Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Anwendungen 2 - SoSe 2011

Andreas Winschu

SemanticWeb Plattform
Morgen in meiner Stadt

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Inhaltsverzeichnis

1 Einführung in das Themengebiet

1.1 Motivation

Beim Fallen des Begriffs „World Wide Web“ ist es nicht unverständlich, dass dabei sinngemäß ein immens dicht-gesponnenes Netzgebilde von unzähligen Webseiten assoziiert wird. Die Beschreibung „beinahe unzählig“ mag angesichts von mindestens 100 Millionen bekannter Seiten zutreffen, „dicht-gesponnen“ jedoch in keiner Weise. Die ursprünglichen Spinnweben des WWW sind Hyperlinks. Sie stellen einen einfachen Verweis auf eine weitere Ressource dar. Dies funktioniert nur weil Menschen sehr flexibel bezüglich der Informationsaufnahme sind und so aus der gesamten Information der aktuellen Seite den Kontext des Verweises einordnen können.

Schon früh genug kam die Vision diesen Kontext der Maschine ebenfalls zu Verfügung zu stellen und die Information, die sich hinter dem Hyperlink verbirgt eindeutig zu beschreiben. Dadurch erhofft man sich eine intelligentere Hilfestellung bei der Navigation des Benutzers in der Datenflut der unzähligen Webressourcen. Die globale Vision des neuen semantischen Webs sieht vor, dass jede öffentliche Webressource ihre Textinhalte auf diese Weise mittels standardisierter Mechanismen bereitstellt. Dies mag sicherlich weit entfernt von der heutigen Realität sein, doch die einzelnen aus dieser Vision entstandenen Methodiken bezüglich der Formalisierung und Analyse der semantischen Beziehungen und das Repräsentieren derer in geeigneten Datenstrukturen können bereits heute auf einzelne Domainbereiche angewendet und für geschlossene Plattformen verwendet werden.

Die „Semantic Web“ Lehre definiert mittlerweile eine breite Palette an solchen Methodiken und Datenstrukturen, jedoch kann sie kein allgemeines Rezept dafür liefern, wie man in jedem konkreten Fall einen Mehrwert aus der neu gewonnenen formalen Dateninterconnection gewinnt. Für jede neue Domain muss die Fragestellung nach der geeigneten Datennavigation und Repräsentierung, sowie dem Gewinn des impliziten Wissens aus den explizit definierten Beziehungen neu beantwortet werden.

Die Plattform „Morgen in meiner Stadt“ bietet ein Lernportal für Schüler und erlaubt es Inhalte zu verschiedenen Forschungsprojekten von diversen Industriefirmen den Schülern zur Verfügung zu stellen. Ferner können von den Lehrern beispielsweise weitere Inhalte zu den Themen dieser Projekte bereitgestellt werden und Aufgaben für die Schüler definiert werden. Voraussichtlich entstehen dabei semantische Querverbindungen zwischen den unterschiedlichen Inhalten, welche bei der Recherche über ein bestimmtes Thema relevant sein können. Als Forschungsthema stellt sich dieses Projekt die Aufgabe herauszufinden, inwiefern Semantic Web Technologien genutzt werden können um diese Querverbindungen implizit aufzudecken und ein benutzerfreundliches Interface zu Navigation durch diese Verbindungen bereitzustellen.

1.2 Grundlagen

Semantic Web

Wie bereits aus der Einleitung deutlich wurde, versteht man unter Semantic Web ein maschinenlesbares Web, bei dem die einzelnen Inhalte annotiert mit Metadaten öffentlich bereitgestellt werden.

Diese Ideologie steht im Gegensatz zu herkömmlichen datenbankzentrierten Webapplikationen. Diese herkömmlichen Applikationen sind geschlossene Systeme, die mit einem festem Datenschema, welches zentral definiert wird arbeiten. Semantic Web Applikationen sind in ihrer Ideologie

offene Systeme, welche in anspruchvollsten Fällen damit rechnen müssen mit heterogenen Daten aus verschiedenen verteilten Datenquellen umzugehen (Siehe Abb. 1).

Database-driven web applications	Semantic Web applications
Centralised	Decentralised
One fixed schema	Semi-structured
One fixed vocabulary	Arbitrary vocabulary
Centralised publishing	Publish anywhere
One datasource	Many distributed datasources
Closed systems Constraint-based semantics	Open systems Open-world (DL) semantics

Abbildung 1: Semantic Web Ideologie

RDF

Der Versuch möglichst viele unterschiedliche Datentypen in einem festem Schema zu definieren führt zu sehr komplexen Schemas, die zusätzliche schwerfällig gegen Datenänderungen sind. Resultierend aus der oben erläuterten Ideologie der semantischen Applikationen, die eben diese Heterogenität händeln muss, wurde ein spezielles flexibles Datenformat namens RDF konzipiert.

