2007 Cucerzan, S.: Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In: *EMNLP 2007: Empirical Methods in Natural Language Processing, June 28-30, 2007, Prague, Czech Republic* (2007)

CYC : *Cyc*. – URL http://www.cyc.com/

Dakka und Cucerzan 2008 Dakka, Wisam ; Cucerzan, Silviu: Augmenting Wikipedia with Named Entity Tags. In: *IJCNLP*, URL http://research.microsoft.com/users/silviu/Papers/ijcnlp08.pdf, 2008

Davulcu u. a. 2006 Davulcu, H. ; Nguyen, H. V. ; Ramachandran, V.: Boosting Item Findability: Bridging the Semantic Gap Between Search Phrases and Item Information. In: *International Journal of Intelligent Information Technologies* 2 (2006), Nr. 3, S. 1–20. – URL http://www.public.asu.edu/~hdavulcu/ICEIS05.pdf

DBpedia : *DBpedia*. – URL http://dbpedia.org/

DCME 2008 : *Dublin Core Metadata Element Set, Version 1.1*. 2008. – URL http://dublincore.org/documents/dces/

DDC : *Dewey Decimal Classification*. – URL http://www.oclc.org/dewey/

Deerwester u. a. 1990 Deerwester, S. ; Dumais, S. T. ; Furnas, G. W. ; Landauer, T. K. ; Harshman, R.: Indexing by latent semantic analysis. In: *Journal of the American Society for Information Science* 41 (1990), Nr. 6, S. 391–407

Dewey 1965 Dewey, M.: Dewey; Decimal classification and relative index. In: *New York; Lake Placid Club Education Foundation, 1965, 2 s..(2480) p. Tablas* (1965)

Eiron und Mccurley 2003 EIRON, N. ; MCCURLEY, K.: *Analysis of anchor text for web search*. 2003. – URL http://www.mccurley.org/papers/anchor.pdf

Evert 2004 EVERT, S.: A simple LNRE model for random character sequences. In: *Proceedings of JADT* (2004), S. 411–422

FCW : *Definition of Free Cultural Works v1.0*. – URL http://freedomdefined.org/Definition

Fellbaum 1998 FELLBAUM, C.: *WordNet: an electronic lexical database*. Cambridge, MA : MIT Press, 1998

Fiebig 2005 FIEBIG, H.: *Wikipedia - Das Buch*. Berlin : Zenodot Verl.-Ges., 2005. – ISBN 3-86640-001-2

Freebase : *Freebase*. – URL http://www.freebase.com/

Gabrilovich und Markovitch 2006 GABRILOVICH, E. ; MARKOVITCH, S.: Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge. In: *Proceedings of the Twenty-First National Conference on Artificial Intelligence, Boston, MA* (2006)

Gabrilovich und Markovitch 2007 GABRILOVICH, E. ; MARKOVITCH, S.: Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In: *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (2007), S. 6–12

Gabrilovich 2006 GABRILOVICH, Evgeniy: *Feature Generation for Textual Information Using World Knowledge*, The Technion - Israel Institute of Technology, Dissertation, 2006. – URL http://www.hpl.hp.com/techreports/2007/HPL-2007-182.pdf.

Garshol 2004 GARSHOL, Lars M.: Metadata? Thesauri? Taxonomies? Topic Maps!: Making sense of it all. In: *Journal of Information Science* 30 (2004), Nr. 4, S. 378–391. – URL http://jis.sagepub.com/cgi/content/abstract/30/4/378

GFDL : *GNU Free Documentation License*. – URL http://www.gnu.org/licenses/fdl.html

Giles 2005 GILES, Jim: Internet encyclopaedias go head to head. In: *Nature* 438 (2005), S. 900–901. – URL http://www.nature.com/nature/journal/v438/n7070/pdf/438900a.pdf

GNU-R : *GNU-R*. – URL http://www.r-project.org/

GPL : *GNU General Public License*. – URL http://www.gnu.org/copyleft/gpl.html

Gregorowicz und Kramer 2006 GREGOROWICZ, Andrew ; KRAMER, Mark A.: Mining a Large-Scale Term-Concept Network from Wikipedia / Mitre. URL http://www.mitre.org/work/tech_papers/tech_papers_06/06_1028/06_1028.pdf, 2006. – Forschungsbericht

Halpin 2006 HALPIN, H.: Identity, Reference, and Meaning on the Web. In: *Proceedings of the Workshop on Identity, Meaning and the Web (IMW06)*, URL http://www.ra.ethz.ch/CDstore/www2006/www.ibiblio.org/hhalpin/irw2006/hhalpin.pdf, 2006

Hammwöhner 2007 HAMMWÖHNER, Rainer: Semantic Wikipedia - Checking the Premises. In: *The Social Semantic Web 2007 - Proceedings of the 1st Conference on Social Semantic Web*. Bonn : Gesellschaft für Informatik, 2007 (Lecture Notes in Informatics)

