UNIVERSITÄT LEIPZIG
Fakultät für Mathematik und Informatik
Institut für Informatik

Semantikextraktion aus Wikipedia

Diplomarbeit

Leipzig, Juli 2007

vorgelegt von

Jörg Schüppel
geb. am: 16.05.1981

Studiengang Informatik

Betreuender Hochschullehrer:
Prof. Dr. Ing. habil. Klaus-Peter Fähnrich
(Inst. für Informatik, Univ. Leipzig)

Betreuer:
Dr. Sören Auer (Inst. für Informatik, Univ. Leipzig)
Jens Lehmann (Inst. für Informatik, Univ. Leipzig)

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Motivation 1
1.2 Ziele 1
1.3 Überblick über die Arbeit 2

2 Grundlagen **3**
2.1 Semantic Web 3
2.1.1 RDF 5
2.1.2 OWL 7
2.1.3 SPARQL 8
2.2 Wikis: MediaWiki 9
2.3 AJAX 9

3 Semantikextraktion aus Wikipedia Formatvorlagen **11**
3.1 Wikipedia Dump 11
3.2 Wikipedia Formatvorlagen 12
3.3 Template Extraktionsskript - Algorithmusbeschreibung 14
3.4 Extraktionsergebnisse 20
3.5 Qualitätsmessung der Extraktion 22

4 DBpedia **24**
4.1 Allgemeine Beschreibung des Projektes 24
4.2 Benutzeroberflächen zur Betrachtung der Ergebnisse 26
4.3 Anfragen an DBpedia 32
4.4 3rd Party Anwendungen: WikiStory 36

5 Verbindungen zwischen Objekten in DBpedia **37**
5.1 Kurzbeschreibung 37
5.2 Cluster-Algorithmus 38
5.3 DBpedia Relationship Finder 42
5.3.1 Benutzeroberfläche 42
5.3.2 Technische Implementierung 43
5.3.3 Nutzung des DBpedia Relationship Finders als Benutzeroberfläche zur Erkundung von RDF-Wissensbasen 52

6 Verwandte Arbeiten **54**
6.1 Semantic MediaWiki 54
6.2 YAGO 55
6.3 Freebase 55
6.4 Wikipedia3 56

7 Zusammenfassung und weitere Arbeit **57**

1 Einleitung

1.1 Motivation

Gegenwärtig erfreut sich das World Wide Web einer nie dagewesenen Beliebtheit. Nie gab es soviele Webseiten und soviele Benutzer, die Informationen dort veröffentlichen oder recherchieren. Mit der wachsenden Größe des World Wide Web steigt natürlich die Schwierigkeit für die Benutzer genau die Daten zu finden, die sie auch suchen. Aus diesem Grund rückt die Idee des Semantic Web (vgl. Kapitel 2.1) immer weiter in den Vordergrund. Die Daten und Informationen im Web sollen künftig auch von Maschinen interpretiert werden können. Das Problem dabei ist, dass ein Computer oft keine strukturierten Informationen über den Inhalt eines Dokumentes besitzt und somit dem Benutzer nicht raten kann, eine bestimmte Seite für ein bestimmtes Suchkriterium aufzusuchen. Die einzige Möglichkeit sind bisher Volltextsuchmethoden, die jedoch häufig nicht die gewünschten Ergebnisse bringen, da es schwierig für Maschinen ist, die Texte zu verstehen. Bisherige Ansätze Daten strukturiert abzuspeichern gingen von der Idee aus, zuerst die Struktur der Daten, das Schema, festzulegen und dann die Daten in dieses erstellte Schema einzupassen. Dies würde jedoch bedeuten, dass alle schon vorhandenen Daten bearbeitet werden müssten, um sie auf ein Schema auszurichten.

1.2 Ziele

Eine sinnvollere, aber fehleranfälligere, Methode Daten strukturiert zu speichern ist es herauszufinden, wie Daten auf bestimmten Seiten gespeichert sind und diese anschließend zu extrahieren. Gleichzeitig müssen dazu aus den Daten selbst die nötigen Schemata zur Strukturierung hervorgehen. Diesem Ansatz folgend soll die Wikipedia Enzyklopädie[1], deren Inhalt frei im World Wide Web verfügbar ist, als Grundlage für eine solche Datenextraktion dienen. Die Wikipedia ist hierfür besonders geeignet, da bereits eine große Masse an strukturierten Daten vorhanden ist, die jedoch bisher nur ungenügend genutzt wurden. Momentan umfasst die englische Version der Wikipedia fast 1,9 Millionen Artikel. Diese Artikel können und werden von den Benutzern täglich gelesen und gewartet. Die Wikipedia ist in über 250 Sprachversionen zugänglich. Ziel dieser Diplomarbeit ist es, eine Möglichkeit zu entwickeln, die strukturierten Daten die in vielen Wikipedia Artikeln in Form einer Formatvorlage, auch als Template be-

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

zeichnet, vorhanden sind, aus den Artikelquelltexten zu extrahieren und in Resource Description Framework (RDF) Daten (vgl. Kapitel 2.1.1) umzuwandeln. Anschließend sollen Methoden entwickelt werden, mit denen man auf Basis der extrahierten Daten die Vernetzung der Wikipedia Artikel untereinander bestimmen kann. Aufbauend darauf soll dann eine Benutzeroberfläche entwickelt werden, die dazu dienen soll Verbindungen zwischen Artikeln darzustellen.

1.3 Überblick über die Arbeit

In Kapitel 2 werden grundlegende Begriffe, wie Semantic Web, und die dazugehörigen Konzepte erläutert. Auf die Vorgehensweise der Datenextraktion aus Wikipedia Artikeln wird in Kapitel 3 eingegangen, bevor in Kapitel 4 das unter anderem aus der Template-Extraktion hervorgegangene DBpedia Projekt vorgestellt wird. Aufbauend auf den Ergebnissen der Extraktion und im Hinblick auf die in Kapitel 4 vorgestellten Benutzeroberflächen werden in Kapitel 5 die Methoden erläutert, mit denen festgestellt werden soll, wie sehr die Daten der Wikipedia miteinander zusammenhängen. Außerdem wird hier eine Benutzeroberfläche vorgestellt, die es möglich macht, Verbindungen zwischen verschiedenen in der Wikipedia beschriebenen Dingen zu finden und darzustellen. In Kapitel 6 werden einige Arbeiten vorgestellt, die ebenfalls die Extraktion von Daten aus verschiedenen Internetseiten durchführen. Als Abschluss wird in Kapitel 7 eine Zusammenfassung der Arbeit gegeben und noch weitere mögliche Verbesserungen an den Extraktionsskripten vorgestellt.

2 Grundlagen

2.1 Semantic Web

Das World Wide Web (WWW) befindet sich seit seiner Einführung im Jahr 1990 ständig im Wachstum[2]. Heutzutage ist das WWW auf Millionen von Benutzern und Milliarden von Dokumenten angewachsen. Dieses Wachstum hält weiterhin an und bringt dabei einige Probleme mit sich:

- *Informationssuche:*. Es ist schon jetzt schwierig im WWW die Informationen zu finden, die man braucht. Die gesuchten Informationen gehen in der großen Menge von irrelevanten Daten meist unter. Oft liefern Suchmaschinen für einen Suchbegriff tausende von Ergebnissen, die sich der Benutzer erst anschauen muss, um die für ihn relevanten Informationen herausfiltern zu können.

- *Präsentation:*. Ein weiteres Problem ist die Wartung von Web Ressourcen. Ein Webmaster muss sich darum kümmern, dass die zur Verfügung gestellten Daten auf seiner Webseite konsistent und aktuell gehalten sind. Die steigende Belastung der Webmaster, aufgrund von immer komplexer und umfangreicher werdenden Webseiten, führt daher zu immer mehr Webseiten mit falschen oder gegensätzlichen Inhalten.[8, S. 1f]

Aus diesen Problemen heraus entstand die Idee des Semantic Web. Diese Idee wurde im Artikel von Tim Berners-Lee, James Hendler und Ora Lassila aus dem Jahr 2001 vorgestellt: „The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.“ [9] Aus diesem Zitat lassen sich die drei grundlegenden Ziele des Semantic Web ableiten:

- Das existierende Internet soll nicht abgelöst, sondern nur erweitert werden. Es bleibt dem Anwender überlassen, ob er semantischen Inhalt nutzen möchte oder nicht.

- Es werden zusätzliche semantische, strukturierte Informationen von Menschen oder Maschinen hinzugefügt.

[2] http://www.internetworldstats.com/emarketing.htm#stats [Stand: 24.07.2007]

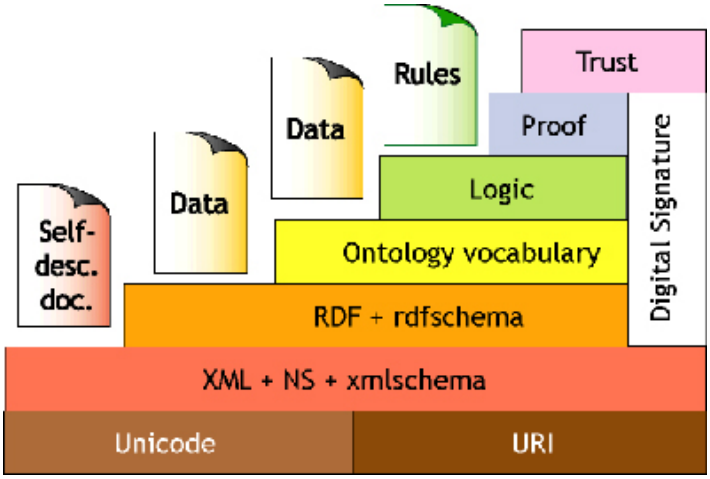


Abbildung 1: Semantic Web Stack

- Das Ziel des Semantic Web ist es, den Menschen bei der Nutzung des Webs zu unterstützen und die Zusammenarbeit zwischen Computern zu vereinfachen.[11]

Zur Erfüllung dieser Anforderungen, wurde der sogenannte „Semantic Web Stack“ oder „SW Layer Cake“ vorgestellt (vgl. Abbildung 1[3]). In der untersten Schicht stehen Unicode und URI. Unicode ist ein internationaler Zeichensatz, der unterschiedliche Zeichensätze (arabisch, kyrillisch, japanisch, lateinisch) ersetzt. Er bildet den grundlegenden Kodierungsstandard für Daten im Semantic Web. Die Uniform Resource Identifier (URI) sind ein vom World Wide Web Consortium (W3C) spezifizierter Standard zur eindeutigen Identifikation von Web-Ressourcen. Die darauffolgende Schicht beschreibt die eXtensible Markup Language[4] (XML). XML erlaubt Daten strukturiert zu speichern und auszutauschen. Die Struktur kann dabei vom Anwender selbst festgelegt werden. Über der XML-Schicht liegt die RDF[5]-Schicht. Mit RDF können Aussagen über bestimmte Ressourcen mit einer Subjekt-Prädikat-Objekt Beziehung getroffen werden. Die RDF-Syntax basiert auf XML und bietet auch einen Mechanismus um Metadaten auszudrücken (vgl. Kapitel 2.1.1). Durch RDF Schema[6] (RDFS) können Klassen (Class) und Unterklassen (subClass) festgelegt werden, denen verschiedene Ressourcen zugeordnet werden. Ebenso werden Eigenschaften (property) zu Klassen und Un-

[3]http://w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html
[4]http://www.w3.org/XML/
[5]http://www.w3.org/RDF/
[6]http://www.w3.org/TR/rdf-schema/

terklassen (subproperty) zugeordnet. Dabei können Eigenschaften einen bestimmten Anwendungsbereich (domain) bezüglich einer Klasse und einen Wertebereich (range) haben. Die darauf folgende Ontologie-Schicht bietet eine größere Ausdrucksstärke als RDFS. Die Sprache der Ontologie-Schicht ist OWL[7], welches die Empfehlung des W3C ist. In der Logic-Ebene sollen aus Informationen und aus den in der Ontologie-Ebene beschriebenen Ontologien logische Schlussfolgerungen abgeleitet werden, welche dann auf der Proof-Ebene bewiesen werden sollen. An der Spitze des Semantic Web Stacks steht die Trust-Ebene, durch die Informationen automatisch durch Computer auf ihre Vertrauenswürdigkeit überprüft werden sollen.[13, S. 13-17]

2.1.1 RDF

Mit Hilfe des Resource Description Framework (RDF) wird eine Infrastruktur definiert, die es erlaubt Metadaten zu beschreiben. Diese Metadaten sind Daten, die Daten beschreiben. Sie dienen dazu Dokumente zu charakterisieren, um sie maschinenlesbar zu machen, so dass eine Suche nach Informationen durch Computer erleichtert wird. Die Möglichkeit RDF zu notieren besteht in der Verwendung von XML, Notation 3 (N3) oder dessen Unterart N-Triples. Das RDF Datenmodell besteht aus folgenden Komponenten:

1. Ressourcen: Ein durch eine URI (Uniform Resource Identifier) eindeutig identifiziertes Objekt ist eine Ressource.

2. Blanknode (leere Knoten): Um Aussagen für komplexe Beziehungen zwischen Ressourcen zu erstellen, können Blanknodes verwendet werden. Somit können Verweise auf Ressourcen umgangen werden, die bei einer Suche in jedem Fall sowieso übersprungen werden.

3. Eigenschaften (Prädikate): Mittels eines Prädikates können Eigenschaften von Ressourcen oder Blanknotes beschrieben werden. Prädikate ermöglichen es Ressourcen mit Werten zu verknüpfen. Diese Werte sind entweder Literale oder wieder Ressourcen.

[7] http://www.w3.org/TR/owl-features/

4. Aussagen (Statements, Tripel): Es kann mit Hilfe einer Ressource, einer Eigenschaft und eines Wertes eine Aussage getroffen werden. Alle Aussagen in RDF (RDF-Statements) können durch solch eine Subjekt-Prädikat-Objekt Beziehung ausgedrückt werden. Durch die Möglichkeit alle Ressourcen eindeutig durch URIs zu identifizieren, können ganze Netze von Aussagen, sogenannte RDF-Graphen, aufgebaut werden. Außerdem können Werte selbst wieder als Ressourcen aufgefasst werden, um somit Aussagen über Aussagen zu treffen.[18, S. 304-309][5]

Ein RDF-Statement, oder Tripel, wird dann nach einem Subjekt-Prädikat-Objekt Schema aufgebaut. Dabei kann als Subjekt entweder eine Ressource oder eine Blanknode verwendet werden. Als Prädikate werden URIs verwendet. Ein Objekt enthält entweder eine Ressource, eine Blanknode oder ein Literal. Ein Literal beschreibt einen bestimmten Wert. Dabei ist es möglich einem Literal einen Datentyp zuzuweisen.

Die Syntax von RDF kann in XML notiert werden. Das soll hier nur kurz angerissen werden. Dabei ist eine RDF-Beschreibung in ein `<rdf:RDF>` Element eingeschlossen. Die Aussagen über eine Ressource können dann in `<Description>` Elementen erfolgen. Alle RDF-spezifischen Elemente müssen zum Namespace `http://www.w3.org/1999/02/22-rdf-syntax-ns#` gehören, welcher das XML Schema für RDF enthält. Das folgende Beispiel beschreibt die Universität Leipzig, wobei das Attribut „city“ wieder auf eine Ressource, nämlich Leipzig, verweist. Die Anzahl der Studenten ist hier als Literal mit dem Datentyp „Integer“ dargestellt.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dbpedia="http://dbpedia.org/property/">
  <rdf:Description
   rdf:about="http://dbpedia.org/resource/University_of_Leipzig">
    <dbpedia:city resource="http://dbpedia.org/resource/Leipzig">
    <dbpedia:students
     rdf:datatype="http://www.w3.org/2000/01/rdf-schema#Integer">
     29000
    </dbpedia:students>
  </rdf:Description>
</rdf:RDF>
```

Eine weitere Möglichkeit RDF Daten zu schreiben, besteht in den N-Triples[6], eine Unterart von N3. Diese Schreibweise bietet den Vorteil, dass sie kürzer als RDF/XML und damit leichter für einen Menschen zu überblicken ist. Es gibt zahlreiche Software-Tools um N-Triples bzw. N3 in RDF/XML umzuwandeln (zum Beispiel: Closed World Machine (CWM)[8] und Raptor[9]). Das obige Beispiel in N-Triples ausgedrückt ergibt folgendes:

```
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/city>
<http://dbpedia.org/resource/Leipzig> .