Hammwöhner 2007a HAMMWÖHNER, R.: Qualitätsaspekte der Wikipedia. In: *Kommunikation@ gesellschaft* 8 (2007). – Wikis - Diskurse, Theorien und Anwendungen

Hammwöhner 2007b Hammwöhner, Rainer: Interlingual Aspects if Wikipedia's Quality. In: Robbert, Mary A. (Hrsg.) ; Markus, M. L. (Hrsg.) ; Klein, Barbara (Hrsg.): *12th International Conference on Information Quality (ICIQ-2007)*, M.I.T., 2007. – URL http://mitiq.mit.edu/iciq/PDF/INTERLINGUAL%20ASPECTS%20OF%20WIKIPEDIAS%20QUALITY.pdf. – Universität Regensburg

Hepp u. a. 2006 Hepp, M. ; Bachlechner, D. ; Siorpaes, K.: Harvesting Wiki Consensus - Using Wikipedia Entries as Ontology Elements. In: *First Workshop on Semantic Wikis* (2006)

Heyer u. a. 2006 Heyer, Gerhard ; Quasthoff, Uwe ; Wittig, Thomas: *Text Mining: Wissensrohstoff Text*. W3L, 2006. – ISBN 3-937137-30-0

Ide und Veronis 1998 Ide, Nancy ; Veronis, Jean: Word Sense Disambiguation: The State of the Art. In: *Computational Linguistics* 24 (1998), S. 1–40. – URL http://sites.univ-provence.fr/veronis/pdf/1998wsd.pdf

IEEE:SUO-WG : *Standard Upper Ontology Working Group*. – URL http://suo.ieee.org/

ISO:13250 2003 : *ISO/IEC 13250:2003 – Information technology – SGML applications – Topic maps*. 2003. – URL http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38068

ISO:2788 1986 : *ISO 2788:1986 – Documentation – Guidelines for the establishment and development of monolingual thesauri*. 1986. – URL http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=7776

ISO:639-1 2002 : *ISO 639-1:2002 – Codes for the representation of names of languages – Part 1: Alpha-2 code*. 2002. – URL http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22109

ISO:646 1991 : *ISO/IEC 646:1991 – Information technology – ISO 7-bit coded character set for information interchange*. 1991. – URL http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=4777

Jarmasz und Szpakowicz 2001 Jarmasz, M. ; Szpakowicz, S.: *Roget's Thesaurus as an Electronic Lexical Knowledge Base*, Bialystok, 2001 (NIE BEZ ZNACZENIA. Prace ofiarowane Profesorowi Zygmuntowi Saloniemu z okazji). – URL http://www.site.uottawa.ca/~mjarmasz/pubs/TR-2000-02.pdf

Jarmasz und Szpakowicz 2003 Jarmasz, Mario ; Szpakowicz, Stan: Roget's thesaurus and semantic similarity. In: *Conference on Recent Advances in Natural Language Processing*, URL http://www.site.uottawa.ca/~mjarmasz/pubs/jarmasz_roget_sim.pdf, 2003, S. 212–219

JAVA:DAO : *Core J2EE Patterns - Data Access Object*. – URL http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html

JAVA:RE : *Pattern*. – URL http://java.sun.com/javase/6/docs/api/java/util/regex/Pattern.html

JDBC : *JDK 6 Java Database Connectivity (JDBC) – related APIs & Developer Guides*. – URL http://java.sun.com/javase/6/docs/technotes/guides/jdbc/

Johnston u. a. 1998 Johnston, D. ; Nelson, S. J. ; Schulman, J. L. A. ; Savage, A. G. ; Powell, T. P.: Redefining a Thesaurus: Term-Centric No More. In: *Proceedings of the 1998 AMIA Annual Symposium*, American Medical Informatics Association, 1998. – URL http://www.amia.org/pubs/symposia/D004917.PDF

Kazama und Torisawa 2007 Kazama, Jun'ichi ; Torisawa, Kentaro: Exploiting Wikipedia as External Knowledge for Named Entity Recognition. In: *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, URL http://www.aclweb.org/anthology-new/D/D07/D07-1073.pdf, 2007, S. 698–707

Kraft und Zien 2004 Kraft, Reiner ; Zien, Jason: Mining anchor text for query refinement. In: *WWW '04: Proceedings of the 13th international conference on World Wide Web*, ACM Press, 2004, S. 666–674. – URL http://portal.acm.org/citation.cfm?id=988763. – ISBN 1-58113-844-X

Krizhanovsky 2006 Krizhanovsky, A.: *Synonym search in Wikipedia: Synarcher*. Juni 2006. – URL http://arxiv.org/abs/cs.IR/0606097

Krötzsch u. a. 2005 Krötzsch, M. ; Vrandecic, D. ; Völkel, M.: *Wikipedia and the Semantic Web - The Missing Links*. 2005. – URL http://citeseer.ist.psu.edu/krotzsch05wikipedia.html