<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/students>
"29000"^^<http://www.w3.org/2000/01/rdf-schema#Integer> .
```

Hier ist die Subjekt-Prädikat-Objekt Beziehung besser zu erkennen. Es werden so „Sätze“ gebildet, die Aussagen über die Ressourcen treffen. Es können Datentypen und Sprachverknüpfungen zu den Objekten zugewiesen werden. Außerdem können Objekte Blanknodes enthalten, die auf eine verschachtelte Beschreibung des RDF-Graphen verweisen.

2.1.2 OWL

„An ontology is an explicit specification of a conceptualization.“[14]

Eine Ontologie beschreibt also eine bestimmte Domäne formal. Dies geschieht mittels verschiedener Begriffe, die zueinander in Relation stehen. Eine Ontologie ist folglich eine strukturierte Sammlung von Begriffen, welche in einer bestimmten Sprache beschrieben sein müssen. Die W3C Empfehlung für eine Ontologiesprache ist die Web Ontology Language (OWL)[15]. Durch OWL und RDFS wird die Modellierung von Wissen durch Klassen, Prädikate und Objekte mit logischen Strukturen ermöglicht. Es werden mit OWL Möglichkeiten zur Datenmodellierung geboten, die über die Grenzen von RDFS hinausgehen. Klassen und Eigenschaften können ausführlicher beschrieben werden als mit RDFS. Es können z.B. Aussagen über Disjunktheit, Äquivalenzen, Eigenschaften von Prädikaten, usw. getroffen werden.

[8] http://www.w3.org/2000/10/swap/doc/cwm.html
[9] http://librdf.org/raptor/

2.1.3 SPARQL

SPARQL[17] steht für SPARQL Protocol and RDF Query Language und hat im Juni 2007 den Status „Candidate Recommendation“ von der RDF Data Access Working Group des W3C, welche die Entwicklung und die Standardisierung von SPARQL betreibt, erhalten. Mit SPARQL ist es möglich auf RDF Daten zuzugreifen und Anfragen an verschiedenste Wissensbasen zu stellen. Dabei ist die Syntax von SPARQL der Structured Query Language (SQL) sehr ähnlich. Es werden Graphen-Muster (query patterns) gebildet, die mit den zugrunde liegenden RDF Daten abgeglichen werden, um Übereinstimmungen zu finden. Diese Graphen-Muster enthalten konkrete Ressourcen, Prädikate oder Variablen. Diese Variablen werden bei der Suche im RDF Graph „gefüllt.“ An die RDF-Daten:

```
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/country>
<http://dbpedia.org/resource/Germany> .
<http://dbpedia.org/resource/Humboldt_University_of_Berlin>
<http://dbpedia.org/property/country>
<http://dbpedia.org/resource/Germany> .
```

kann nun folgende Anfrage, „Was befindet sich in Deutschland?“, gestellt werden:

```
SELECT ?Universitaet
FROM <http://dbpedia.org>
WHERE {
  ?Universitaet
  <http://dbpedia.org/property/country>
  <http://dbpedia.org/resource/Germany> .
}
```

Der Anfrageteil in der WHERE-Klausel ist dabei genauso aufgebaut wie das N-Triples Format. Die Anfrage liefert dann als Ergebnis:

```
Universitaet
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/resource/University_of_Berlin>
```

Natürlich können mit SPARQL auch komplexere Anfragen gestellt werden. Es können Vergleiche geführt werden, Fragen gestellt werden, ob eine Variable beispielsweise eine

URI ist oder reguläre Ausdrücke angewendet werden. Außerdem können die Ergebnisse direkt in ein neues RDF-Dokument geschrieben werden.

2.2 Wikis: MediaWiki

Der Begriff „Wiki“ stammt vom hawaiianischen Wort für „schnell“. [10] Ein Wiki besteht aus miteinander verbundenen Webseiten, die von jedem Benutzer bearbeitet werden können. Es gibt in Wikis meist zwei Betrachtungsmöglichkeiten, der Betrachtungsmodus ist dafür da, sich fertige Seiten anzusehen. Dagegen nutzt man den Bearbeitungsmodus, um den Quelltext der Seite anzuzeigen und gegebenenfalls zu verändern. Hierbei gibt es einfache Möglichkeiten den Text zu formatieren. Es können Verknüpfungen zu anderen Wiki Seiten gesetzt werden. Außerdem erfolgt die Erstellung von beispielsweise Tabellen nach einem einfachen Schema.[16]

Eine verbreitete und auch von der Wikipedia verwendete Wiki-Software ist MediaWiki[11]. Die MediaWiki Software ist unter der GNU Public License (GPL) frei verfügbar. Die verschiedenen Artikeltexte werden mit PHP Hypertext Preprocessor verarbeitet und in MySQL Tabellen abgelegt. Jede Seite des MediaWikis bietet auch die Möglichkeit diese zu editieren. Dabei benötigt der Benutzer keine Kenntnisse von der HyperText Markup Language (HTML) oder anderen Programmiersprachen. Wurde eine Seite geändert so werden auch ältere Versionen dieser Seite abgespeichert, um Vandalismus entgegen wirken zu können. MediaWiki kann zudem Bild- und Multimediadateien verarbeiten und abspeichern.[12]

2.3 AJAX

Asynchronous Javascript and XML (AJAX) beschreibt kein neuartiges Konzept zur Erstellung von Internetseiten, sondern vielmehr die geschickte Nutzung von seit längerer Zeit bekannten Techniken. Der Begriff AJAX wurde erstmals von Jesse James Garrett in seinem Artikel „A New Approach to Web Applications“[13] erwähnt. Durch AJAX besteht die Möglichkeit Daten zwischen Server und Client(Browser) im Hintergrund zu übertragen, so dass der Benutzer nichts davon mitbekommt. Die aktuelle Seite wird dabei nicht neu geladen, wenn der Benutzer eine Anfrage stellt.

[10] http://de.wikipedia.org/wiki/Wiki
[11] http://www.mediawiki.org/wiki/MediaWiki
[12] http://www.mediawiki.org/wiki/MediaWiki/de
[13] http://www.adaptivepath.com/publications/essays/archives/000385.php

Es werden lediglich die für die Anfrage relevanten Teile der Seite aktualisiert. AJAX beruht dabei auf folgenden Techniken:

- Extensible HyperText Markup Language (XHTML) und Cascading Style Sheets (CSS) für standardisierte Präsentation
- Document Object Model (DOM) für den dynamischen Zugriff auf das Dokument
- XML für Datenaustausch und die eXtensible Stylesheet Language Transformation (XSLT) für Transformation
- XMLHttpRequest Objekt für asynchrone Datenübertragung zwischen Server und Client
- JavaScript um alle Techniken zu verbinden[12, S. 9f]

Populäre Beispiele für die Nutzung von AJAX Techniken sind Google Suggest[14], Google Maps[15] oder Microsofts Live Search Maps[16].

[14]http://www.google.com/webhp?complete=1&hl=en
[15]http://maps.google.com/
[16]http://maps.live.com/

3 Semantikextraktion aus Wikipedia Formatvorlagen

Im folgenden Kapitel wird die Vorgehensweise zum ersten Ziel dieser Arbeit, der Entwicklung des Extraktionsalgorithmus für Daten aus den Wikipedia Templates, vorgestellt. Hierfür soll zuerst erläutert werden, wie die Wikipedia Artikeldaten lokal installiert werden. Danach werden die Wikipedia Templates vorgestellt, um schließlich auf die Vorgehensweise zur Extraktion der Wikipedia Template Daten einzugehen. Außerdem werden die Ergebnisse der Extraktion ausgewertet. Die Daten der Ergebnisse der Extraktion stammen aus der Ausarbeitung „What have Innsbruck and Leipzig in common? Extracting Semantics from Wiki Content.“[4]

3.1 Wikipedia Dump

Zunächst müssen die Wikipedia Artikelquelltexte, welche regelmäßig aktualisiert und als Datenbankabbild (Dump) von der Wikimedia Foundation zur Verfügung gestellt werden, auf einem Datenbankserver installiert werden. Da die Quelltexte von der WikiMedia Software mittels MySQL verwaltet werden, sollen auch die Dumps in MySQL Tabellen abgelegt werden. Es gibt verschiedene Möglichkeiten diese Dumps als Grundlage für den in Kapitel 3.3 vorgestellten Extraktionsalgorithmus lokal zu installieren. Drei ausgewählte Möglichkeiten werden hier kurz vorgestellt:

- DBpedia import
- MWDumper
- xml2sql/MySQL Konsole

Die einfachste Möglichkeit die benötigten Wikipedia Tabellen zu installieren bietet das *DBpedia Importskript*[17]. Die einzige Voraussetzung, die hierfür erfüllt sein muss, ist ein PHP fähiger HTTP-Server und ein MySQL Server. Die erforderlichen Dateien werden automatisch heruntergeladen, die benötigten Tabellen erstellt und die Daten in die Tabellen eingefügt.

[17] http://dbpedia.svn.sourceforge.net/viewvc/dbpedia/importwiki/

Voraussetzung für den Import mit dem *MWDumper*[18] Tool ist ein komprimierter XML Dump, der von den Wikimedia Downloadseiten[19] bezogen werden kann. Außerdem muss die Wikimedia Tabellenstruktur bereits auf dem MySQL Server vorhanden sein. Die im Datenbank Dump enthaltenen Datensätze werden dann automatisch von MWDumper in die Wikimedia Datenbank eingefügt.

Mit dem Tool xml2sql[20] lassen sich die entpackten XML Datenbank Dumps in SQL Statements umwandeln. Die so erzeugten SQL Anweisungen zum Füllen der Wikimedia Datenbank lassen sich dann einfach und schnell über die MySQL Konsole in die Datenbank einbringen.

3.2 Wikipedia Formatvorlagen

Die Wikipedia Formatvorlagen, auch Templates genannt, bieten die Möglichkeit, Informationen in einem zuvor global definierten, einheitlichen Erscheinungsbild darzustellen. Eine Art dieser Vorlagen sind die sogenannten Infoboxen. Diese erscheinen als Tabelle auf der rechten Seite eines Wikipedia-Artikels und enthalten verschiedene Informationen über das Thema des Artikels. Der Vorteil der Infoboxen ist, dass sie für andere Artikel wiederverwendet werden können, deren Inhalt eine andere Instanz des gleichen Themas beschreibt. Ein weiterer Vorteil ist, dass bei einer Layoutänderung an der Infobox nicht alle damit verbundenen Artikel editiert werden müssen, da das Layout bei allen Artikeln aus der Infoboxvorlage übernommen wird. Der Quelltext einer jeden Infoboxvorlage beginnt mit dem, in geschweifte Klammern eingehüllten, Namen der Infobox, gefolgt von durch | abgetrennte Attribute mit ihren Attributwerten. In Abbildung 2 wird die gerenderte Infobox für die Stadt Leipzig und der zugehörige Quelltext gezeigt. Weitere Informationen sind im Wikipedia Infobox Template Artikel zu finden[21].

[18] http://www.mediawiki.org/wiki/MWDumper
[19] http://download.wikimedia.org/
[20] http://meta.wikimedia.org/wiki/Xml2sql
[21] http://en.wikipedia.org/wiki/Wikipedia:Infobox_templates

```
{{Infobox German Location
|Art                = Stadt
|Wappen             = Stadtwappen Leipzig.svg
|lat_deg            = 51 |lat_min = 20
|lon_deg            = 12 |lon_min = 23
|Lageplan           =
|Bundesland         = Sachsen
|Regierungsbezirk= Leipzig
|Kreis= Leipzig
|Höhe               =
|image_caption=St Thomas' Church in the evening.
|image_photo=Thomaskirche.jpg
|Fläche             = 297.60
|Einwohner          = 506578
|Stand              = 2006-12
|PLZ                = 04003-04357
|Vorwahl            = 0341
|Kfz                = L
|Website            = [http://www.leipzig.de/ www.leipzig.de]
|Bürgermeister      = Burkhard Jung
|Partei             = SPD
}}
```

Abbildung 2: Beispiel der Wikipedia Infobox: Leipzig - Quelltext und gerenderte Ausgabe

3.3 Template Extraktionsskript - Algorithmusbeschreibung

Für die Extraktion der semantischen Daten aus den Wikipedia Artikeln wird ein kommandozeilenbasierendes Extraktionsskript entwickelt. Durch den Aufruf auf der Kommandozeile wird eine Konfigurationsdatei mit verschiedenen Einstellungen in den Extraktionsprozess eingebunden. Der Extraktionsalgorithmus läuft in fünf Phasen ab:

- Auswahl aller Seiten, die Templates enthalten
- Auswahl und Extraktion der richtigen Templates
- Suchen der Informationen in den ausgewählten Templates und Erstellung der zugehörigen Tripel
- Nachbearbeitung der Objektwerte
- Klassenzugehörigkeiten der aktuellen Wikipedia Seite festellen

Im ersten Schritt werden durch eine SQL Anfrage alle Wikipedia Artikel ausgewählt, die Templates enthalten. Dies ist durch doppelt geschweifte Klammern („{{„) zu erkennen. Diese Anfrage wird auf die **text**-Tabelle im MediaWiki Datenbankschema ausgeführt. Das Extraktionsskript lässt sich so konfigurieren, dass an dieser Stelle nur Templates mit bestimmtem Titel ausgewählt werden um beispielsweise nur eine Extraktion über Filme zu erhalten. In diesem Schritt wird also zum Beispiel der Artikelquelltext des Artikels der Universität Leipzig ausgewählt (gekürzt):

```
{{Infobox University|
name=University of Leipzig|
native_name=Universität Leipzig|
established=[[1409]]|
faculty=14|
students=29,000|
city=[[Leipzig]]|
}}

The '''University of Leipzig''' ([[German language|German]]
''Universität Leipzig''), located in [[Leipzig]] in the
[[Free State of Saxony]] (former [[Kingdom of Saxony]]),
[[Germany]], is one of the oldest [[University|universities]]
in [[Europe]].
[...]
[[Category:Leipzig]]
```

Als nächstes werden die richtigen Templates aus dem Artikelquelltext ausgewählt. Diese Auswahl wird mittels eines rekursiven regulären Ausdrucks durchgeführt. Der reguläre Ausdruck:

```
'/\{((?>[^{}]+)|(?R))*\}/x'
```

wählt genau die Passagen im Artikelquelltext aus, die von geschweiften Klammern eingeschlossen sind. Von dem Beispielquelltext bleibt also hier nur die Infobox übrig:

```
Infobox University|
name=University of Leipzig|
native_name=Universität Leipzig|
established=[[1409]]|
faculty=14|
students=29,000|
city=[[Leipzig]]|
```

Nun wird für jedes gefundene Template Schritt 3 ausgeführt, es sei denn, das Template enthält weniger als zwei Attributwerte. Diese Einschränkung ist konfigurierbar. Ebenso können verschiedene Templates bzw. Templatemengen, durch Angabe von Wildcards,

ignoriert werden. Wurde ein Template für die Extraktion in Betracht gezogen, so werden die Tripel für dieses Template generiert. Mittels des Titels des Wikipedia Artikels wird eine URI gebildet, welche das Subjekt für die Tripel der Artikelseite bildet. In seltenen Fällen kann es vorkommen, dass dasselbe Template mehrmals auf einer Artikelseite auftaucht. In diesem Fall wird eine neue Identifikation der Artikelseite als Subjekt erstellt. Die Prädikate für die Tripel werden aus den Template- Attributen übernommen. Es gibt verschiedene Templatearten, die keine Attribute enthalten, zum Beispiel:

```
{{Elementbox_ionizationenergies2 | 499 | 1170 }}
```

In diesen Fällen wird das Prädikat als Liste der Form *property1, property2, ...* erstellt. Des Weiteren ist es möglich, dass die Wikipedia Templates verschachtelt sind. Tritt dies auf, so wird eine Blanknode als Objekt generiert und das geschachtelte Template mit dieser Blanknode als Subjekt separat extrahiert. Im obigen Beispiel würden also folgende Tripel vorbereitet:

```
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/name>
"University of Leipzig" .
[...]
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/city>
"[[Leipzig]]" .
[...]
```

Im nächsten Schritt werden die gefundenen Objektwerte nachbearbeitet. Es werden URI Verknüpfungen für Verbindungen zu anderen Wikipedia Artikeln erstellt. Außerdem werden den Literalen verschiedene Einheiten zugeordnet und diesen Literalen ein Datentyp zugewiesen. Es wird aber keine Umrechnung zwischen verschiedenen Einheiten durchgeführt. Des Weiteren werden Tripel für durch Komma getrennte Listen von Verknüpfungen auf Wikipedia Artikel generiert. Dies tritt auf, wenn mehr als zwei Verknüpfungen in einer solchen Liste auftauchen. In der Skriptkonfiguration können auch verschiedene Template-Attribute angegeben werden, für die schon im Falle einer Liste von zwei Verknüpfungen eine Linkliste generiert wird. In Tabelle 1 werden die nachbearbeiteten Objektwerte dargestellt. Nach diesem Schritt sind die Objekte in den Tripeln folgendermaßen nachbearbeitet:

Objektart	**Beispiel**	**zugeordneter Datentyp**	**Objektwert**
Integer	`7,058`	`xsd:integer`	`7058`
Dezimal	`13.3`	`xsd:decimal`	`13.3`
Bilder	`[[Image:Innsbruck.png|30px]]`	Ressource	`c:Innsbruck.png`
Links	`[[Tyrol]]`	Ressource	`w:Tyrol`
Ränge	`11<sup>th</sup>`	`u:rank`	`11`
Datum	`[[January 20]] [[2001]]`	`xsd:date`	20010120
Währung	`$30,579`	`u:Dollar`	`30579`
Große Zahlen	`1.13 [[million]]`	`xsd:Integer`	`1130000`
Große Währung	`$1.13 [[million]]`	`u:Dollar`	`1130000`
Prozent	`1.8%`	`u:Percent`	`1.8`
Einheiten	`73 g`	`u:Gramm`	`73`

Tabelle 1: Erkannte Objektwerte

```
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/name>
"University of Leipzig" .
[...]
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/faculty>
"14"^^http://www.w3.org/2001/XMLSchema#Integer .
[...]
<http://dbpedia.org/resource/University_of_Leipzig>
<http://dbpedia.org/property/city>
<http://dbpedia.org/resource/Leipzig> .
[...]
```

Im letzten Schritt werden Tripel generiert um das extrahierte Template verschiedenen Klassen zuzuordnen. Dies geschieht bei der Wikipedia mit Hilfe von Kategorien. Die dem Artikel zugeordneten Kategorien werden ausgelesen und als Tripel den extrahierten Templates zugeordnet. Die gefundenen Kategorien sind in der Wikipedia oft

wieder Oberkategorien zuzuordnen. Diese Zuordnung wird hier allerdings noch nicht mit extrahiert. Außerdem ist eine Klassifikation nach Templatename sinnvoll, welcher ebenfalls typisiert wird. Im Beispiel des Artikels zur Universität Leipzig würde also folgendes Tripel erstellt werden:

```
<http://dbpedia.org/resource/University_of_Leipzig>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://dbpedia.org/resource/Category:Leipzig> .
```

Nach der Templateextraktion werden für jeden Artikel die Überkategorien extrahiert zu denen der jeweilige Artikel gehört. Der Artikel über Leipzig gehört zum Beispiel zu den Überkategorien „Saxony“ und „Cities_in_Saxony.“ In Abbildung 1 wird der Ablauf des Extraktionsalgorithmus dargestellt. Der Quelltext des Extraktionsskriptes ist im Subversion-Verzeichnis des DBpedia Projektes[22] unter der Revision 100 verfügbar. Neben den oben genannten Einstellungsmöglichkeiten des Extraktionsskripts existieren noch folgende weitere:

- Das Ausgabeformat kann zwischen N-Triples und Character Separated Values (CSV) gewählt werden. Bei einer Ausgabe als CSV können die extrahierten Daten auch direkt in eine MySQL-Tabelle geschrieben werden.
- Die Ausgabe erfolgt entweder in einer Datei für die gesamte Extraktion oder in einer Datei pro Template Name
- Statistiken über Templates und Attribute können gesammelt werden.
- Den Prädikaten können Objekttypen zugewiesen werden. (ObjectProperty, DatatypeProperty). Außerdem können nach der ersten Definition eines Objekttyps für ein Prädikat alle weiteren Objekte an den Objekttyp angepasst werden.
- RDF-Typen und RDF-Label können in andere Dateien geschrieben werden, als die „Daten“-Statements.
- Die Liste der erkannten Einheiten und Bilder kann erweitert werden.

[22] http://dbpedia.svn.sourceforge.net/viewvc/dbpedia/extraction/

Eingabe: Datenbank mit Artikeltexten T
solange *noch nicht alle Wikipedia Artikel bearbeitet* **tue**
  Wähle die nächsten 10000 Artikeltexte T aus der Datenbank;
  für jedes T **tue**
    finde Templates X in T mittels des rekursiven regulären Ausdrucks:
    $/\backslash\{((?>[\hat{}\{\}]+)|(?R))*\backslash\}/$;
    für jedes X **tue**
      `extrahiereTemplate(`X*, Titel von* T`)`;
    Ende
    schreibe den Titel (rdfs:label) von T in die Ausgabedatei;
    schreibe Kategoriendaten von T in die Ausgabedatei;
  Ende
Ende
extrahiere Überkategorien der Artikelseiten mit Hilfe einer SQL Anfrage;

Funktion: `extrahiereTemplate(`*Templatequelltext, Subjekt*`)`;
Eingabe: Quelltext eines Templates T, Subjekt
Ergebnis: extrahierte Tripel aus dem Templatequelltext
suche Untertemplates U zum aktuellen Template T;
für jedes U **tue**
  ersetze U in T durch eine Blanknode;
  `extrahiereTemplate(`U*, Blanknode*`)`
Ende
teile T nach | suchend auf $\rightarrow$ Attribute A und Attributwerte A_W;
für jedes A **tue**
  erstelle Wert für Prädikate;
Ende
für jedes A_W **tue**
  suche nach Datentypen für A_W (vgl. Tabelle 1) ;
  erstelle Wert für Objekt;
Ende
schreibe Daten für Subjekte, Prädikate und Objekte in die Ausgabedatei;

Abbildung 3: Template Extraktionsalgorithmus

3.4 Extraktionsergebnisse

Da die englische Version der Wikipedia mit ∼1,5 Millionen Artikeln in 10 GigaByte Daten weitaus umfangreicher ist, als beispielsweise die deutsche Version mit ∼500.000 Artikeln, wird das Extraktionsskript auf die englische Wikipedia angewandt. Wie die Datenbankabbilder der Wikipedia installiert werden können wurde in Kapitel 3.1 näher beleuchtet. Die Template Extraktion nimmt, je nach verwendeter Hardware, etwa zwei bis fünf Stunden in Anspruch. Dabei werden bei der Umwandlung der Infoboxen ca. 9,2 Millionen RDF-Tripel extrahiert. In Tabelle 2 sind verschiedene Statistiken, sowie die häufigsten Templatearten und Attribute zusammengetragen.[4]

Statistiken:	
Template Arten	5,499
Template Verwendungen	754,358
Templates/Typ	137.18
Attribute/Template	8.84
Kategorien	106,049
Klassen	111,548
Instanzen von Klassen	647,348
Prädikate	8,091
Tripel	8,415,531

Templates:	
succession_box	72262
election_box	48206
infobox_album	35190
taxobox	29116
fs_player	25535
nat_fs_player	15312
imdb_title	15042
infobox_film	12733
imdb_name	12449
fs_squad2_player	10078
infobox_cvg	7930
infobox_single	7039
runway	6653

Attribute:	
name	301020
title	143887
image	110939
years	89387
before	79960
after	78806
genre	77987
type	74670
released	74465
votes	59659
reviews	58891
starring	57112
producer	53370

Tabelle 2: Extraktionsergebnisse: Statistik, am häufigsten auftretende Templates und Attribute.

Insgesamt wurden mehr als 8 Millionen Tripel aus fast 5500 verschiedenen Templatearten extrahiert. Es wurden aus mehr als 750000 Templates Daten extrahiert, was bedeutet, dass ein Template im Durchschnitt ∼137 mal verwendet wird. Aus fast 650000 verschiedenen Wikipedia Artikeln wurden Templates mit durchschnittlich 8 Attributen extrahiert. Die Klassenhierarchie, bestehend aus Kategoriedaten und Datenzuordnungen zu Templatearten, umfasst ∼111000 Klassen. In der zweiten Spalte sind häufig verwendete Templates mit der Anzahl ihres Auftetens zu sehen. Die dritte Spalte zeigt oft

verwendete Attribute und ihre Häufigkeit. In Tabelle 3 werden oft verwendete Templates, die Anzahl ihres Auftretens mit den verwendeten Attributen zusammengefasst.[4]

Template	**Anzahl**	**Benutzte Attribute**
Music album	35190	name, artist, cover, released, recorded, genre, length, label, producer, reviews, last album, next album
Species	29116	binomial, genus, genus_authority, classis, phylum, subfamilia, regnum, species, subdivision
Film	12733	starring, producer, writer, director, music, language, budget, released, distributor, image, runtime
Cities	4872	population_total, population_as_of, subdivision_type, area_total, timezone, utc_offset, population_density, leader_name, leader_title
Book	4576	author, genre, release_date, language, publisher, country, media_type, isbn, image, pages, image_caption

Tabelle 3: Extrahierte Attribute aus ausgewählten Templates.

3.5 Qualitätsmessung der Extraktion

Die Qualität der extrahierten Daten spielt eine grundlegende Rolle. Sie sind nur dann nutzbar, wenn möglichst viele Informationen in ein sauberes Format (N-Triples) gebracht wurden. Aus diesem Grund werden aus allen RDF-Tripel zufällig 1000 ausgewählt und nach folgenden Kriterien bewertet:

- sauber extrahiert
- sauber extrahiert, aber unbrauchbare Daten, z.B.:
  `<http://dbpedia.org/resource/Theology%2Fbox-header>`
  `<http://dbpedia.org/property/border>`
  `"#908435" .`
- nicht sauber extrahiert, wegen fehlender Einstellung im Extraktionsskript, z.B.:

  `<http://dbpedia.org/resource/The_End_of_St._Petersburg>`
  `<http://dbpedia.org/property/runtime>`
  `"80 min" .`
- nicht sauber extrahiert, wegen fehlender Einheit, z.B.:
  `<http://dbpedia.org/resource/`
  `Workington_%28UK_Parliament_constituency%29>`
  `<http://dbpedia.org/property/change>`
  `"+2.5" .`
- nicht sauber extrahiert, wegen Fehler im Extraktionsskript
- nicht sauber extrahiert, wegen Templatedefinition, z.B.:
  `<http://dbpedia.org/resource/Bel_Canto_%28novel%29>`
  `<http://dbpedia.org/property/over_artist>`
  `"[[Elizabeth Ackerman]], [[Echos/Nonstock]]" .`
- nicht sauber extrahiert, wegen fehlender Features im Extraktionsalgorithmus, z.B.:

  `<http://dbpedia.org/resource/The_Sea_and_the_Bells_%28album%29>`
  `<http://dbpedia.org/property/Length>`
  `"59:15:00" .`

In den ausgewählten Tripeln für diese Messung sind auch rdf:type und rdf:label Tripel mit in der Zufallsauswahl enthalten. Das Ergebnis dieser Messung weist auf eine hohe Qualität der extrahierten Daten hin. Lediglich 12,5% der Zufallstripel werden nicht als *sauber extrahiert* bewertet. Der größte Teil der fehlerhaften Tripel ist zurückzuführen auf „menschliche“ Fehler bei der Erstellung der Infoboxen in den Wikipedia Artikeln. 63,2% der fehlerhaften Daten entstehen dadurch, dass mehrere Informationen pro Infobox-Attribut verwendet werden. Aufbauend auf dieser Qualitätsmessung könnten Vorschläge an die Wikipedia Community unterbreitet werden, wie die Templates verändert werden müssten, damit die Extraktion fehlerfreier wird:

- Keine Layoutinformationen in den Templates: Layout und Inhalt sollten immer getrennt voneinander bearbeitet werden. Die Definition des Templates enthält alle nötigen Layoutinformationen, so dass keine mehr in den Templates stehen sollten.

- Ein Template pro Thema: Es sollte nur ein Template pro Thema verwendet werden, welches alle nötigen Attribute enthält. Es gibt noch zahlreiche Artikel, die ein Template pro Attribut verwenden.

- Jedes Attribut sollte genau einen Wert, oder eine Liste von Werten, besitzen. Bei mehreren Informationen pro Attribut ist eine genaue Extraktion nicht möglich.

- Bilddateien werden momentan in der Wikipedia an zwei verschiedenen Orten abgespeichert (`http://upload.wikimedia.org/wikipedia/en/` und `http://upload.wikimedia.org/wikipedia/commons/`). Welcher Ort das ist lässt sich nicht aus den Artikelquelltexten feststellen, sondern nur aus den Templatedefinitionen. An dieser Stelle setzt sich jedoch bereits das `/commons/` Verzeichnis durch.

- Für ein Thema sollte immer dasselbe Template verwendet werden. Es gibt beispielsweise verschiedene Schreibweisen für das Template für Filmdaten (Infobox_Film, Infobox film, Infobox Film).

- die Attributbezeichnungen sollten gleich sein. Es existieren zum Beispiel mehrere Schreibweisen für den Geburtsort (birthplace, PLACE_OF_BIRTH).

- Einheiten sollten einheitlich verwendet werden, damit sie vom Extraktionsalgorithmus erkannt werden können.

4 DBpedia

In diesem Kapitel werden die verschiedenen aus der Wikipedia extrahierten Schemata erläutert. Die aus den Wikipedia Artikeln extrahierten Daten (vgl. Kapitel 3) sind ein Teil dieser Schemata. Es werden Benutzeroberflächen und Anfragemöglichkeiten an die Daten vorgestellt. Für die Beschreibung des DBpedia Projektes wurde die Ausarbeitung „DBpedia: A Nucleus for a Web of Open Data“[1] herangezogen.

4.1 Allgemeine Beschreibung des Projektes

Aufgrund des Erfolgs der Template-Extraktion schlossen sich mehrere Forschungsgruppen im Projekt DBpedia zusammen, welche ebenfalls Daten aus der Wikipedia und anderen Quellen extrahierten. All diese extrahierten Informationen bilden die DBpedia Datenbasis, die durch die in Kapitel 4.2 beschriebenen Programme angezeigt bzw. an die, mit Hilfe der in Kapitel 4.3 beschriebenen Programme, Anfragen gestellt werden können.

Die DBpedia Datenbasis umfasst momentan Informationen über 1,6 Millionen Dinge. Dort enthalten sind ca. 58000 Personen, 70000 Orte, 35000 Musikalben und 12000 Filme. Außerdem umfasst die DBpedia Datenbasis 1,3 Millionen Verknüpfungen zu für den Artikel relevanten Wikipedia externen Webseiten, 207000 Wikipedia Kategorien und 75000 YAGO (vgl. Kapitel 6.2) Kategorien. Des Weiteren wurden Kurzzusammenfassungen der Artikeltexte in verschiedenen Sprachen extrahiert. Die extrahierten Ressourcen können mit Hilfe von URIs identifiziert werden. Eine solche URI hat die Form: http://dbpedia.org/resource/*RessourcenName*. Der *RessourcenName* ist gleichzeitig der Name des Wikipedia Artikels und somit kann ein Bezug zu dem Wikipedia Artikel hergestellt werden aus dem die Ressource hervorgegangen ist. Das hat außerdem den Vorteil, dass die Regeln[23] der Wikipedia zur Erstellung von Artikelnamen berücksichtigt werden, die eine hohe Qualität mit sich bringen. In Tabelle 4 werden die verschiedenen extrahierten Datenbasen vorgestellt. Ein Ziel von DBpedia ist es, die bisher in der Wikipedia vorhandenen Daten durch andere zu erweitern, um die Wissensbasis noch größer und vollständiger zu gestalten.