Krötzsch u. a. 2007 Krötzsch, Markus ; Vrandecic, Denny ; Völkel, Max ; Haller, Heiko ; Studer, Rudi: Semantic Wikipedia. In: *Journal of Web Semantics* (2007), Dezember. – URL hhttp://www.aifb.uni-karlsruhe.de/Publikationen/showPublikation_english?publ_id=1551

Kumar u. a. 2006 Kumar, C. A. ; Gupta, A. ; Batool, M. ; Trehan, S.: Latent Semantic Indexing-Based Intelligent Information Retrieval System for Digital Libraries. In: *Journal of Computing and Information technology* 14 (2006), Nr. 3, S. 191

LCC : *Library of Congress Classification*. – URL http://www.loc.gov/catdir/cpso/lcc.html

Lenat und Feigenbaum 1991 Lenat, D. B. ; Feigenbaum, E. A.: On the thresholds of knowledge. In: *Artificial Intelligence* 47 (1991), Nr. 1-3, S. 185–250

Lenat u. a. 1990 Lenat, D. B. ; Guha, RV ; Pittman, K. ; Pratt, D. ; Shepherd, M.: Cyc: toward programs with common sense. In: *Communications of the ACM* 33 (1990), Nr. 8, S. 30–49

Leuf und Cunningham 2001 Leuf, Bo ; Cunningham, Ward: *The Wiki Way: Collaboration and Sharing on the Internet*. Addison-Wesley Professional, April 2001. – ISBN 0-201-71499-X

Levenshtein 1966 Levenshtein, VI: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. In: *Soviet Physics Doklady* 10 (1966), S. 707

LGPL : *GNU Lesser General Public License*. – URL http://www.gnu.org/licenses/lgpl-3.0.html

Lipscomb 2000 Lipscomb, C. E.: Medical Subject Headings (MeSH). In: *Bulletin of the Medical Library Association* 88 (2000), Nr. 3, S. 265

Mahn und Biemann 2005 Mahn, Marek ; Biemann, Chris: Tuning Co-occurrences of Higher Orders for Generating Ontology Extension Candidates. In: *ICML-Workshop on Ontology Learning*, August 2005

Maicher 2007 Maicher, Lutz: *The Impact of Semantic Handshakes*, Universität Leipzig, Diplomarbeit, 2007. – 140–151 S. – URL http://dx.doi.org/10.1007/978-3-540-71945-8_13

Matthews 2003 Matthews, B. ; Miles, A. ; Wilson, M.: *Modelling Thesauri for the Semantic Web*. 2003. – URL http://www.w3c.rl.ac.uk/SWAD/papers/thesaurus/swdbpaper.html. – CCLRC, Rutherford Appleton Laboratory, Didcot, OXON OX11 0QX, UK

MeSH : *Medical Subject Headings*. – URL http://www.nlm.nih.gov/mesh/

Mihalcea u. a. 2004 Mihalcea, R. ; Chklovski, T. ; Kilgarriff, A.: The Senseval-3 English lexical sample task. In: *Proceedings of SENSEVAL-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text [CD-ROM]*, 2004, S. 25–28

Mihalcea 2007 Mihalcea, Rada: Using Wikipedia for Automatic Word Sense Disambiguation. In: *Proceedings of NAACL HLT 2007*, URL http://www.cs.unt.edu/~rada/papers/mihalcea.naacl07.pdf, 2007

Miles 2005a Miles, A.: *Working around the identity crisis*. 2005. – URL http://esw.w3.org/topic/SkosDev/IdentityCrisis

Miles 2005b Miles, Alistair: SKOS: Simple Knowledge Organisation for the Web. In: *Cataloging & Classification Quarterly* (2005), April. – URL http://www.ingentaconnect.com/content/haworth/ccq/2007/00000043/F0020003/art00005

Miller 1995 Miller, G. A.: WordNet: A Lexical Database for English. In: *Communications of the ACM* 38 (1995), Nr. 11, S. 39

Milne u. a. 2006 Milne, David ; Medelyan, Olena ; Witten, Ian H.: Mining Domain-Specific Thesauri from Wikipedia: A case study. In: *ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings)(WI'06)*, URL http://csdl.computer.org/dl/proceedings/wi/2006/2747/00/274700442.pdf, 2006, S. 442–448

Milne u. a. 2007 Milne, David ; Witten, Ian H. ; Nichols, David M.: Extracting corpus specific knowledge bases from Wikipedia / University of Waikato. URL http://researchcommons.waikato.ac.nz/cms_papers/46/, 2007 (Working Paper 03/2007). – Forschungsbericht

Milne und Nichols 2007 Milne, Ian H. Witten D. ; Nichols, David M.: A Knowledge-Based Search Engine Powered by Wikipedia. In: *CIKM '07*, URL http://www.cs.waikato.ac.nz/%7Ednk2/publications/cikm07.pdf, 2007

M:SITES : *Wikimedia wikis*. – URL http://meta.wikimedia.org/wiki/Special:SiteMatrix