[23] http://de.wikipedia.org/wiki/Wikipedia:Namenskonventionen

Datensatz	**Anzahl Tripel**	**Beschreibung des Datensatzes**
Artikeldaten	5,4 Mio.	Dies ist die Grundvoraussetzung für jede DBpedia Installation. Es sind Zusammenfassungen zu allen Artikeln der englischen Wikipedia (1,6 Mio.), sowie Links auf ihre Ursprungsseiten mit den Seitentiteln enthalten. Die Artikelzusammenfassungen enthalten maximal 500 Zeichen.
Ausführliche Zusammenfassungen	1,6 Mio	Hier sind ausführlichere Zusammenfassungen der englischen Wikipedia Artikel enthalten. Sie enthalten maximal 3000 Zeichen.
Verknüpfungen zu externen Seiten	2,8 Mio.	Verknüpfungen zu Wikipedia-Externen Internetseiten von den Wikipedia Artikeln.
Kategorienverknüpfungen	5,5 Mio.	Verknüpfungen der Artikel zu Kategorien (durch SKOS [24]).
Artikeldaten (zusätzliche Sprachen)	4 Mio.	Zusammenfassungen, Titel und Verknüpfungen zu Wikipedia Artikeln in 10 zusätzlichen Sprachen.
Ausführliche Zusammenfassungen (zusätzliche Sprachen)	1,3 Mio.	Ausführliche Zusammenfassungen von Artikeln der zusätzlichen Sprachversionen.
Infoboxextraktion	9,1 Mio.	Extrahierte Daten aus den Infoboxen (vgl. Kapitel 3) der englischen Wikipedia.
Kategorien	0,9 Mio.	Zuordnung, welche Seite eine Kategorie ist und welche nicht, sowie ihre Verbindungen.
Personendaten	437000	Personendaten zu ca. 58000 Personen (Geburtsdatum, Geburtsort). Das Friend of a Friend Projekt[25](FOAF) wird benutzt.
Verknüpfungen zu Geonames	213000	Enthält Verknüpfungen zwischen Ortsartikeln in Wikipedia und ihren Pendants in der Geonames Datenbank[26].
Geonames Typen	65000	Beinhaltet „rdf:type-Tripel“ für die Orte aus der Geonames Datenbank.
Verknüpfungen zu DBLP	200	Verknüpfungen zu Informatikern und ihren Veröffentlichungen in der DBLP Datenbank[27].
Verknüpfungen zu RDF Book Mashup	9000	Verknüpfungen von Buchtiteln in DBpedia und Daten im RDF Book Mashup[28].
Artikelverknüpfungen	60 Mio.	Dieser Datensatz enthält die Wikipedia Internen Verknüpfungen. Er ist nützlich für Untersuchungen der Struktur der Wikipedia.
YAGO Klassen	1.9 Mio.	Dieser Datensatz enthält YAGO „rdf:type-Tripel“ (vgl. Kapitel6.2) für alle Wikipedia Artikel.
Verknüpfungen zu US Census	12000	Verbindungen zwischen Ortschaften und Bundestaaten der USA. Die Daten stammen von US Census[29].
Verknüpfungen zum CIA Factbook	230	Verbindungen zwischen Ländern der DBpedia Datensätze und ihren Daten im CIA Factbook[30].
Verbindungen zum Project Gutenberg	2500	Verbindungen zwischen Schriftstellern in den DBpedia Datensätzen und ihren Daten beim Project Gutenberg[31].
Verknüpfungen mit Musicbrainz	23000	Verbindungen von Musikern, Musikalben und Liedern in den DBpedia Datensätzen und Musicbrainz[32].
Verbindungen zu Eurostat	137	Verknüpfungen von Ländern und Regionen in den DBpedia Datensätzen und ihren Daten in Eurostat[33].

Tabelle 4: Datenbasen der DBpedia Extraktion

4.2 Benutzeroberflächen zur Betrachtung der Ergebnisse

Im folgenden Unterkapitel werden verschiedene Möglichkeiten aufgezeigt, wie die durch DBpedia extrahierten Wissensbasen betrachtet werden können.

Disco

Der Disco - Hyperdata Browser[34], entwickelt von Chris Bizer und Tobias Gauß bietet die Möglichkeit Daten des Semantic Web zu betrachten. Dabei werden alle verfügbaren Daten einer Ressource in HTML gerenderter Form dargestellt (vgl. Abbildung 4).

Es werden Verknüpfungen mit anderen Ressourcen sowie Beschreibungen der Ressource gezeigt. Mit Hilfe der so dargestellten Verknüpfungen können andere Ressourcen erreicht werden und somit eine ganze Wissensbasis erkundet werden. Um den Disco - Hyperdata Browser[35] zu benutzen ist keine Installation auf einem lokalen Rechner nötig, Informationen über eine Ressource werden direkt über eine eingegebene URI ausgelesen, sofern diese RDF Daten enthält. Dabei werden für alle dargestellten Informationen die Quellen angegeben, aus denen die Informationen stammen. Diese Quellen werden am Ende der Seite zusammengefasst. Außerdem ist es möglich mit dem Disco - Hyperdata Browser alle RDF Graphen darzustellen, die in einer Sitzung empfangen wurden und somit einfach zu einem späteren Zeitpunkt wieder leicht zu einer am Anfang betrachteten Seite zurückzuspringen.

[24] http://www.w3.org/2004/02/skos/
[25] http://www.foaf-project.org/
[26] http://www.geonames.org/ontology/
[27] http://www4.wiwiss.fu-berlin.de/dblp/
[28] http://sites.wiwiss.fu-berlin.de/suhl/bizer/bookmashup/
[29] http://www.census.gov/
[30] https://www.cia.gov/library/publications/the-world-factbook/index.html
[31] http://www.gutenberg.org/
[32] http://musicbrainz.org/
[33] http://epp.eurostat.ec.europa.eu/
[34] http://www4.wiwiss.fu-berlin.de/rdf_browser/
[35] http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/disco/

Abbildung 4: Disco Hyperdata Browser Benutzeroberfläche

Tabulator

Der Tabulator[36][7] ist ein RDF Browser, der sowohl für Neunutzer des Semantic Web, als auch für Entwickler, gedacht ist. Der Nutzer soll auf einfach nachvollziehbare Weise RDF Daten erkunden können, wobei Entwickler direkt erkennen sollen, wie sich ihre Daten in das Semantic Web einbringen lassen. Der Tabulator zeichnet sich durch eine sehr intuitive Benutzeroberfläche aus, die in etwa mit der eines Terminplaners oder Addressbuches zu vergleichen ist. Man startet bei einer URI einer Wissensbasis und für diese werden dann alle verfügbaren Informationen angezeigt. Sollten unter diesen Informationen Verweise auf wieder andere RDF Knoten vorhanden sein, so kann man diese direkt als Unterpunkt der dargestellten Informationen aufklappen und so wiederum neue Informationen und Verweise erhalten. In Abbildung 5 ist das Tabulatorinterface der DBpedia Leipzig Seite mit aufgeklappter Information zu Johann Sebastian Bach zu sehen. Der Tabulator Browser wurde unter anderem von Tim Berners-Lee und Jim Hollenbach Ende 2006 entwickelt und schreitet bis heute voran. Der Tabulator ist unter der W3C software license frei verfügbar. Es werden Open Source Javascripts verwendet (AJAX). Nachteil des Tabulators ist, dass er nicht in allen gängigen Browsern funktioniert. Er kann lediglich mit Mozilla Firefox[37] oder als Opera[38] Widget betrieben werden.

OntoWiki

Durch OntoWiki[3][2] können semantische Wissensbasen bearbeitet und visuell dargestellt werden. Dabei werden auch verschiedene Sichten auf die Daten berücksichtigt. Außerdem gibt es die Möglichkeit mit OntoWiki die RDF Daten zu editieren und durch semantische Annotationen zu ergänzen. Natürlich können auch neue Wissensbasen mit Hilfe der integrierten Eingabefelder erstellt werden. Diese Einsatzmöglichkeiten kommen zum Betrachten der extrahierten Daten nicht zur Geltung, seien hier aber erwähnt.

Mit OntoWiki werden verschiedene Möglichkeiten geboten auf RDF-Daten zuzugreifen. Da eine Klassifizierung von Inhalten in der Wikipedia über verschiedene Kategorien stattfindet, ist es möglich mit OntoWiki diese Struktur zu erkunden. Das Problem dabei ist, dass die Ober- und Unterkategorien in der Wikipedia auf derselben Ebene liegen und damit eine Betrachtung ausgehend von einer Oberkategorie so nicht möglich ist.

[36] http://www.w3.org/2005/ajar/tab/
[37] http://www.mozilla-europe.org/
[38] http://www.opera.com/

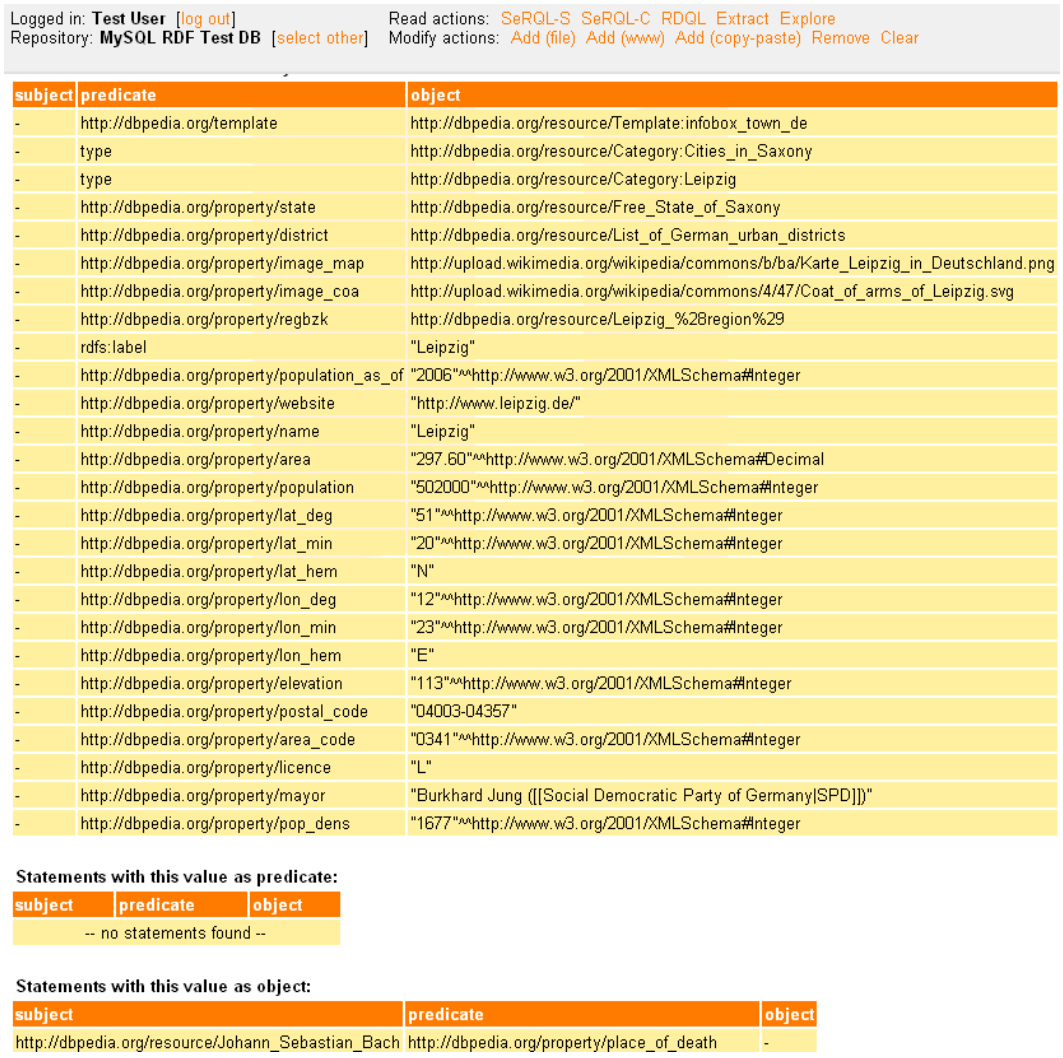
Abbildung 5: Tabulator Benutzeroberfläche

Eine solche Hierarchie muss separat extrahiert werden, um davon zu profitieren (z.B.: YAGO vgl. Kapitel 6.2). Nichtsdestotrotz kann man mit OntoWiki die verschiedenen extrahierten Instanzen von Wikipedia Templates betrachten. Außerdem ist es möglich verschiedene Suchmechanismen zu benutzen um die Ergebnislisten einzuschränken bzw. zu erweitern.

Sesame

Mit Sesame[39][10] ist es möglich RDF-Datenbasen, sogenannte Repositories, zu speichern, zu betrachten und Anfragen an sie zu stellen. Dabei können die Repositories entweder in den Hauptspeicher geladen, in Dateien abgelegt oder in einer Datenbank abgespeichert werden. In Abbildung 6 ist das Interface zum Erkunden der Daten dargestellt.

[39] http://www.openrdf.org/



Logged in: **Test User** [log out] Read actions: SeRQL-S SeRQL-C RDQL Extract Explore
Repository: **MySQL RDF Test DB** [select other] Modify actions: Add (file) Add (www) Add (copy-paste) Remove Clear

subject	predicate	object
-	http://dbpedia.org/template	http://dbpedia.org/resource/Template:infobox_town_de
-	type	http://dbpedia.org/resource/Category:Cities_in_Saxony
-	type	http://dbpedia.org/resource/Category:Leipzig
-	http://dbpedia.org/property/state	http://dbpedia.org/resource/Free_State_of_Saxony
-	http://dbpedia.org/property/district	http://dbpedia.org/resource/List_of_German_urban_districts
-	http://dbpedia.org/property/image_map	http://upload.wikimedia.org/wikipedia/commons/b/ba/Karte_Leipzig_in_Deutschland.png
-	http://dbpedia.org/property/image_coa	http://upload.wikimedia.org/wikipedia/commons/4/47/Coat_of_arms_of_Leipzig.svg
-	http://dbpedia.org/property/regbzk	http://dbpedia.org/resource/Leipzig_%28region%29
-	rdfs:label	"Leipzig"
-	http://dbpedia.org/property/population_as_of	"2006"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/website	"http://www.leipzig.de/"
-	http://dbpedia.org/property/name	"Leipzig"
-	http://dbpedia.org/property/area	"297.60"^^http://www.w3.org/2001/XMLSchema#Decimal
-	http://dbpedia.org/property/population	"502000"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/lat_deg	"51"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/lat_min	"20"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/lat_hem	"N"
-	http://dbpedia.org/property/lon_deg	"12"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/lon_min	"23"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/lon_hem	"E"
-	http://dbpedia.org/property/elevation	"113"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/postal_code	"04003-04357"
-	http://dbpedia.org/property/area_code	"0341"^^http://www.w3.org/2001/XMLSchema#Integer
-	http://dbpedia.org/property/licence	"L"
-	http://dbpedia.org/property/mayor	"Burkhard Jung ([[Social Democratic Party of Germany|SPD]])"
-	http://dbpedia.org/property/pop_dens	"1677"^^http://www.w3.org/2001/XMLSchema#Integer

Statements with this value as predicate:

subject	predicate	object
-- no statements found --		

Statements with this value as object:

subject	predicate	object
http://dbpedia.org/resource/Johann_Sebastian_Bach	http://dbpedia.org/property/place_of_death	-

Abbildung 6: Sesame Screenshot Leipzig

Es wird eine Ressource eingegeben und daraufhin werden alle RDF Statements mit dieser Ressource als Subjekt, Prädikat und Objekt angezeigt. Hierbei ist es möglich eine andere Ressource auszuwählen, um die Eigenschaften dieser Ressource zu betrachten.

Sesame ist ein von Aduna[40] entwickeltes Open Source Java Framework, man kann also auf die von Sesame gespeicherten RDF Repositories mit externen Applikationen zugreifen und Informationen verarbeiten.