Muchnik u. a. 2007 Muchnik, Lev ; Itzhack, Royi ; Solomon, Sorin ; Louzoun, Yoram: Self-emergence of knowledge trees: Extraction of the Wikipedia hierarchies. In: *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)* 76 (2007), Nr. 1. – URL http://scitation.aip.org/getabs/servlet/GetabsServlet?prog=normal\&id=PLEEE8000076000001016106000001\&idtype=cvips\&gifs=yes

van Mulligen u. a. 2006 Mulligen, E. M. van ; Möller, E. ; Roes, P. J. ; Weeber, M. ; Meijssen, G. ; Chichester, C. ; Mons, B.: An Online Ontology: WiktionaryZ. In: *Proceedings of Knowledge Representation in Medicine (KR-MED) 2006*, 2006

MVN : *Apache Maven*. – URL http://maven.apache.org/

MW:ABOUT : *Welcome to MediaWiki.org*. – URL http://www.mediawiki.org/wiki/MediaWiki

MW:DUMPER : *MWDumper*. – URL http://www.mediawiki.org/wiki/MWDumper

MW:EDIT : *Help:Editing*. – URL http://meta.wikimedia.org/wiki/Help:Editing

MW:IMAGES : *Help:Images*. – URL http://www.mediawiki.org/wiki/Help:Images

MW:MAGIC : *Help:Magic words*. – URL http://www.mediawiki.org/wiki/Help:Magic_words

MW:NS : *Manual:Namespace*. – URL http://www.mediawiki.org/wiki/Manual:Namespace

MW:SPEC : *MediaWiki markup spec project*. – URL http://www.mediawiki.org/wiki/Markup_spec

MW:XML : *Help:Export*. – URL http://meta.wikimedia.org/wiki/Help:Export

MYSQL : *MySQL*. – URL http://mysql.com/

Nakayama u. a. 2007a Nakayama, Kotaro ; Hara, Takahiro ; Nishio, Shojiro: A Thesaurus Construction Method from Large Scale Web Dictionaries. In: *aina* 00 (2007), S. 932–939

Nakayama u. a. 2007b Nakayama, Kotaro ; Hara, Takahiro ; Nishio, Shojiro: Wikipedia Mining for an Association Web Thesaurus Construction. In: Benatallah, Boualem (Hrsg.) ; Casati, Fabio (Hrsg.) ; Georgakopoulos, Dimitrios (Hrsg.) ; Bartolini, Claudio (Hrsg.) ; Sadiq, Wasim (Hrsg.) ; Godart, Claude (Hrsg.): *WISE* Bd. 4831, Springer, 2007, S. 322–334. – URL http://dblp.uni-trier.de/db/conf/wise/wise2007.html#NakayamaHN07

Nguyen u. a. 2007 Nguyen, D. P. T. ; Matsuo, Y. ; Ishizuka, M.: Exploiting Syntactic and Semantic Information for Relation Extraction from Wikipedia. In: *IJCAI Workshop on Text-Mining & Link-Analysis (TextLink 2007)* (2007). – URL http://www.miv.t.u-tokyo.ac.jp/papers/dat-IJCAI07-TextLinkWS.pdf

Nguyen und Ishizuka 2007 Nguyen, Yutaka Matsuo Dat P. T. ; Ishizuka, Mitsuru: Relation Extraction from Wikipedia Using Subtree Mining. In: *AAAI '07*, URL http://www.miv.t.u-tokyo.ac.jp/papers/dat-AAAI07.pdf, 2007

Niles und Pease 2001 Niles, I. ; Pease, A.: Towards a standard upper ontology. In: *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001* (2001), S. 2–9

Nock 2004 Nock, C.: *Data Access Patterns: Database Interactions in Object-Oriented Applications*. Addison-Wesley Professional, 2004

OKD : *Open Knowledge Definition v1.0*. – URL http://www.opendefinition.org/1.0/

OmegaWiki : *OmegaWiki*. – URL http://www.omegawiki.org/

OpenCYC : *OpenCyc*. – URL http://www.opencyc.org/

Otlet und Fontaine 1905 Otlet, P. ; Fontaine, H. L.: Universal Decimal Classification. In: *Institut International de Bibliographie, Brussels* (1905)

Pepper und Schwab 2003 Pepper, S. ; Schwab, S.: Curing the Web's Identity Crisis: Subject Indicators for RDF. In: *XML Conference* (2003), S. 2006–04. – URL http://www.ontopia.net/topicmaps/materials/identitycrisis.html

Ponzetto und Strube 2007a Ponzetto, S. P. ; Strube, M.: An API for Measuring the Relatedness of Words in Wikipedia. In: *Companion Volume to the Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics* (2007), S. 23–30

Ponzetto und Strube 2007b Ponzetto, Simone P. ; Strube, Michael: Deriving a large scale taxonomy from wikipedia. In: *Proceedings of the 22nd National Conference on Artificial Intelligence*, URL http://www.eml-research.de/english/homes/strube/papers/aaai07.pdf, 2007

Ponzetto und Strube 2007c Ponzetto, Simone P. ; Strube, Michael: Knowledge Derived from Wikipedia for Computing Semantic Relatedness. In: *Journal of Artificial Intelligence Research* 30 (2007), S. 181–212

Ponzetto 2007 Ponzetto, Simone P.: Creating a Knowledge Base from a Collaboratively Generated Encyclopedia. In: *Proceedings of the NAACL-HLT 2007 Doctoral Consortium*, URL http://www.aclweb.org/anthology-new/N/N07/N07-3003.pdf., 2007

Quasthoff 1998 Quasthoff, Uwe: Projekt Deutscher Wortschatz. In: Heyer, Gerhard (Hrsg.) ; Wolff, Christian (Hrsg.): *Linguistik und neue Medien. Proc. 10. Jahrestagung der Gesellschaft für Linguistische Datenverarbeitung.* Wiesbaden : Deutscher Universitätsverlag, 1998

Redmann und Thomas 2007 Redmann, Tobias ; Thomas, Hendrik: The Wiki Way of Knowledge Management with Topic Maps. In: *Proceedings of the International Conference on Information Society*, URL http://www.i-society.org/2007/Contributors/Proceedings.htm, 2007, S. 352–357

RFC:3986 2005 : *Uniform Resource Identifier (URI): Generic Syntax*. 2005. – URL http://www.ietf.org/rfc/rfc3986.txt

Ruiz-Casado u. a. 2005 Ruiz-Casado, Maria ; Alfonseca, Enrique ; Castells, Pablo: Automatic Assignment of Wikipedia Encyclopedic Entries to WordNet Synsets. In: *Advances in Web Intelligence*, URL http://dx.doi.org/10.1007/11495772_59, 2005, S. 380–386

Salton und McGill 1986 Salton, G. ; McGill, M. J.: *Introduction to Modern Information Retrieval*. New York, NY, USA : McGraw-Hill, Inc., 1986. – ISBN 0-07-054484-0

Salton u. a. 1975 Salton, G. ; Wong, A. ; Yang, C. S.: A vector space model for automatic indexing. In: *Commun. ACM* 18 (1975), Nr. 11, S. 613–620. – URL http://doi.acm.org/10.1145/361219.361220

de Saussure 2001 Saussure, F. de: *Grundfragen der allgemeinen Sprachwissenschaft*. Walter de Gruyter, 2001. – Fr. original published 1916

Schönhofen 2006 Schönhofen, Péter: Identifying Document Topics Using the Wikipedia Category Network. In: *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, URL http://portal.acm.org/citation.cfm?id=1249180, 2006

SMW : *Semantic MediaWiki*. – URL http://semantic-mediawiki.org/

Steyvers und Tenenbaum 2005 Steyvers, Mark ; Tenenbaum, Joshua B.: The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth. In: *Cognitive Science* 29 (2005), Nr. 1, S. 41–78. – URL http://dblp.uni-trier.de/db/journals/cogsci/cogsci29.html#SteyversT05

Stock 2007 Stock, Wolfgang G.: *Information Retrieval*. Oldenbourg, 2007. – ISBN 3-486-58172-4, 978-3-486-58172-0

Strube und Ponzetto 2006 Strube, M. ; Ponzetto, S. P.: WikiRelate! Computing Semantic Relatedness Using Wikipedia. In: *AAAI*, URL http://www.eml-research.de/english/homes/strube/papers/aaai06.pdf, 2006

Stvilia u. a. 2005 Stvilia, B. ; Twidale, M. B. ; Smith, L. C. ; Gasser, L.: Assessing information quality of a community-based encyclopedia. In: *Proceedings of the International Conference on Information Quality*, 2005, S. 442–454

Suchanek u. a. 2007 Suchanek, F. M. ; Kasneci, G. ; Weikum, G.: Yago: a core of semantic knowledge. In: *Proceedings of the 16th international conference on World Wide Web* (2007), S. 697–706

SUMO : *Suggested Upper Merged Ontology*. – URL http://www.ontologyportal.org/

Syed u. a. 2008 Syed, Z. ; Finin, T. ; Joshi, A.: Wikipedia as an Ontology for Describing Documents. In: *Proceedings of the Second International Conference on Weblogs and Social Media*, URL http://ebiquity.umbc.edu/paper/html/id/383/Wikipedia-as-an-Ontology-for-Describing-Documents, 2008

Toral und Munoz 2006 Toral, A. ; Munoz, R.: A proposal to automatically build and maintain gazetteers for Named Entity Recognition by using Wikipedia. In: *EACL 2006* (2006)

Toral und Munoz 2007 Toral, Antonio ; Munoz, Rafael: Towards a Named Entity Wordnet (NEWN). In: *Proceedings of the 6th International Conference on Recent Advances in Natural Language Processing (RANLP)*, URL http://www.dlsi.ua.es/~atoral/publications/2007_ranlp_newn_poster.pdf, 2007