[40] http://www.aduna-software.com/

4.3 Anfragen an DBpedia

SPARQL Endpoint

Mit dem OpenLink Virtuoso SPARQL protocol endpoint können SPARQL Anfragen an die DBpedia Daten gestellt werden. Eine solche Anfrage ist auf dem bereitgestellten Virtuoso Server sehr schnell. Die Ergebnisse der SPARQL Anfragen können entweder als HTML, XML, JavaScript Object Notation (JSON) oder JavaScript ausgegeben werden. Der von OpenLink Software entwickelte Virtuoso Universal Server ist ein plattformunabhängiger Server, der auf verschiedene Datenbanktreiber zugreifen kann. Er ist sozusagen ein virtueller Datenbanktreiber, der die verschiedenen Datenbanksysteme vereinen kann. Außerdem ist in Virtuoso ein HyperText Transfer Protocol-Server (HTTP-Server) enthalten und enthält noch weitere Funktionen, die aber hier nicht weiter erläutert werden. Der SPARQL Endpoint ist also dafür gedacht, verschiedene Anfragen an ihn zu senden und somit eine Ergebnisseite aufzubauen. Auf http://dbpedia.org/page/Leipzig wird zum Beispiel genau das bereitgestellt. Mit Hilfe des SPARQL Endpoints werden die Daten für Leipzig ausgegeben. Alle DBpedia Daten können hier betrachtet werden. Außerdem enthält jede Seite mehrere Verknüpfungen zu anderen Browsern. Der SPARQL Endpoint wird ebenfalls von der in Kapitel 4.3 beschriebenen Benutzeroberfläche genutzt.

DBpedia Graph Pattern Builder

Nachdem die ersten Daten durch die Infobox Extraktion bereitgestellt werden konnten, wurde der „DBpedia Graph Pattern Builder“[41] entwickelt. Mit diesem Programm wird es Benutzern ermöglicht, Anfragen an die extrahierten Daten auf einfache Weise abzusenden. Die Benutzer erstellen Graphen-Muster, ähnlich wie bei SPARQL Anfragen. Es müssen Eingabefelder für Subjekt, Prädikat und Objekt mit Variablennamen, Ressourcen, Prädikaten oder Filtern gefüllt werden. Um dem Benutzer die Auswahl der vorhandenen Ressourcen bzw. Prädikate zu erleichtern wird bei einer Eingabe im Hintergrund mittels AJAX gesucht, welche Eingaben für eine Anfrage in Frage kommen. Diese Vorschläge werden dann dem Benutzer unterbreitet. Auf diese Weise wird auch sofort klar, ob eine Anfrage überhaupt ein Ergebnis liefern kann. Um die Ausgabe zu filtern stehen Vergleichsoperationen und reguläre Ausdrücke zur Verfügung. In Abbildung 7 ist die Benutzeroberfläche des DBpedia Graph Pattern Builder für eine

[41] http://wikipedia.aksw.org/

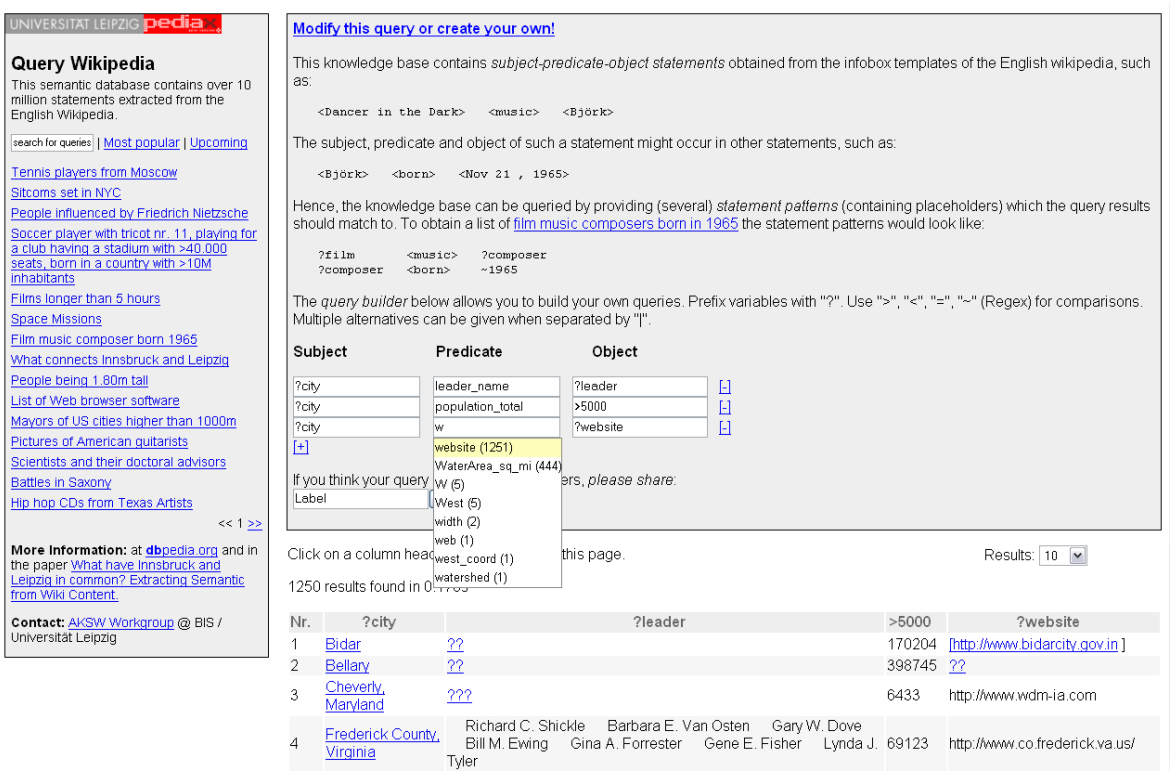


Abbildung 7: Benutzeroberfläche des DBpedia Graph Pattern Builder

Anfrage nach Städten und ihren Oberhäuptern mit mehr als 5000 Einwohnern und ihrer Internetseite dargestellt. War eine Anfrage erfolgreich, so werden zu allen gefundenen Ressourcen Verweise auf die entsprechenden Wikipedia Seiten erstellt. Außerdem werden im Ergebnis vorhandene Bilddateien direkt angezeigt. Die Anzahl der maximal angezeigten Ergebnisse kann vom Benutzer gewählt werden. Außerdem ist es möglich seine Anfrage abzuspeichern und anderen Benutzern somit zugänglich zu machen. Die Anzeige der bereits gespeicherten Anfragen kann nach der Häufigkeit ihrer Aufrufe oder nach dem Zeitpunkt des Abspeicherns geordnet werden.

Open Link Visual Query Builder

Mit dem, von OpenLink entwickelten, Visual Query Builder können SPARQL Queries mit der Hilfe einer graphischen Benutzeroberfläche erstellt und beantwortet werden. In Abbildung 8 ist die Benutzeroberfläche des Visual Query Builders zu sehen. Es lassen sich leicht Knoten und Verbindungen hinzufügen. Außerdem können an Knoten und

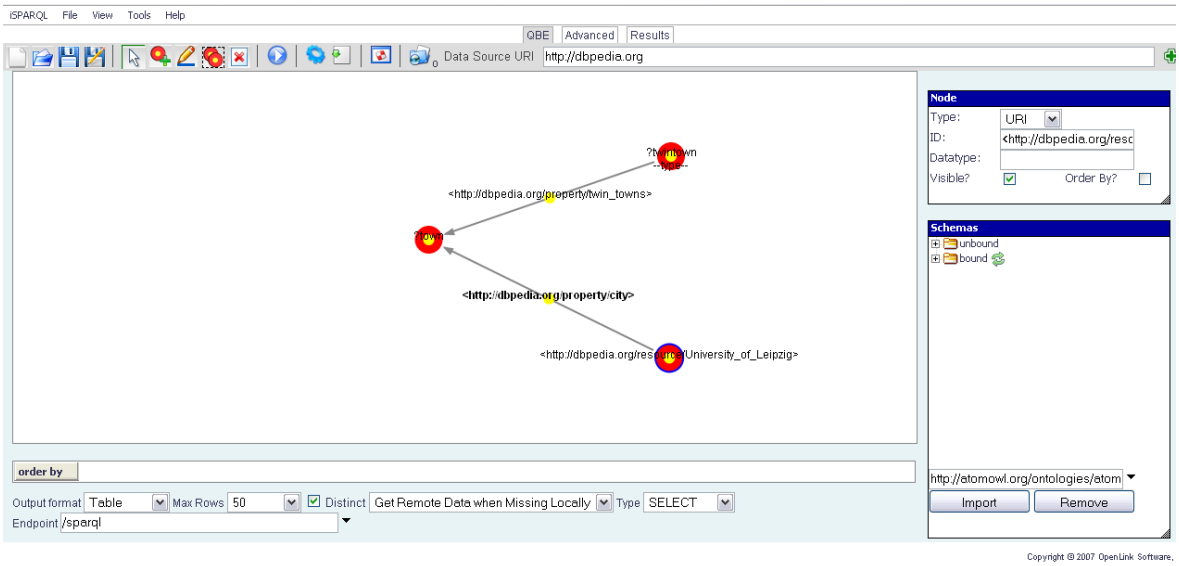


Abbildung 8: Benutzeroberfläche des OpenLink Visual Query Builder

Verbindungspfeilen Variablen eingesetzt werden, welche für die Anfrage wichtig sind. Ist der Abfragegraph fertiggestellt, so wird automatisch eine entsprechende SPARQL Abfrage generiert und ihr Ergebnis ausgegeben. Auch hier sind verschiedene Ausgabeformate möglich (HTML, XML, JSON, JavaScript). Des Weiteren können auch SPARQL Anfragen in SPARQL Syntax eingegeben werden, um dann in die graphische Oberfläche übernommen zu werden.

Search DBpedia.org

Nachdem die ersten Daten aus der Wikipedia extrahiert wurden, fehlte eine für Benutzer und Entwickler einfache Möglichkeit in den extrahierten Daten zu suchen. Mit Search DBpedia.org[42] wurde eine solche Suche, entwickelt von Georgi Kobilarov, ermöglicht. Bei Aufruf der Seite ist ein Eingabefeld zur Suche zu sehen. Gibt der Benutzer hier etwas ein, so wird zuerst eine Volltextsuche in den strukturierten DBpedia Daten gestartet. Im folgenden Schritt werden solche Seiten zur Ergebnisliste hinzugefügt, die in Verbindung mit den gefundenen Seiten stehen. Diese Verbindungssuche wird bis zu einer Entfernung von zwei Knoten von der ursprünglichen Seite durchgeführt. Des Weiteren ist die Reihenfolge mit der die Ergebnisseiten ausgegeben werden interessant.

[42] http://dbpedia.org/search/

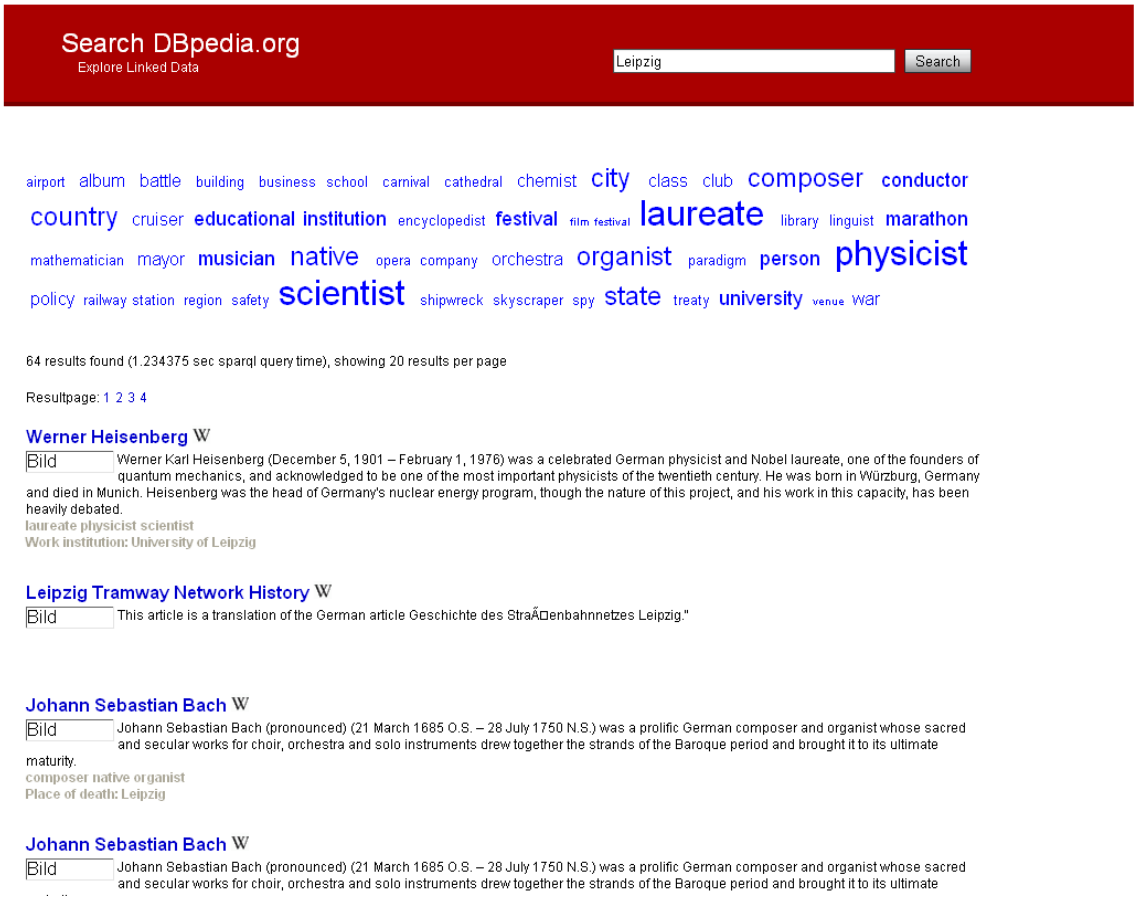


Abbildung 9: Benutzeroberfläche von search DBpedia.org mit Suchbegriff Leipzig

Es werden eingehende und ausgehende Verknüpfungen zu anderen Artikeln für diese Betrachtung herangezogen. Hierbei besitzen höher zu bewertende Seiten eine größere Zahl von eingehenden Verknüpfungen, als Seiten mit niedrigerer Bewertung. Außer den, eventuell sehr vielen, Suchergebnissen wird eine Liste mit Filtern für die Suchergebnisse angegeben. Die Filter klassifizieren die Suchergebnisse und erleichtern es somit dem Benutzer das zu finden, was gerade gesucht wird. In Abbildung 9 sind die Suchergebnisse für „Leipzig“ zu sehen. Wird ein Suchergebnis aufgerufen, so erscheint eine Darstellung der Daten, die im DBpedia Datensatz über das aufgerufene Suchergebnis enthalten sind. Außerdem werden hier Verweise auf die originalen Wikipedia Seiten aufgeführt. Mit diesen Suchmechanismen sind Sucherfolge möglich, wie man sie bisher, zum Beispiel von der Wikipedia Volltextsuche, nicht kennt. Sucht man beispielsweise nach Boris, so erscheint in der Filterliste ein „player.“ Auf diese Weise findet man

leicht die Daten zu „Boris Becker.“ Dies ist mit der Wikipedia Volltextsuche nicht zu erreichen.

4.4 3rd Party Anwendungen: WikiStory

Nach der ersten Veröffentlichung der DBpedia Extraktion wurde von Pierre Lindenbaum ein Browser namens WikiStory[43] entwickelt, der Personendaten von beispielsweise Wissenschaftlern, in einer Zeitleiste anzeigt. Für diese Anzeige wurden die extrahierten Geburts- und Todestagsdaten verarbeitet. Abbildung 10 zeigt die Benutzeroberfläche von WikiStory mit deutschen Wissenschaftlern in der Zeitleiste. Wird eine Person per Mausklick aufgerufen, so erscheinen Daten aus dem Wikipedia Artikel zu dieser Person. Außerdem werden in der Extraktion gefundene, direkte Verbindungen zwischen Personen als blaue Linie in der Zeitleiste dargestellt. Das Programm benötigt Java Webstart 1.6[44].

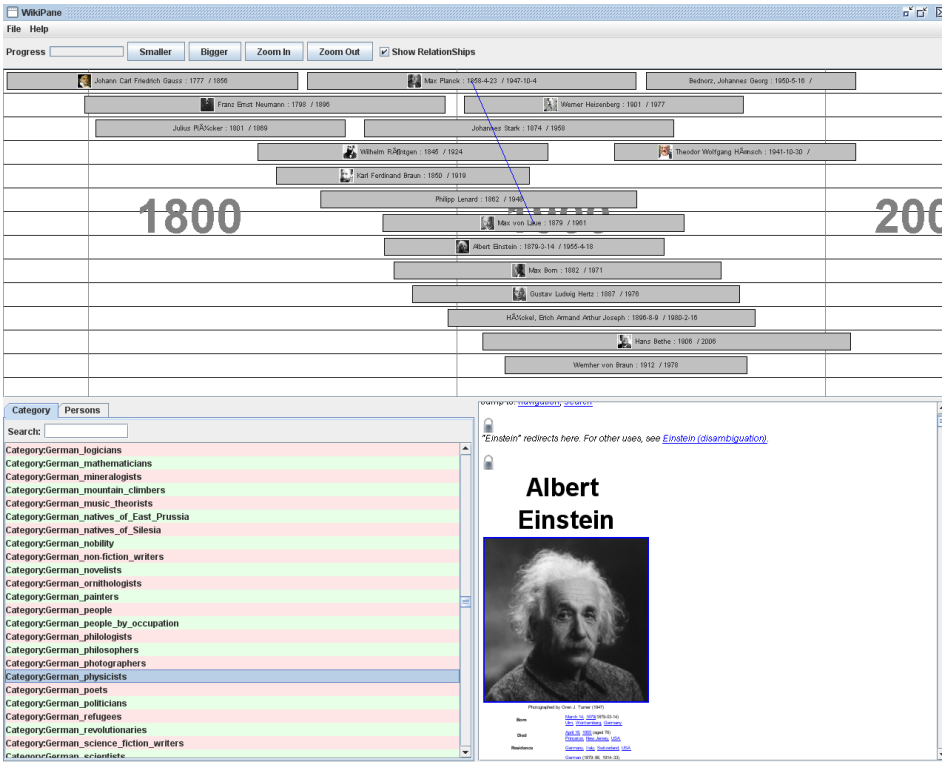


Abbildung 10: Benutzeroberfläche WikiStory

[43] http://www.urbigene.com/wikistory/

[44] http://java.sun.com/javase/6/docs/technotes/guides/javaws/index.html

5 Verbindungen zwischen Objekten in DBpedia

Nachdem in Kapitel 4 schon einige Benutzeroberflächen zum Betrachten der Daten und Anfragen an sie zu stellen vorgestellt wurden, sollen nun im nächsten Schritt der Arbeit die aus den Wikipedia Templates extrahierten Daten dazu dienen, den Verbindungsgrad zwischen den einzelnen Wikipedia Artikeln festzustellen. Auf dieser Betrachtung aufbauend soll eine Benutzeroberfläche entwickelt werden, mit der Verknüpfungen zwischen Artikeln angesehen und auch Anfragen gestellt werden können.

5.1 Kurzbeschreibung

Die Untersuchung wie stark verschiedene Ressourcen der Template Extraktion miteinander verbunden sind, bildet eine weitere Aufgabe der Arbeit. Die RDF Wissensbasis, welche durch die Template Extraktion entstanden ist, enthält eine Vielzahl von Verbindungen zwischen Objekten. In den RDF-Daten der Extraktion sind also verschiedene Ressourcen miteinander verbunden. Der RDF-Graph der Extraktion wird dann insofern in Komponenten aufgeteilt, dass in einer Komponente des RDF-Graphen die Ressourcen durch entsprechende Prädikate miteinander verbunden sind. Dabei müssen nicht alle Ressourcen miteinander verbunden sein, sondern eine Verbindung zwischen zwei Ressourcen kann auch über dazwischenliegende Ressourcen bestehen. Die so gebildeten Komponenten werden im folgenden als Cluster bezeichnet.

Das Ziel ist es, die extrahierten Ressourcen so miteinander zu verbinden, dass unterschiedliche Cluster gebildet werden können (zum Beispiel: Filme, Tiere). In Kapitel 5.2 wird die Funktionsweise des entwickelten Algorithmus, um die extrahierten Templatedaten in Cluster einzuteilen, beschrieben.

Wie stark die Vernetzung der einzelnen Ressourcen ist, zeigt schon ein erster Versuch, die Ressourcen miteinander zu verknüpfen: Hierfür wird ein Programm entwickelt, welches zuerst eine spezielle Kopie der aus der Templateextraktion gebildeten Statements-Tabelle erstellt. In dieser Statementscopy-Tabelle sind dann nur noch die Tripel vorhanden, die als Subjekt und Objekt jeweils eine DBpedia Ressource enthalten. Die Literale der Templateextraktion werden also ausgeblendet. In dieser Tabelle sind dann noch ∼1,5 Millionen verschiedene Ressourcen enthalten.

Die Untersuchung beginnt damit, dass für eine beliebige Ressource alle eingehenden und ausgehenden Verknüpfungen gezählt werden. Dies wird solange wiederholt, bis alle Ressourcen abgearbeitet sind.

Der erste Durchlauf dieses Algorithmus liefert zu den ∼1,5 Millionen verschiedenen Ressourcen bereits ∼8,5 Millionen verbundene Ressourcen. Werden die Verbindungen der gefundenen ∼8,5 Millionen Ressourcen gezählt, so werden Verknüpfungen zu ∼27 Millionen Ressourcen gefunden. Weitere Informationen über dieses Phänomen liefert der Cluster-Algorithmus (Kapitel 5.2).

5.2 Cluster-Algorithmus

Voraussetzung für den Cluster-Algorithmus ist ebenfalls die Statements-Tabelle der Templateextraktion, aus der die Statementscopy-Tabelle erstellt wird (vgl. Kapitel 5.1). Bei der Erstellung dieser Kopie ist es bereits möglich, verschiedene Ressourcen oder Prädikate zu ignorieren und nicht mit in die Statementscopy-Tabelle zu übernehmen. Des Weiteren ist es hier möglich, solche Ressourcen nicht zu kopieren, für die in der Wikipedia gleichzeitig eine Kategorienseite existiert (zum Beispiel: http://en.wikipedia.org/wiki/Leipzig und http://en.wikipedia.org/wiki/Category:Leipzig). Mit diesen Vorgehensweisen soll das Ziel erreicht werden, die Wikipedia Artikel in kleinere Cluster einzuteilen und somit Seiten aus der Verbindungssuche zu entfernen, die offensichtlich weitreichende Verbindungen besitzen.

Eine weitere Voraussetzung ist eine Cluster-Tabelle, in der später die Ergebnisse des Algorithmus abgelegt werden. Es werden sowohl die vorhandenen Ressourcen im Cluster abgespeichert, als auch die Prädikate und Ressourcen aus denen sie referenziert wurden. Des Weiteren wird eine ClusterCount-Tabelle benötigt, um statistische Informationen über die Ergebnisse der Clustersuche abzuspeichern. Hierdurch erkennt man die Anzahl der verschiedenen Cluster und die Anzahl der in ihnen vorhandenen Tripel und Ressourcen. Außerdem kann mit dem Cluster-Algorithmus die grundlegende Tabelle für den DBpedia Relationship Finder erstellt werden (vgl. Kapitel 5.3.2). Der Ablauf des eigentlichen Clusteralgorithmus erfolgt in 3 Schritten:

1. Auswahl einer Ressource R aus der kopierten Statements-Tabelle und Schreiben von R in die Cluster- Tabelle

2. Suchen der mit R verbundenen Ressourcen: Dabei werden alle Tripel gesucht, in denen R entweder als Subjekt oder als Objekt auftaucht. Die jeweils gefundenen Ressourcen werden in die Cluster- Tabelle aufgenommen und gleichzeitig das entsprechende Tripel aus der „Statementscopy- Tabelle“ gelöscht. Außerdem werden sie am Ende einer Warteliste zur weiteren Abarbeitung abgespeichert.

3. Auswahl der nächsten Ressource vom Anfang der Warteliste und Wiederholung des zweiten Schrittes solange, bis die Warteliste leer ist. Danach wird wieder bei Schritt 1 gestartet, wenn noch Tripel in der Statementscopy-Tabelle vorhanden sind.

Die Vorgehensweise des Cluster-Algorithmus, dessen Quelltext im Subversion-Verzeichnis des DBpedia Projektes[45] verfügbar ist, ist auch in Abbildung 11 als Ablaufdiagramm dargestellt.

Daten: eine Statementstabelle der DBpedia Extraktion
Ergebnis: Die Ressourcen der Statementscopytabelle eingeteilt in Cluster. Die Ergebnisse stehen in der Cluster Tabelle und in der Clustercount Tabelle.
Erstellung der nötigen Tabellen;
solange *Tripel in der Statementscopy Tabelle vorhanden* **tue**
  nimm erstes Objekt R aus der Statemenscopy Tabelle und füge es in Q ein;
  füge R in die Cluster Tabelle ein;
  solange *Warteschlange Q nicht leer* **tue**
    suche alle Subjekte/Objekte K in der Statementscopytabelle, die R als Objekt/Subjekt haben;
    wenn *K nicht in Q enthalten* **dann**
      füge K am Ende von Q ein;
      füge K in Cluster Tabelle ein;
    Ende
    lösche die Tripel aus der Statementscopy Tabelle, welche K enthalten;
    nimm Erstes Objekt aus Q;
  Ende
Ende

Abbildung 11: Cluster-Algorithmus

[45] http://dbpedia.svn.sourceforge.net/viewvc/dbpedia/relfinder/

Die Ergebnisse des Cluster-Algorithmus bestätigen die Vermutung, dass die Ressourcen der Wikipedia sehr stark miteinander verbunden sind. Der Cluster-Algorihmus liefert 13625 verschiedene Cluster, was noch nicht auf einen starken Zusammenhang hindeutet. Werden die Ergebnisse jedoch näher betrachtet, so ist schnell festzustellen, dass fast alle Ressourcen in einem Cluster zu finden sind. In Abbildung 12 ist eine Übersicht der Anzahl der Ressourcen und Tripel in den fünf größten gefundenen Clustern zu sehen.

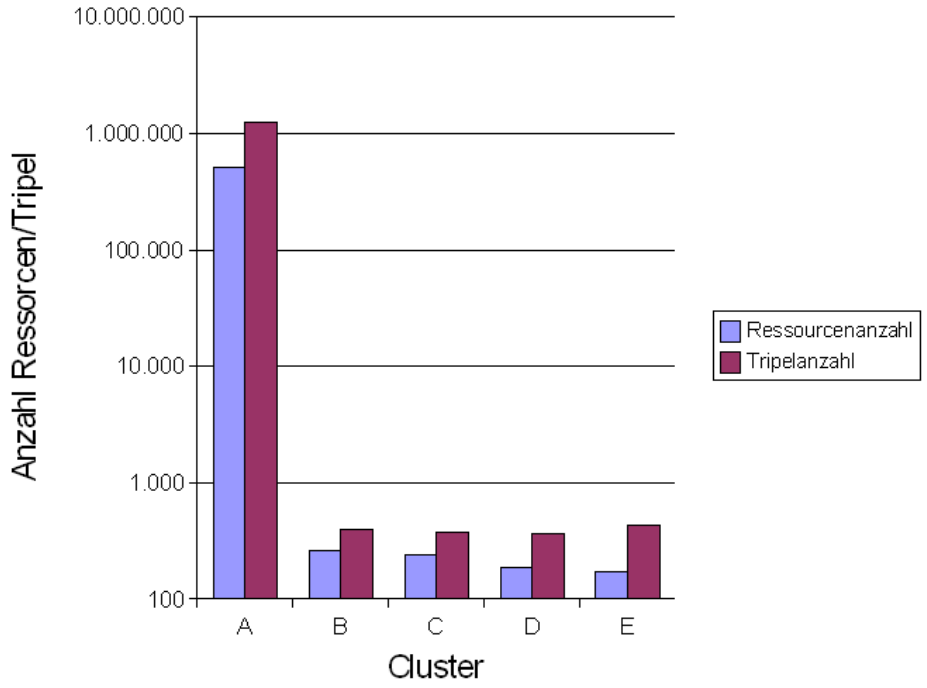


Abbildung 12: Ergebnisse des Cluster-Algorithmus - Ressourcenverteilung

Insgesamt sind im Ergebnis des Cluster-Algorithmus 567100 verschiedene Ressourcen enthalten. Es sind fast 91% dieser Ressourcen in Cluster A enthalten. Die Berechnung der Resultate dauert ca. 9 Stunden. Ein weiteres interessantes Ergebnis der Untersuchung ist, dass die aus der Template Extraktion gewonnenen Ressourcen fast alle „nah beieinander“ liegen. In Abbildung 13 wird die Anzahl der verschiedenen Ressourcen in Bezug zur Entfernung zur zufällig ausgewählten Ursprungsressource gestellt. Die meisten Ressourcen finden sich demnach bei einer Entfernung von 5 bis 9.

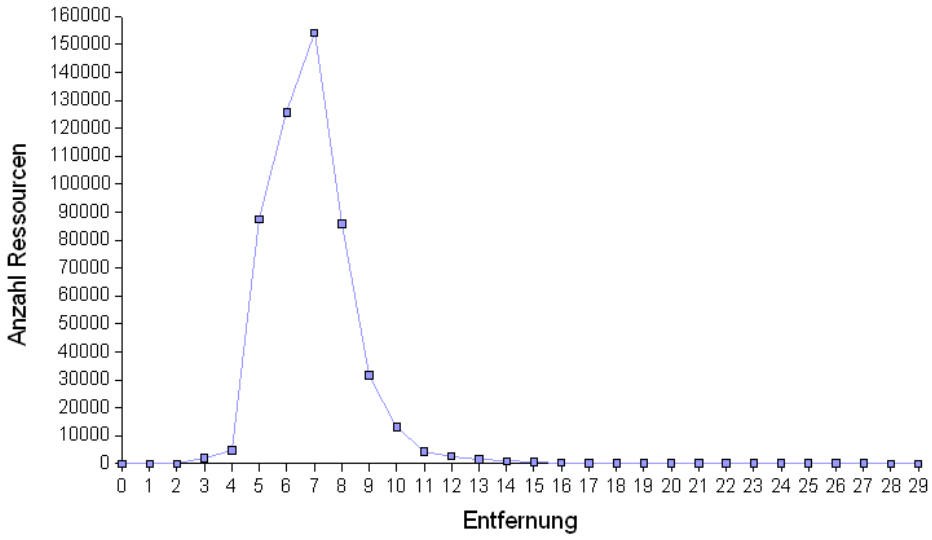


Abbildung 13: Anzahl unterschiedlicher Ressourcen bei unterschiedlicher Entfernung zur Ursprungsressource

5.3 DBpedia Relationship Finder

Da die Einteilung der Wikipedia Artikel in verschiedene, große Cluster - wie oben erläutert - nicht möglich war, soll ein Konzept entwickelt werden, die starke Verbindung zwischen den Wikipedia Artikeln für die Benutzer sichtbar zu machen. Dabei sollen die Ergebnis-Tripel des entwickelten Extraktionsalgorithmus verwendet werden, als auch die Ergebnistabellen des Cluster-Algorithmus.

5.3.1 Benutzeroberfläche

Die Benutzeroberfläche des DBpedia Relationship Finders ist intuitiv aufgebaut. Wird die Seite neu aufgerufen, so sind - wie in Abbildung 14 zu erkennen - lediglich zuvor abgespeicherte Anfragen und zwei Eingabefelder für Ressourcen zusammen mit der Suchtiefe und der maximalen Anzahl der anzuzeigenden Ergebnisse zu sehen.

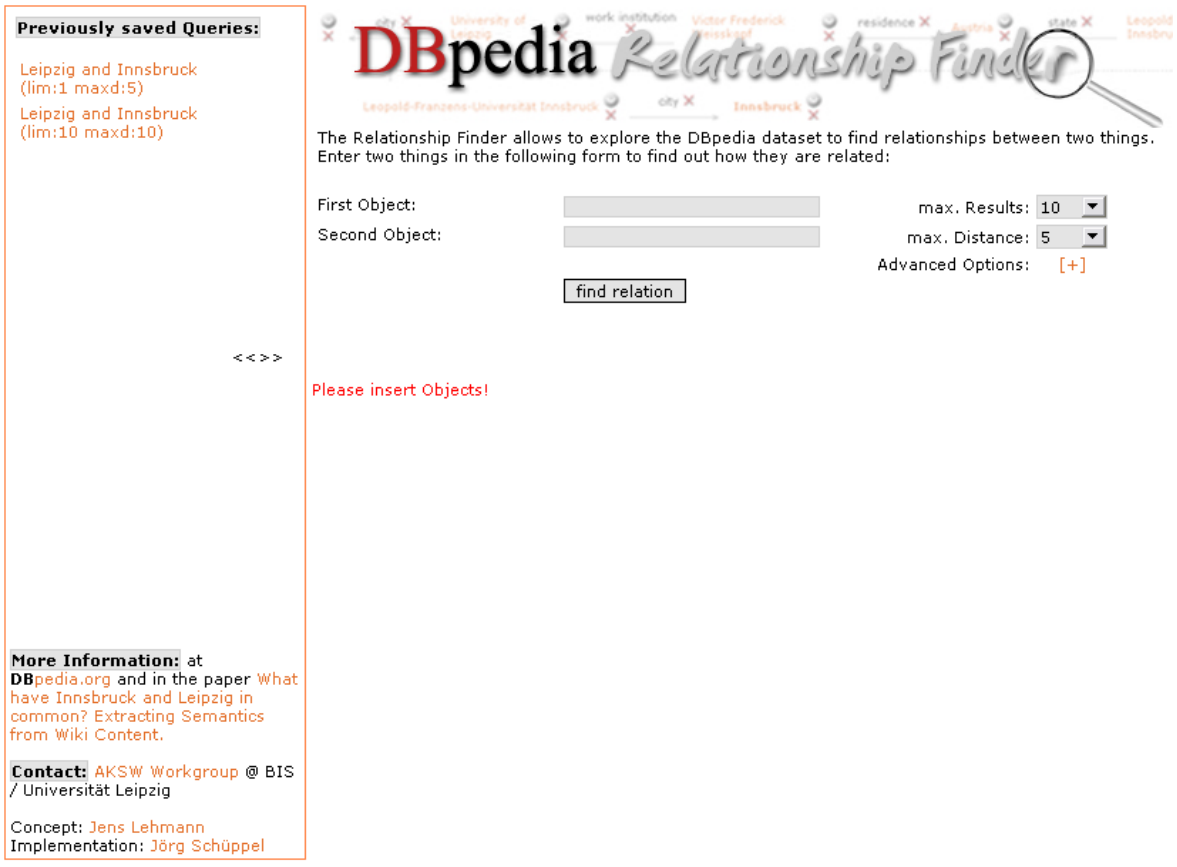


Abbildung 14: DBpedia Relationship Finder - Benutzeroberfläche bei Start

Für die Eingabe einer Ressource in die Formularfelder wird eine frei verfügbare JavaScript Bibliothek[46] verwendet, die mittels AJAX automatische Vervollständigungen für den Inhalt der Formularfelder anbietet. Nachdem der Benutzer die Anfrage mittels „find relation“ ausgeführt hat, wird die Anzeige der Ergebnisse sortiert nach Suchtiefe aufgebaut. Während der Berechnung der Ergebnisliste kann eine auf den Resultaten des Cluster-Algorithmus basierende Verbindung mittels AJAX aufgerufen werden (vgl. Abbildung 15). Die Ergebnisliste wird nur dann berechnet, wenn eine Verbindung zwischen den beiden gegebenen Ressourcen überhaupt existieren kann.

Auf der linken Seite jedes Ergebnisses steht die erste eingegebene Ressource, auf der rechten Seite die Zweite. In dieser Ergebnisansicht erhält jede angezeigte Ressource einen Button, mit dem es möglich ist mittels AJAX eine Infobox zu öffnen, in der andere extrahierte Informationen über diese Ressource angezeigt werden. Das Prädikat, welches zu einer Verknüpfung geführt hat, wird dabei rot eingefärbt. Des Weiteren wird die Verbindungsrichtung durch einen Pfeil angezeigt, auf dem das verbindende Prädikat angezeigt wird. Alle Ressourcen und Prädikate besitzen ein kleines rotes Kreuz, durch das sich das Prädikat oder die Ressource auf eine Liste setzen lässt, um im nächsten Durchlauf bei der Verbindungssuche ignoriert zu werden. Diese Liste kann auch von Hand - wieder mit Hilfe von automatischer Vervollständigung - erweitert werden. In Abbildung 16 ist die Benutzeroberfläche nach einer ausgeführten Anfrage zu sehen. Wurde eine Verbindung zwischen verschiedenen Ressourcen gefunden, so kann der Benutzer diese abspeichern und somit für andere Benutzer zugänglich machen. Das Abspeichern erfolgt ebenfalls durch AJAX-Techniken.

5.3.2 Technische Implementierung

Die Grundvoraussetzung für den DBpedia Relationship Finder ist eine aus der in Kapitel 5.2 beschriebenen kopierten Statementstabelle hervorgegangene Direct Connection-Tabelle. Hierbei werden Verbindungsrichtungen, die durch die extrahierten Tripel entstehen, außer Acht gelassen und die Ressourcen redundant gespeichert. Gibt es beispielsweise ein Tripel mit Subjekt: Universität Leipzig, Prädikat: befindet sich in, Objekt: Leipzig, so werden in der Direct Connection-Tabelle dann „Universität Leipzig, Leipzig, befindet sich in“ und „Leipzig, Universität Leipzig, befindet sich in“ abgespeichert. Das hat den Vorteil, dass eine Verbindungssuche unabhängig davon geschehen kann, welche Ressource als Erstes bzw. Zweites eingegeben wurde. Außerdem wird die

[46] http://script.aculo.us/

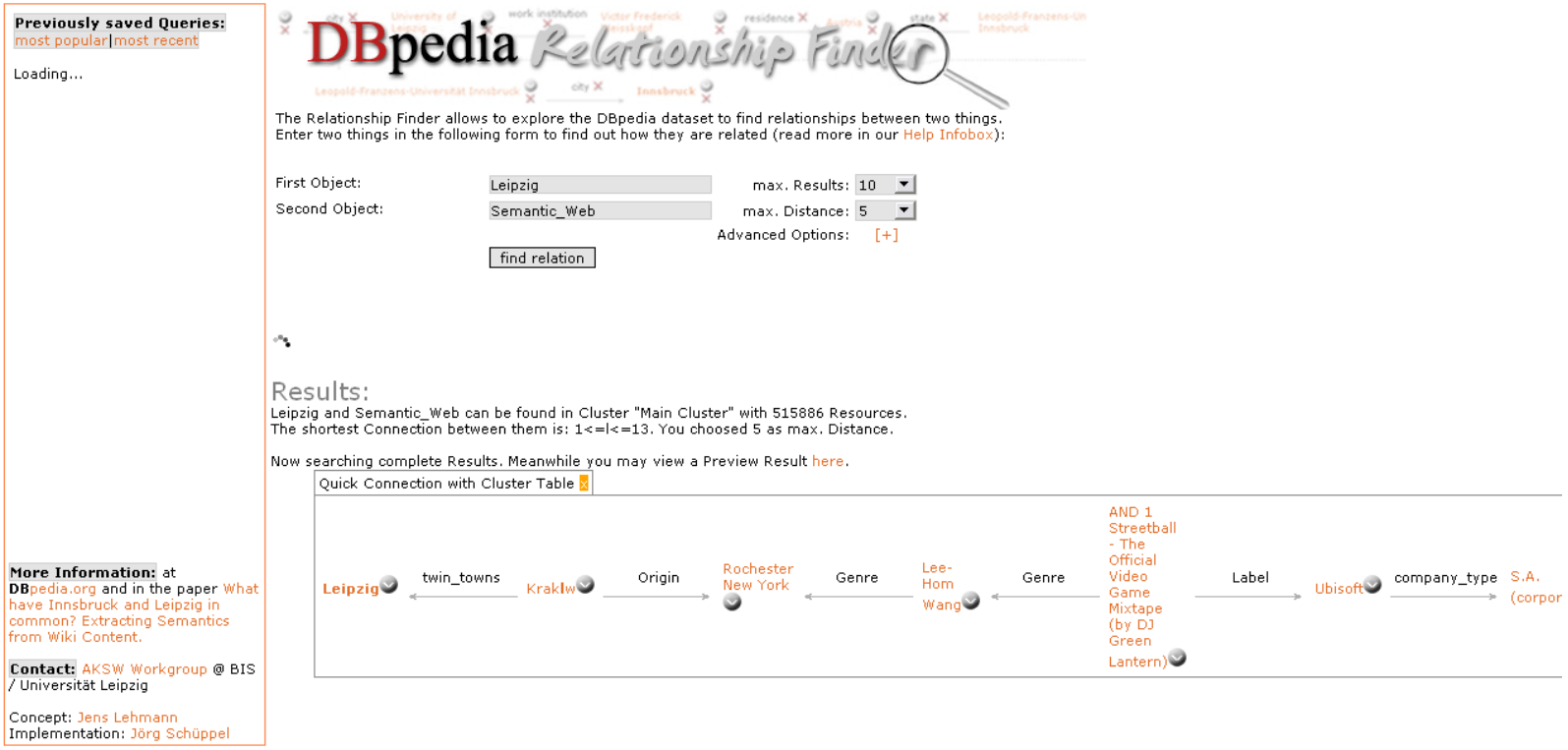


Abbildung 15: DBpedia Relationship Finder - Vorberechnung einer Verbindung mittels Cluster-Algorithmus - Resultat

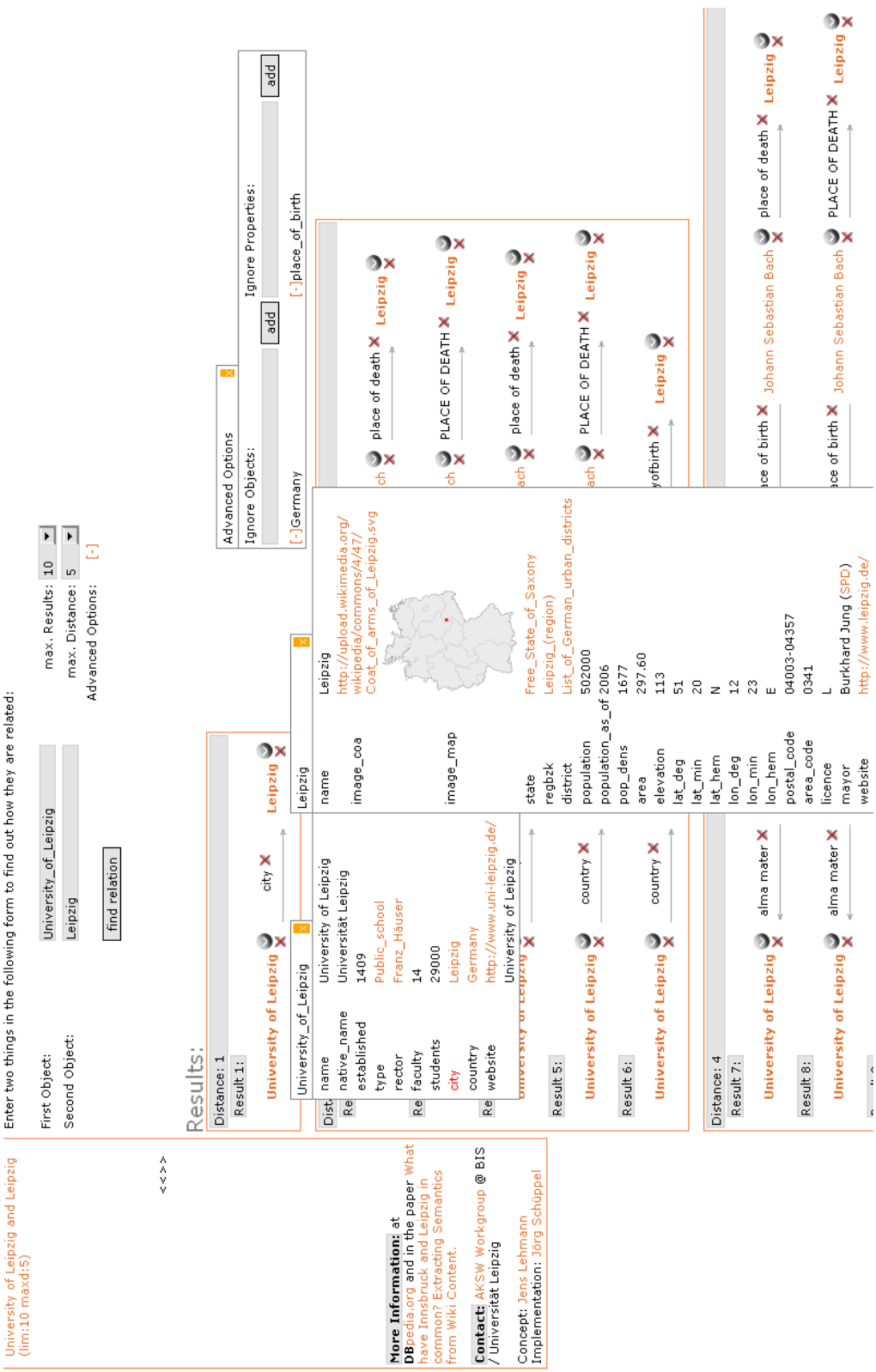


Abbildung 16: DBpedia Relationship Finder - Benutzeroberfläche Ergebnisanzeige

in Kapitel 5.2 beschriebene Kopie der Statementstabelle, sowie die Statementstabelle selbst für den DBpedia Relationship Finder benötigt. Des Weiteren besteht die Möglichkeit den DBpedia Relationship Finder so zu konfigurieren, dass auch die Ergebnistabellen des in Kapitel 5.2 beschriebenen Cluster-Algorithmus verwendet werden. Damit können Verbindungen im Hintergrund schnell vorberechnet werden, oder sofort Aussagen darüber getroffen werden, ob eine Verbindung zwischen den gegebenen Ressourcen bei gegebenen Distanzwerten überhaupt möglich ist.

Die Berechnung der Verbindung zwischen den verschiedenen Ressourcen läuft in 6 Schritten ab:

1. Eingabe der Ressourcen durch den Benutzer und Auswahl der Ergebnisanzahl sowie der maximalen Suchtiefe
2. Überprüfung, ob gestellte Anfrage bereits gespeichert wurde.
3. Testen, ob Verbindung zwischen den angefragten Ressourcen überhaupt existieren kann (nur bei Verwendung der Cluster- Algorithmus Ergebnisse)
4. Berechnung der Verbindung
5. Ausgabe der Ergebnisse
6. Abspeichern des Ergebnisses (optional)

Bei Schritt 1 werden die gewünschten Ressourcen in die Formularfelder eingegeben. Zur Vereinfachung der Eingabe von Objekten, werden mittels AJAX Vorschläge für Ressourcen gegeben. Außerdem ist es möglich, über die „Advanced Options“ sogleich verschiedene Ressourcen und Prädikate für die Verbindungsberechnung auszuschließen. Auch bei der Eingabe in diese Formularfelder werden Vorschläge für vorhandene Ressourcen/Prädikate gegeben (vgl. Abbildung 17).

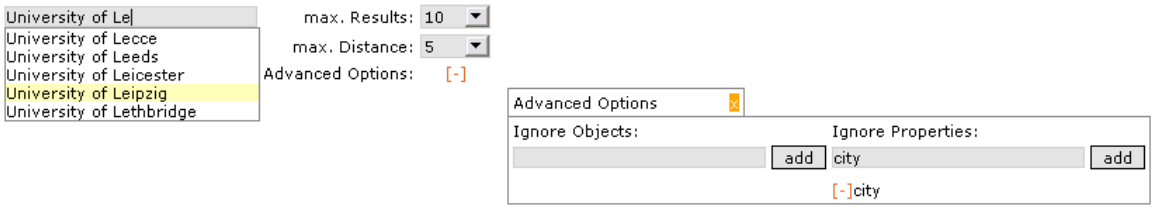


Abbildung 17: DBpedia Relationship Finder - Autovervollständigung

Im Schritt 2 wird überprüft, ob die gestellte Anfrage bereits gespeichert ist. Ist dies der Fall, so wird das gespeicherte Ergebnis direkt ausgegeben und keine Neuberechnung durchgeführt. Sollte die Anfrage noch nicht gespeichert sein, so wird mit Schritt 3 fortgefahren. Dieser dritte Schritt wird nur dann ausgeführt, wenn, laut Konfiguration, die Ergebnisse des Cluster-Algorithmus verwendet werden sollen. In diesem Fall wird mittels SQL-Abfrage überprüft, ob beide Ressourcen im selben Cluster zu finden sind. Ist dies der Fall, so wird über die Formel $|\texttt{Tiefe Ressource 1} - \texttt{Tiefe Ressource 2}| \leq \texttt{Distanz} \leq \texttt{Tiefe Ressource 1} + \texttt{Tiefe Ressource 2}$ ermittelt, ob die zur Verbindungsberechnung eingegebene Distanz ausreichend ist um die Minimalbedingung zu erfüllen.

Dazu werden aus der Cluster-Tabelle die Tiefenwerte der eingegebenen Ressourcen bezüglich des Ursprungsobjektes aus dem Cluster-Algorithmus ausgewählt. Die vom Benutzer gewünschte maximale Distanz zwischen den Ressourcen darf also nicht kleiner sein als $|\texttt{Tiefe Ressource 1} - \texttt{Tiefe Ressource 2}|$. Außerdem wird in diesem Schritt ein Link generiert, durch den der Benutzer sich während der Ausführung des 4. Schrittes eine auf den Ergebnissen des Cluster-Algorithmus basierende Verbindung anzeigen lassen kann (vgl. Abbildung 18). Für die Berechnung dieser Verbindung werden die in der Cluster-Tabelle abgespeicherten Referenzobjekte und Referenzprädikate genutzt. Für die eingegebenen Ressourcen werden damit die verbundenen Ressourcen bis hin zur Ursprungsressource des Cluster-Algorithmus gesucht. Die so entstandenen Listen von verbundenen Ressourcen werden nun miteinander verglichen und ab der Stelle ausgegeben, wo sie sich das erste Mal voneinander unterscheiden.

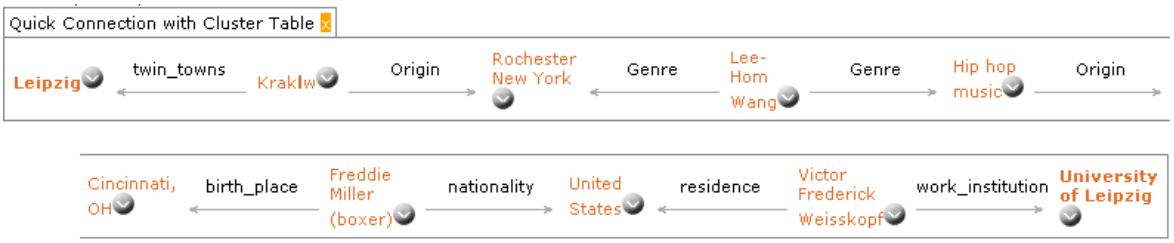


Abbildung 18: DBpedia Relationship Finder - Vorberechnung einer Verbindung

Bei Schritt 4 wird die Verbindung zwischen den beiden Ressourcen berechnet. Hierzu wird eine von der Suchtiefe abhängige SQL-Anfrage generiert, welche JOINS enthält, die die DirectConnection-Tabelle immer wieder mit sich selbst verknüpfen, bis eine Verbindung gefunden wurde oder die Suchtiefe erreicht ist. In dieser Anfrage werden

auch eventuell zu ignorierende Ressourcen bzw. Prädikate mit berücksichtigt.
Im 5. Schritt werden die Ergebnisse der Verbindungsberechnung ausgegeben. Abhängig von der gewählten Anzahl der Ergebnisse wird der 4. Schritt auch mehrfach für jeweils steigende Suchtiefen ausgeführt, wenn die maximale Anzahl der Ergebnisse noch nicht erreicht wurde. In der Ergebnisübersicht befinden sich an jeder angezeigten Ressource Buttons, welche die Möglichkeit bieten, ein Fenster mit Informationen zu der Ressource zu öffnen. Die Berechnung der Inhalte dieses Fensters erfolgt mittels AJAX. Es werden aus der Statements-Tabelle alle Tripel ausgewählt, die die gewünschte Ressource als Subjekt besitzen. Danach werden die Prädikate mit den zugehörigen Objekten ausgegeben. Hierbei werden mehrfach auftretende Prädikate nur einmal angezeigt, um die Übersichtlichkeit zu wahren. Es wird des Weiteren versucht, die bei der Infobox Extraktion extrahierten Bildverweise direkt zu öffnen und mit im Informationsfenster anzuzeigen. Wikipedia-Verweise werden automatisch in Verweise auf die richtige Pediax-Seite[47], einer Verbindung der Benutzeroberflächen von Google Maps und Wikipedia, umgewandelt. Außerdem wird das Prädikat, welches für eine Verbindung zur nächsten/letzten Ressource sorgte, rot eingefärbt (vgl. Abbildung 19).

[47] http://www.pediax.de/

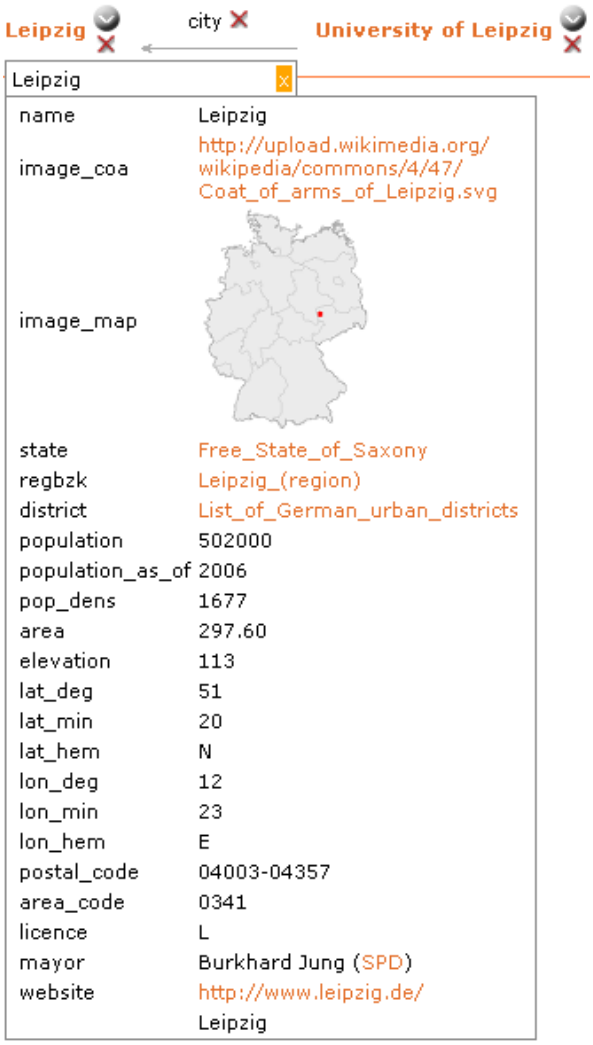


Abbildung 19: DBpedia Relationship Finder - Infoboxausgabe

Die Ergebnisübersicht enthält außerdem an allen Ressourcen und Prädikaten Verweiskreuze. Wenn diese betätigt werden, so werden die gewünschten Ressourcen/Prädikate mittels JavaScript sofort in die Liste der zu ignorierenden Ressourcen/ Prädikate aufgenommen.

Im letzten Schritt wird dem Benutzer die Möglichkeit gegeben seine Anfrage abzuspeichern und somit für andere Benutzer schneller zugänglich zu machen. Hierbei werden die eingegebenen Ressourcen, die gewählte Ergebnisanzahl, die maximale Suchtiefe, die Ergebnisse der Berechnung aus Schritt 4, die Zeit, eventuelle ignorierte Ressourcen/ Prädikate und die Anzahl der Aufrufe dieser Anfrage abgespeichert. Das Abspeichern

erfolgt ebenfalls mittels AJAX in einer separaten Datei. War das Abspeichern erfolgreich, so erscheint die Anfrage sofort auf der linken Seite angezeigten Liste mit gespeicherten Anfragen (vgl. Abbildung 20). Es ist möglich, diese Liste nach der Anzahl der Aufrufe oder nach dem Zeitpunkt des Abspeicherns zu sortieren.

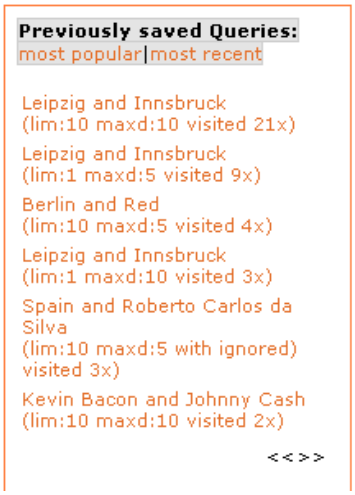


Abbildung 20: DBpedia Relationship Finder - gespeicherte Anfragen

In Abbildung 21 ist die Funktionsweise des Relationship Finders dargestellt. Der aktuelle Quelltext des DBpedia Relationship Finder kann aus dem Subversion-Verzeichnis des DBpedia Projektes[48] bezogen werden und die aktuelle Veröffentlichung unter http://wikipedia.aksw.org/relfinder/ benutzt werden.

[48] http://dbpedia.svn.sourceforge.net/viewvc/dbpedia/relfinder/

Eingabe: erste eingegebene Ressource $\rightarrow R_1$
zweite eingegebene Ressource $\rightarrow R_2$
maximale Distanz $\rightarrow D$
maximale Ergebnisanzahl $\rightarrow L$
optional: zu ignorierende Ressourcen/ Prädikate $\rightarrow IGNORE$
Darstellung der Benutzeroberfläche;
wenn *Werte für* R_1, R_2, D, L *gegeben* **dann**
  wenn *Ergebnis für* $R_1, R_2, D, L, IGNORE$ *bereits abgespeichert* **dann**
    Ausgabe des gespeicherten Ergebnisses;
  sonst
    wenn *Verbindung zwischen* R_1, R_2 *mit* D *möglich ist (Test mit Cluster-Tabelle)* **dann**
      Berechnung der minimalen Distanz D_{min} mittels Cluster-Tabelle;
      Ausgabe in welchem Cluster R_1 und R_2 liegen;
      Linkausgabe zur Verbindungsberechnung mittels Cluster-Tabelle;
      aktuelle Distanz $D_A = D_{min}$;
      solange $D_A \prec D$ *und* $L \succ 0$ **tue**
        Suche Verbindung in DirectConnection-Tabelle mit der für die aktuelle Distanz D_A gebildeten SQL Anfrage;
        $L = L-$Anzahl Suchergebnisse;
        $D_A + 1$;
        wenn *Anzahl Suchergebnisse* $\succ$ *0* **dann**
          Ausgabe der gefundenen Verbindungen;
        sonst
          wenn *maximale Distanz* D *erreicht* **dann**
            Fehlschlag für die Suche ausgeben;
          Ende
        Ende
      Ende
    sonst
      Fehlerausgabe: gewählte Distanz D zu gering für Verbindung;
      Linkausgabe zur Verbindungsberechnung mittels Clustertabelle;
    Ende
  Ende
sonst
  Weise Benutzer darauf hin Werte einzugeben;
Ende

Abbildung 21: DBpedia Relationship Finder - Funktionsweise

5.3.3 Nutzung des DBpedia Relationship Finders als Benutzeroberfläche zur Erkundung von RDF-Wissensbasen

Der Ansatz mit dem DBpedia Relationship Finder verschiedene Verbindungen zwischen den Ressourcen der Infobox Extraktion zu finden, ist auch auf andere Wissensbasen anwendbar. Als Einzige Voraussetzung gilt hier eine MySQL-Tabelle, die unterschiedliche RDF-Tripel enthält. Aus dieser Tabelle werden, mit Hilfe des Clusterskripts, alle anderen benötigten Tabellen generiert. Für die Erstellung dieser Tabellen müssen eventuell, je nach Wissensbasis, noch einige Änderungen, zum Beispiel für Blanknodes, welchen momentan keine Beachtung dabei zukommt, implementiert werden. Sind diese Tabellen vorhanden, so kann der DBpedia Relationship Finder auf jede Wissensbasis angewendet werden, um Verbindungen zwischen verschiedenen Ressourcen zu finden. Natürlich ist es auch denkbar, Daten an den Relationship Finder mittels anderer SQL Server oder durch einen SPARQL Endpoint zu senden. Dazu müsste dann einiges an der Funktionsweise verändert werden, die Art der Ausgabe der Ergebnisdaten würde jedoch erhalten bleiben. Der DBpedia Relationship Finder könnte somit die folgende Liste der bereits vorhandenen Benutzeroberflächen zur Betrachtung von RDF-Graphen erweitern:

- Graphen: Aus zusammenhängenden Daten werden Graphen gebildet, die dann untersucht werden können.
- Tabellen: Die Daten werden in tabellarischer Form dargestellt.
- Tripel: Darstellung von Daten als Tripel.
- Zeittafeln: Darstellung von zeitbezogenen Daten in Zeittafeln (zB.: Termindaten, Personendaten).
- Karten: Ortsbezogene Daten können auf Karten dargestellt werden.
- Datensammlungen: gesammelte Daten, zum Beispiel von Personen werden auf einer Internetseite zusammengefasst und anzeigbar gemacht.

Das ausgegebene Verbindungsdiagramm erweitert diese Liste und benutzt dabei selbst Teile dieser Liste. Im DBpedia Relationship Finder werden Pfade in Graphen dargestellt. Er operiert auf einer Datensammlung zu einem - oder mehreren - Wissensgebieten

und zeigt diese Daten an. Auch die Tripel, aus denen die gefundenen Verbindungen hervorgegangen sind, bleiben noch erkennbar. Mit einer solchen Oberfläche können interessante Verbindungen, die auch weit auseinander liegen, zwischen verschiedenen Wissensgebieten gefunden werden. Solche berechnete Verbindungen könnten beispielsweise genutzt werden, um direkte Verbindungen zwischen verschiedenen Daten herzustellen, die vorher nicht in Betracht gezogen wurden.

6 Verwandte Arbeiten

Es existieren bereits andere Arbeiten, mit dem Ziel, Daten strukturiert und somit maschinenlesbar darzustellen. Diese Daten wurden zum Teil ebenfalls aus vorhandenen Internetquellen extrahiert und in RDF-Daten umgewandelt.

6.1 Semantic MediaWiki

Das Semantic MediaWiki[20] ist eine Erweiterung der MediaWiki Software (vgl. Kapitel 2.2). Es soll das MediaWiki in ein semantisches Wiki verwandeln, indem es den Benutzern erlaubt strukturierte Daten in die Artikeltexte einzubringen. Um diese Informationen einzufügen ist die Wikipedia Link Syntax etwas verändert worden: Es werden Typen, welche den Link beschreiben, eingefügt. So wird beispielsweise der Link [[England]] im Artikel von London durch [[is capital of::England]] ersetzt. Es können auch mehrere Typen zugeordnet werden, falls dies nötig sein sollte ([[Typ_1::Typ_2::...::Typ_n::Zielartikel]]). Außerdem können Attribute vergeben werden, um zu einem Datenwert Maschinenlesbarkeit hinzuzufügen: Anstatt im Artikeltext einfach eine Einwohnerzahl einer Stadt zu schreiben, wird [[population:=7,421,328|around 7.5 million]] eingefügt. Mit Hilfe dieser strukturierten Daten besteht die Möglichkeit die Suche stark zu verbessern, indem man die hinzugefügten Daten nutzt. Des Weiteren können viele Wikipedia Kategorien ersetzt werden durch einfache Suchergebnisse, wie zum Beispiel die Kategorie: *Category:Asteroids_named_for_people* würde ersetzt durch eine Anfrage nach *Category:Asteroids, Category:People, Relation:named after*.[49] Die Performance dieser Änderungen sollte sich kaum von der der MediaWiki Software unterscheiden, da alle strukturierten Informationen in von der Wiki Software abgetrennten Datenbanken abgespeichert werden. Lediglich das Abspeichern der semantischen Informationen in einem Artikel benötigt etwas mehr Zeit, da das Parsen des Artikelquelltextes durch die hinzugefügten Informationen geringfügig mehr Zeit braucht, als ohne diese Informationen. Im Vergleich zu den in dieser Arbeit entwickelten Extraktionsmöglichkeiten hat dieser Ansatz den Nachteil, dass die Wikipedia Artikel von Hand angepasst werden müssten. Außerdem müssten die zahlreichen Wikipedia Artikelautoren die Syntaxerweiterungen erlernen, um dieses System zu etablieren. Die in Kapitel 3.5 benannten Möglichkeiten seitens der Wikipedia Artikelautoren für eine bessere Templatextraktion sollten dagegen leichter zu vermitteln sein.

[49] http://ontoworld.org/wiki/Semantic_MediaWiki

6.2 YAGO

YAGO[19] wurde am Max-Planck-Institut für Informatik in Saarbrücken entwickelt. Es bedeutet „Yet Another Great Ontology.“ Die Daten, die YAGO enthält, wurden aus der Wikipedia extrahiert und mit den in WordNet[50] enthaltenen Daten zusammengeführt. In WordNet werden Verben, Substantive und Adverbe in Synonymgruppen eingeteilt. Aus der Wikipedia werden die Kategoriedaten eines jeden Artikels extrahiert, um Aussagen über den Artikel, bzw. die Ressource zu treffen. Die Wissensbasis enthält zur Zeit ca. eine Million Ressourcen und ca. sechs Millionen Fakten über diese Ressourcen. Dabei ist anzumerken, dass hierbei ein hohes Maß an Korrektheit der Daten festgestellt wurde. Es ist möglich mittels eines entwickelten Werkzeuges „Leila“ verschiedene Webseiten und Datenbanken zu durchsuchen und neue Fakten für die YAGO Ontologie zu extrahieren. Diese sollten nur dann zur Ontologie hinzugefügt werden, wenn sie sich nicht mit anderen, bereits enthaltenen, Daten widersprechen. Der YAGO Klassifikationsalgorithmus wurde genutzt um rdf:type-Tripel zu allen DBpedia Ressourcen hinzuzufügen. Der Unterschied zur entwickelten Templateextraktion besteht darin, dass in YAGO keine Inhaltsdaten enthalten sind, sondern lediglich Klassifizierungsdaten.

6.3 Freebase