UDC : *UDC Consortium*. – URL http://www.udcc.org/

UNICODE 2007 Consortium, The U. (Hrsg.): *The Unicode Standard, Version 5.0.0*. 2007. – URL http://www.unicode.org/versions/Unicode5.0.0/

VBNW 2003 VBNW (Hrsg.): *Allgemeine Systematik für öffentliche Bibliotheken (ASB), Gliederung und Alphabetisches Schlagwortregister*. Bock und Herchen, 2003. – Erarbeitet vom Ausschuss für Systematik beim Verband der Bibliotheken des Landes Nordrhein-Westfalen

Voß 2005a Voss, Jakob: *Informetrische Untersuchungen an der Online-Enzyklopädie Wikipedia*. Berlin, Humboldt University, Diplomarbeit, 2005. – URL http://jakobvoss.de/magisterarbeit/MagisterarbeitJakobVoss.pdf

Voß 2005b Voss, Jakob: Measuring Wikipedia. In: *Proceedings of the 10th International Conference of the International Society for Scientometrics and Informetrics*, Karolinska University Press, 2005. – URL http://eprints.rclis.org/archive/00003610/

Voß 2006 Voss, Jakob: *Collaborative thesaurus tagging the Wikipedia way*. April 2006. – URL http://arxiv.org/abs/cs.IR/0604036

W3C 2005 W3C ; Miles, Alistair (Hrsg.): Quick Guide to Publishing a Thesaurus on the Semantic Web / W3C. URL http://www.w3.org/TR/2005/WD-swbp-thesaurus-pubguide-20050517, Mai 2005 (TR/2005/WD-swbp-thesaurus-pubguide-20050517). – Working Draft. W3C Working Draft 17 May 2005

W3C:HTML 1999 : *HTML 4.01 Specification*. 1999. – URL http://www.w3.org/TR/html401/

W3C:N3 2006 Berners-Lee, Tim (Hrsg.): *Notation 3 – An readable language for data on the Web*. 2006. – URL http://www.w3.org/DesignIssues/Notation3.html

W3C:NTriples 2004 : *RDF Test Cases: N-Triples*. 2004. – URL http://www.w3.org/TR/rdf-testcases/#ntriples

W3C:OWL 2004 : *OWL Web Ontology Language Overview*. 2004. – URL http://www.w3.org/TR/owl-features/

W3C:RDF 2004 : *RDF Primer*. 2004. – URL http://www.w3.org/TR/rdf-primer/

W3C:RDFS 2000 : *Resource Description Framework (RDF) Schema Specification 1.0*. 2000. – URL http://www.w3.org/TR/2000/CR-rdf-schema-20000327/

W3C:RDF/XML 2004 : *RDF/XML Syntax Specification (Revised)*. 2004. – URL http://www.w3.org/TR/rdf-syntax-grammar/

W3C:SKOS 2004 : *Simple Knowledge Organization System (SKOS) - Specifications & Documentation*. 2004. – URL http://www.w3.org/2004/02/skos/specs

W3C:SPARQL 2008 : *SPARQL Query Language for RDF*. 2008. – URL http://www.w3.org/TR/rdf-sparql-query/

W3C:URI 2001 : *URIs, URLs, and URNs: Clarifications and Recommendations 1.0*. 2001. – URL http://www.w3.org/TR/uri-clarification/

W3C:XML 2006 : *Extensible Markup Language (XML) 1.0 (Fourth Edition)*. 2006. – URL http://www.w3.org/TR/xml/

W3C:XSD 2004 : *XML Schema Part 0: Primer Second Edition*. 2004. – URL http://www.w3.org/TR/xmlschema-0/

Wales 2004 Wales, J.: Wikipedia Sociographics. In: *Proceedings of 21C3, the 21st Chaos Communication Conference*, 2004

Wilks u. a. 1996 Wilks, Y. ; Slator, B. M. ; Guthrie, L. M.: *Electric Words: Dictionaries, Computers, and Meanings*. MIT Press, 1996

Witte und Gitzinger 2007 Witte, René ; Gitzinger, Thomas: Connecting wikis and natural language processing systems. In: Désilets, Alain (Hrsg.) ; Biddle, Robert (Hrsg.): *Int. Sym. Wikis*, ACM, 2007, S. 165–176. – URL http://dblp.uni-trier.de/db/conf/wikis/wikis2007.html#WitteG07. – ISBN 978-1-59593-861-9

WM:10M 2008 : *Wikipedia Hits Milestone of Ten Million Articles Across 250 Languages*. press release. März 2008. – URL http://wikimediafoundation.org/wiki/Press_releases/10M_articles

WM:2M 2007 : *Wikipedia Reaches 2 Million Articles*. press release. September 2007. – URL http://wikimediafoundation.org/wiki/Press_releases/Wikipedia_Reaches_2_Million_Articles

WM:ABOUT : *About Wikimedia*. – URL http://wikimediafoundation.org/wiki/About_Wikimedia

WM:DOWNLOAD : *Wikimedia dump service*. – URL http://download.wikipedia.org/backup-index.html

WM:PROJECTS : *Wikimedia: Our projects*. – URL http://wikimediafoundation.org/wiki/Our_projects

WM:SURVEY 2008 : *Wikimedia Foundation and UNU-MERIT announce First Survey of Wikipedians*. press release. Januar 2008. – URL http://wikimediafoundation.org/wiki/Press_releases/UNU_survey_agreement

WordNet : *WordNet: a lexical database for the English language*. – URL http://wordnet.princeton.edu/

Wortschatz : *Projekt Deutscher Wortschatz*. – URL http://wortschatz.informatik.uni-leipzig.de/

WP:100K : *Wikipedia:100,000 feature-quality articles*. – URL http://en.wikipedia.org/wiki/Wikipedia:100%2C000_feature-quality_articles

WP:ARTIKEL : *Wikipedia:Artikel*. – URL http://de.wikipedia.org/wiki/Wikipedia:Artikel

WP:BKL : *Wikipedia:Begriffsklärung*. – URL http://de.wikipedia.org/wiki/Wikipedia:Begriffskl%C3%A4rung

WP:EA : *Wikipedia:Exzellente Artikel*. – URL http://de.wikipedia.org/wiki/Wikipedia:Exzellente_Artikel

WP:GS : *Wikipedia:Gesichtete Versionen*. – URL http://de.wikipedia.org/wiki/Wikipedia:Gesichtete_Versionen

WP:IB : *Hilfe:Infoboxen*. – URL http://de.wikipedia.org/wiki/Hilfe:Infoboxen

WP:INTERLANG : *Hilfe:Internationalisierung*. – URL http://de.wikipedia.org/wiki/Hilfe:Internationalisierung

WP:INTERWIKI : *Hilfe:Interwiki-Links*. – URL http://de.wikipedia.org/wiki/Hilfe:Interwiki-Links

WP:KAT : *Wikipedia:Kategorien*. – URL http://de.wikipedia.org/wiki/Wikipedia:Kategorien

WP:LINKS : *Hilfe:Links*. – URL http://de.wikipedia.org/wiki/Hilfe:Links

WP:LR : *Wikipedia:Löschregeln*. – URL http://de.wikipedia.org/wiki/Wikipedia:L%C3%B6schregeln

WP:NAVI : *Hilfe:Navigationsleisten*. – URL http://de.wikipedia.org/wiki/Hilfe:Navigationsleisten

WP:NK : *Wikipedia:Namenskonventionen*. – URL http://de.wikipedia.org/wiki/Wikipedia:Namenskonventionen

WP:NS : *Wikipedia:Namensräume*. – URL http://de.wikipedia.org/wiki/Wikipedia:Namensr%C3%A4ume

WP:QS : *Wikipedia:Wikipedia:Qualitätssicherung*. – URL http://de.wikipedia.org/wiki/Wikipedia:Wikipedia:Qualitätssicherung

WP:RED : *Wikipedia:Redundanz*. – URL http://de.wikipedia.org/wiki/Wikipedia:Redundanz

WP:RL : *Wikipedia:Red link*. – URL http://en.wikipedia.org/wiki/Wikipedia:Red_link

WP:STATS : *Wikipedia:Statistik*. – URL http://de.wikipedia.org/wiki/Wikipedia:Statistik

WP:TB : *Wikipedia:Textbausteine*. – URL http://de.wikipedia.org/wiki/Wikipedia:Textbausteine

WP:V : *Wikipedia:Verlinken*. – URL http://de.wikipedia.org/wiki/Wikipedia:Verlinken

WP:VOR : *Hilfe:Vorlagen*. – URL http://de.wikipedia.org/wiki/Hilfe:Vorlagen

WP:WL : *Hilfe:Weiterleitung*. – URL http://de.wikipedia.org/wiki/Hilfe:Weiterleitung

WP:WSIGA : *Wikipedia:Wie schreibe ich gute Artikel*. – URL http://de.wikipedia.org/wiki/Wikipedia:Wie_schreibe_ich_gute_Artikel

Wu und Weld 2007 Wu, Fei ; Weld, Daniel S.: Autonomously semantifying wikipedia. In: *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. New York, NY, USA : ACM, 2007, S. 41–50. – URL http://portal.acm.org/citation.cfm?id=1321440.1321449. – ISBN 978-1-59593-803-9

YAGO : *Yago*. – URL http://www.mpi-inf.mpg.de/~suchanek/downloads/yago/

Zesch und Gurevych 2006 Zesch, Torsten ; Gurevych, Iryna: Automatically creating datasets for measures of semantic relatedness. In: *COLING/ACL 2006 Workshop on Linguistic Distances*. Sydney, Australia, 2006, S. 16–24. – URL http://www.ukp.tu-darmstadt.de/sites/www.ukp.tu-darmstadt.de/files/datasets.zip