Freebase[51], entwickelt von Metaweb[52], ist eine Wissensdatenbank gebildet aus Themen der Wikipedia und Musicbrainz[53]. Es wird den Benutzern erlaubt, ohne jede Kenntnis von Programmierung, Artikel zu erstellen oder einfach zu ändern. Momentan umfasst die Freebase Wissensdatenbank Informationen über mehr als 22000 Filme, ca. 350000 Musikalben und ca. 350000 Personen. Wird eine Suche in Freebase ausgeführt, so kann der Benutzer nach Erhalt aller Suchergebnisse diese nach selbst gewählten Filtern einschränken. Diese Filter werden in ein Eingabefeld eingegeben und automatisch werden Vorschläge für mögliche Filter erteilt. Freebase befindet sich momentan noch im Alpha Status, soll aber bei dem Erscheinen der Beta Version frei für jeden zugänglich sein. Das Freebase Projekt folgt einem ähnlichen Ansatz wie das DBpedia Projekt. Es wurden Informationen von zahlreichen Internetseiten, darunter auch die Wikipedia, extrahiert. Des Weiteren sollen auch Informationen aus der DBpedia Datenbasis einge-

[50] http://wordnet.princeton.edu/
[51] http://www.freebase.com/
[52] http://www.metaweb.com/
[53] http://www.musicbrainz.org

fügt werden. Der Unterschied besteht darin, dass die Benutzer in der bereitgestellten Benutzeroberfläche direkt Daten verändern oder hinzufügen können.

6.4 Wikipedia3

Die Wikipedia3 Daten werden, wie die DBpedia Daten, aus den Wikipedia Artikel Seiten gewonnen. Es werden allerdings lediglich alle Wikipedia internen Verknüpfungen von einer Seite extrahiert. Darunter fallen auch Verknüpfungen auf Wikipedia Kategorien. Die Wikipedia3 Datenbasis umfasst derzeit ca. 47 Millionen Tripel. Die Ergebnisse dieser Extraktion sind frei verfügbar, die Software, von System One[54] entwickelt, allerdings nicht. Für die Daten von Wikipedia3 wurden bisher noch keine Benutzeroberflächen zum betrachten - oder um Anfragen zu stellen - bereit gestellt.

[54]http://www.systemone.at/

7 Zusammenfassung und weitere Arbeit

Das Ziel der Arbeit war es, die in den Formatvorlagen der Wikipedia Artikeltexte vorhandenen Daten zu extrahieren und strukturiert abzuspeichern. Dies ist bis auf wenige auftretende Fehler (vgl. Kapitel 3.5) gelungen. Natürlich besteht hierbei die Möglichkeit die Extraktionsalgorithmen noch zu verbessern und noch weniger Fehler zuzulassen. Des Weiteren könnte der Extraktionsalgorithmus noch so erweitert werden, dass verschiedene Synonyme von Prädikaten gebildet werden, um Unterschiede zwischen Templates in diesem Bereich aus dem Weg zu räumen. Es können außerdem Vorschläge an die Wikipedia Community gegeben werden, wie die Templates verändert werden müssen, bzw. wie neu erstellte Templates aussehen müssen, um die Datenextraktion noch besser zu gestalten.

Des Weiteren wurde mit der Arbeit am DBpedia Relationship Finder und seines zugrundeliegenden Cluster-Algorithmus eine Möglichkeit geschaffen die extrahierten Daten zu betrachten und Verbindungen zwischen ihnen zu finden. Für eine solche Verbindungssuche existierte bisher noch keine Benutzeroberfläche. Auch hier kann die Arbeit fortgeführt werden, indem man den DBpedia Relationship Finder so abändert, dass er mit jeder Wissensbasis ohne Weiteres benutzt werden kann. Die Zeit, die für eine Verbindungsberechnung benötigt wird, sollte ebenfalls noch verkürzt werden.

Die Entwicklung von noch benutzerfreundlicheren und performanteren Oberflächen sollte auch weiter vorangetrieben werden. Die in dieser Arbeit vorgestellten Benutzeroberflächen brauchen oft schon eine URI, um den Benutzer zu den Daten zu bringen. Diese URI sollte durch eine effiziente Suche ersetzt werden. Außerdem sollte die von der DBpedia gesammelte Datenbasis noch um andere Ontologien erweitert werden, um eine noch umfassendere und vollständigere Wissensbasis herzustellen.

Die in dieser Arbeit entwickelten Möglichkeiten, Daten aus der Wikipedia zu extrahieren, liefern die erste große strukturierte Wissensbasis für Allgemeinwissen, wie es in der Wikipedia zu finden ist. Nur dadurch kann die Idee des Semantic Web unter den Benutzern des Internets so populär gemacht werden, dass sich dieser Ansatz eines Tages durchsetzt.

Kurzzusammenfassung

In der vorliegenden Arbeit soll gezeigt werden, welche Möglichkeiten es gibt, um strukturierte Informationen aus der Wikipedia Enzyklopädie zu gewinnen. Außerdem werden verschiedene Benutzeroberflächen zum Betrachten der extrahierten Daten bzw. zur Stellung von Anfragen an die extrahierten Daten vorgestellt. Es wird gezeigt, wie die Extraktion der Informationen aus der Wikipedia funktioniert und wie man eine Messung durchführt, um den Vernetzungsgrad der Daten zu ermitteln. Aufbauend auf den Ergebnissen dieser Vernetzungsberechnung wurde eine Benutzeroberfläche konstruiert, mit der der Benutzer selbst Verbindungen zwischen den Daten finden kann.

Literatur

[1] Auer, S., C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak und Z. Ives: *DBpedia: A Nucleus for a Web of Open Data*. Erscheint auf der 6. International Semantic Web Conference, ISWC 2007, Busan, Juli 2007.

[2] Auer, S., S. Dietzold, J. Lehmann und T. Riechert: *OntoWiki - A Tool for Social, Semantic Collaboration*. In: *Proceedings of the WWW-07 Workshop on Social and Collaborative Construction of Structured Knowledge Neural-Symbolic Learning and Reasoning*, 2007.

[3] Auer, S., S. Dietzold und T. Riechert: *OntoWiki - A Tool for Social, Semantic Collaboration*. In: *The Semantic Web - ISWC 2006: 5th International Semantic Web Conference, Athens, USA, November 5-9, 2006, Proceedings*, 2006.

[4] Auer, S. und J. Lehmann: *What have Innsbruck and Leipzig in common? Extracting Semantics from Wiki Content*. In: Franconi, E., M. Kifer und W. May (Hrsg.): *Proceedings of the European Semantic Web Conference, ESWC2007*, Bd. 4519 d. Reihe *Lecture Notes in Computer Science*. Springer-Verlag, Juli 2007.

[5] Beckett, D.: *RDF Primer*. W3C Recommendation, World Wide Web Consortium (W3C), 10. Feb. 2004. http://www.w3.org/TR/rdf-primer/.

[6] Beckett, D.: *RDF Test Cases*. W3C Recommendation, World Wide Web Consortium (W3C), 10. Feb. 2004. http://www.w3.org/TR/rdf-testcases/.

[7] Berners-Lee, T., Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer und D. Sheets: *Tabulator: Exploring and Analyzing linked data on the Semantic Web*. In: *Proceedings of the 3rd International Semantic Web User Interaction Workshop*, Athens, Georgia, Nov. 2006.

[8] Berners-Lee, T., D. Fensel, J. A. Hendler, H. Lieberman und W. Wahlster (Hrsg.): *Spinning the Semantic Web: Bringing the World Wide Web to its full potential*. MIT Press, Massachusetts, 2003.

[9] Berners-Lee, T., J. Hendler und O. Lassila: *The Semantic Web*. Scientific American, 284(5):34–44, 17. Mai 2001.

[10] Broekstra, J., A. Kampman und F. van Harmelen: *Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema*. In: Horrocks, I. und J. Hendler (Hrsg.): *The Semantic Web - ISWC 2002. First International Semantic Web Conference, Sardinia, Italy, June 9-12, 2002, Proceedings*, Bd. 2342 d. Reihe *Lecture Notes in Computer Science*, S. 54–68. Springer, 2002.

[11] Dostal, Melzer, Jeckle und Zengler: *Semantic Web*. OBJEKTspektrum, 5:30–36, Mai 2004.

[12] Gamperl, J.: *AJAX - Web 2.0 in der Praxis*. Galileo Press, Bonn, 2006.

[13] Geroimenko, V. und C. Chen: *Visualizing the Semantic Web: XML-based Internet and Information Visualization*. Springer-Verlag, London, 2006.

[14] Gruber, T. R.: *A Translation Approach to Portable Ontology Secifications*. In: *Knowledge Acquisition*, Bd. 5, S. 199–220. Academic Press, Juni 1993.

[15] McGuinness, D. L. und F. van Harmelen: *OWL Web Ontology Language*. Techn. Ber., World Wide Web Consortium (W3C), Feb. 2004. http://www.w3.org/TR/owl-features/.

[16] Oren, E.: *SemperWiki: A Semantic Personal Wiki*. In: Decker, S., J. Park, D. Quan und L. Sauermann (Hrsg.): *Proceedings of Semantic Desktop Workshop at the ISWC2005, Galway, Ireland*, 2005.

[17] Prud'hommeaux, E. und A. Seaborne: *SPARQL Query Language for RDF (Working Draft)*. W3C Working Draft, World Wide Web Consortium (W3C), 2006.

[18] Rothfuss, G. und C. Ried: *Content Management mit XML*. Springer, 2003.

[19] Suchanek, F. M., G. Kasneci und G. Weikum: *Yago: A Core of Semantic Knowledge - Unifying WordNet and Wikipedia*. In: *16th International World Wide Web Conference (WWW 2007)*, Banff, Canada, 2007. IW3C2.

[20] Völkel, M., M. Krötzsch, D. Vrandecic, H. Haller und R. Studer: *Semantic Wikipedia*. In: *Proceedings of the 15th international conference on World Wide Web, WWW 2006, Edinburgh, Scotland, May 23-26, 2006*, Mai 2006.

Abbildungsverzeichnis

1 Semantic Web Stack 4
2 Beispiel der Wikipedia Infobox: Leipzig - Quelltext und gerenderte Ausgabe 13
3 Template Extraktionsalgorithmus 19
4 Disco Hyperdata Browser Benutzeroberfläche 27
5 Tabulator Benutzeroberfläche 29
6 Sesame Screenshot Leipzig 31
7 Benutzeroberfläche des DBpedia Graph Pattern Builder 33
8 Benutzeroberfläche des OpenLink Visual Query Builder 34
9 Benutzeroberfläche von search DBpedia.org mit Suchbegriff Leipzig . . 35
10 Benutzeroberfläche WikiStory 36
11 Cluster-Algorithmus 39
12 Ergebnisse des Cluster-Algorithmus - Ressourcenverteilung 40
13 Anzahl unterschiedlicher Ressourcen bei unterschiedlicher Entfernung zur Ursprungsressource 41
14 DBpedia Relationship Finder - Benutzeroberfläche bei Start 42
15 DBpedia Relationship Finder - Vorberechnung einer Verbindung mittels Cluster-Algorithmus - Resultat 44
16 DBpedia Relationship Finder - Benutzeroberfläche Ergebnisanzeige . . 45
17 DBpedia Relationship Finder - Autovervollständigung 46
18 DBpedia Relationship Finder - Vorberechnung einer Verbindung 47
19 DBpedia Relationship Finder - Infoboxausgabe 49
20 DBpedia Relationship Finder - gespeicherte Anfragen 50
21 DBpedia Relationship Finder - Funktionsweise 51

Tabellenverzeichnis

1 Erkannte Objektwerte 17
2 Extraktionsergebnisse: Statistik, am häufigsten auftretende Templates und Attribute. 20
3 Extrahierte Attribute aus ausgewählten Templates. 21
4 Datenbasen der DBpedia Extraktion 25

Abkürzungsverzeichnis

AJAX	**A**synchronous **J**avaScript **a**nd **X**ML
bzw.	beziehungsweise
CSS	**C**ascading **S**tyle **S**heet
CSV	**C**haracter **S**eparated **V**alues
CWM	**C**losed **W**orld **M**achine
FOAF	**F**riend **o**f **a** **F**riend
GPL	**G**NU **P**ublic **L**icense
HTML	**H**yper**T**ext **M**arkup **L**anguage
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
JSON	**J**ava**S**cript **O**bject **N**otation
N3	**N**otation **3**
OWL	**W**eb **O**ntology **L**anguage
PHP	**P**HP **H**yperText **P**reprocessor
RDF	**R**esource **D**escription **F**ramework
RDFS	**RDF S**chema
SPARQL	**SPARQL P**rotocol **a**nd **R**DF **Q**uery **L**anguage
SQL	**S**tructured **Q**uery **L**anguage
URI	**U**niform **R**esource **I**dentifier
usw.	und so weiter
vgl.	vergleiche
W3C	**W**orld **W**ide **W**eb **C**onsortium
WWW	**W**orld **W**ide **W**eb
XHTML	e**X**tensible **H**yper**T**ext **M**arkup **L**anguage
XML	e**X**tensible **M**arkup **L**anguage
XSLT	e**X**tensible **S**tylesheet **L**anguage **T**ransformation
YAGO	**Y**et **A**nother **G**reat **O**ntology
z.B.	zum Beispiel

Eidesstattliche Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Leipzig, den 25. Juli 2007 _____

Unterschrift