Zesch und Gurevych 2007 Zesch, Torsten ; Gurevych, Iryna: Analysis of the Wikipedia Category Graph for NLP Applications. In: *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, URL http://elara.tk.informatik.tu-darmstadt.de/publications/2007/hlt-textgraphs.pdf, 2007

Zesch u. a. 2007a Zesch, Torsten ; Gurevych, Iryna ; Mühlhäuser, Max: Analyzing and Accessing Wikipedia as a Lexical Semantic Resource. In: *Biannual Conference of the Society for Computational Linguistics and Language Technology*, 2007

Zesch u. a. 2007b Zesch, Torsten ; Gurevych, Iryna ; Mühlhäuser, Max: Comparing Wikipedia and German Wordnet by Evaluating Semantic Relatedness on Multiple Datasets. In: *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, URL http://elara.tk.informatik.tu-darmstadt.de/publications/2007/hlt-short.pdf, 2007

Zipf 1935 Zipf, George K.: *The Psycho-Biology of Language: An Introduction to Dynamic Philology*. Boston : Goughton-Mifflin, 1935

ZipfR : *ZipfR*. – URL http://www.cogsci.uni-osnabrueck.de/~severt/zipfR/

Teil VI.

Rechtliches

Lizenz

Urheber der vorliegenden Diplomarbeit ist Daniel Kinzler. Sie steht, mit der Zustimmung der Universität Leipzig in der Person von *Prof. Dr. Gerhard Heyer*, unter den *GNU Free Documentation License (kurz GFDL, deutsch „GNU-Lizenz für freie Dokumentation“)* Version 1.2 [GFDL] und darf, gemäß den Bestimmungen dieser Lizenz in dieser oder einer neueren Version, frei kopiert, verbreitet und/oder verändert werden (Veränderte Versionen sind, laut der Lizenz, als solche kenntlich zu machen). Die Arbeit wurde mit Hilfe des Textsatzsystems LaTeXverfasst, der Quelltext ist unter `/WikiWord/WikiWordThesis/Diplomarbeit/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/WikiWordThesis/Diplomarbeit/`> sowie als Archiv unter `/WikiWordThesis-tex.tar.gz` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWordThesis-tex.tar.gz`> verfügbar (siehe auch ▹G.5). Es gibt keinen vorderen oder hinteren Umschlagtext im Sinne der Lizenz. Eine Kopie des Lizenztextes der GFDL ist unter dem Titel GNU Free Documentation License enthalten.

Bei Verbreitung der vorliegenden Arbeit in veränderter oder unveränderter Form sollen Titel und Urheberschaft der Originalversion wie folgt angegeben werden:

> Daniel Kinzler, *Automatischer Aufbau eines multilingualen Thesaurus durch Extraktion semantischer und lexikalischer Relationen aus der Wikipedia*, Diplomarbeit an der Abteilung für Automatische Sprachverarbeitung, Institut für Informatik, Universität Leipzig, 2008.

Urheber des im Rahmen der vorliegenden Diplomarbeit entstandenen Softwarepaketes *WikiWord* ist Daniel Kinzler. Es steht, mit der Zustimmung der Universität Leipzig in der Person von *Prof. Dr. Gerhard Heyer* unter der *GNU Public License* Version 2 [GPL] und darf, gemäß den Bestimmungen dieser Lizenz in dieser oder einer neueren Version, frei kopiert, verbreitet und/oder verändert werden. Eine Kopie des Lizenztextes der GPL wird mit dem Quellcode von WikiWord mitgeliefert. Der Quellcode von WikiWord ist unter `/WikiWord/` auf der beiliegenden CD oder online unter <`http://brightbyte.de/DA/WikiWord/`> verfügbar, siehe ▹G.3 für eine Aufstellung von Archivdateien.

Sämtliche Texte, die aus der der Wikipedia übernommen wurden, und im Rahmen der vorliegenden Diplomarbeit verfügbar gemacht werden, stehen unter der *GNU Free Documentation License (GFDL)* Version 1.2. Sie sind als solche speziell gekennzeichnet und mit einem Verweis auf die erforderlichen Daten zur Urheberschaft versehen. Die Gesamtheit der Wikipedia ist möglicherweise als *Datenbankwerk* zu betrachten. Sie ist dann ein *Gemeinschaftswerk* (*Collective Work*) aller mitwirkenden Autoren und steht ebenfalls unter der GFDL. Statistisch ermittelte Daten, die aus der Wikipedia als Gesamtwerk abgeleitet werden, genießen mangels Schöpfungshöhe keinen eigenen urheberrechtlichen Schutz, sind gegebenenfalls aber als *Bearbeitung* (*Derivative Work*) der Wikipedia als Datenbankwerk anzusehen.

GNU Free Documentation License

Version 1.2, November 2002

51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "**Invariant Sections**" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "**Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "**Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "**Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "**Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "**Entitled XYZ**" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "**Acknowledgements**", "**Dedications**", "**Endorsements**", or "**History**".) To "**Preserve the Title**" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the

Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann.

Ort und Datum

Unterschrift