Hierbei werden alle Daten in sog. Tripel abgelegt, welche das fundamentale semantische Bauelement bilden. Dabei besteht jedes Tripel aus einem Subjekt, einem Prädikat und einem Objekt (Abb. 2). Diese Tripel können als simple linguistische Aussagen angesehen werden. [vgl. Segaran u. a. (2009) , Kap. 4]

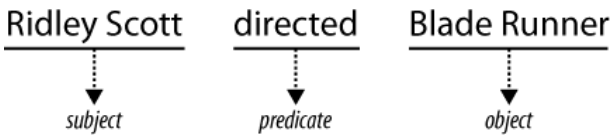

Abbildung 2: RDF Tripel [Segaran u. a. (2009) , Figure 2-1]

Im Grunde ist das entstehende Datenmodell ein Graph aus solchen Trippeln (Abb 3). RDF definiert lediglich eine formale Syntax hierfür und fügt zusätzlich sämtliche weitere Konzepte, welche die Modelle präzise und robust machen hinzu um die Doppeldeutigkeiten bei Übertragung semantischer Daten zu vermeiden.

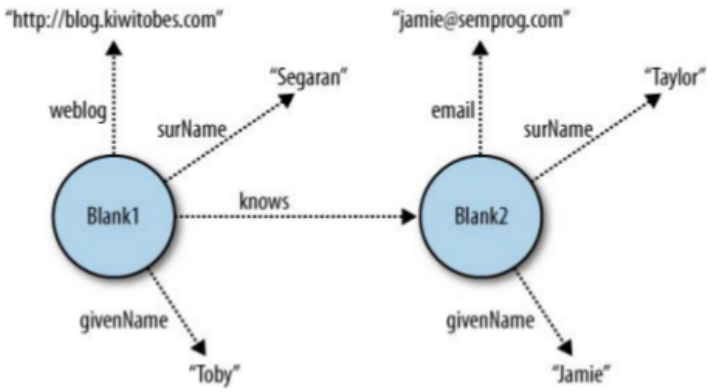

Abbildung 3: RDF Graph [Segaran u. a. (2009) , Figure 4-2]

Werden die RDF Daten etwa in einem RDF Store persistiert, so kann auf sie mit Hilfe von einer graphbasierten Abfragesprache namens SPARQL zugegriffen werden.

Ontologie

Das RDF Tripel aus der Abb. 2, „Riddley Scott directed Blade Runner“ zeigt ebenfalls, dass die Möglichkeit Wissen in so einer Form abzulegen allein nicht ausreichend ist. Diese Information kann ohne entsprechenden Kontext nicht verarbeitet werden. Der Kontext kann natürlich in den Programmcode der Applikation eingebaut werden, doch das Konzept der semantischen Webs möchte idealerweise auch allen Anderen erlauben, die gleichen Schlussfolgerungen aus den existierenden Information ziehen zu können.

Ontologien erlauben es dieses zu erreichen, indem sie formale Regeln aufstellen, welche sich auf die RDF Daten beziehen. Bezüglich des Beispiels oben könnte eine Ontologie festlegen, dass das Prädikat „directed“ nur Subjekte vom Typ „Person“ mit Objekten vom Typ „Movie“ verbinden kann. Weiterhin erlauben Ontologien implizite Regeln zu definieren, sodass weitere Informationen erschlossen werden können, ohne dass sie explizit in das System eingegeben werden müssen. Als simple Veranschaulichung solcher Regeln, kann die typische „If . . . Then . . . “ Aussage genommen werden, welche für das zuvor erwähnte Beispiel wie folgt lauten könnte: „If someone directed a Movie, then he is a director“.

Einen Rückschluss von einer solchen Ontologie auf die einfachen Wissensaussagen nennt man Inferenz oder Reasoning. [vgl. Segaran u. a. (2009), Kap. 6]

2 Vorstellung vergleichbarer Arbeiten

2.1 ActiveRDF

Es wurde gezeigt, dass die Entwicklung von Semantic Web Applikationen in erster Linie das Arbeiten mit RDF Daten nach sich zieht. Ebenfalls wurde deutlich, dass dieses Format vom Konzept her für möglichst heterogene Daten ausgelegt ist. Diese gilt es aber nicht nur dauerhaft in einem maschinenlesbaren Format abzulegen, sondern auch in einer Softwareapplikation kontextspezifisch zu verarbeiten. Während das Kodieren der RDF Daten tripelbasiert ist, wird die Mehrheit der heutigen Software in einem objektorientierten Paradigma entwickelt.

Für relationale Datenbanken existieren schon lange Frameworks, wie Hibernate, Doctrine oder ActiveRecord, welche ein objektrelationales Mapping erzeugen. Ein ähnliches Verfahren auf das RDF Paradigma anzuwenden erweist sich jedoch nicht als unproblematisch. Das Digital Enterprise Research Institute[1] zeigt in [Oren u. a. (2007)] sowie in [Oren und Delbru (2006)] eine stark idealisierte Lösung und verdeutlicht ausführlich die Diskrepanzen zwischen den beiden Weltanschauungen.

RDF im Gegensatz zu OO

Folgende Gegenüberstellung zeigt zusammengefasst die wesentlichen Unterschiede der objektorientierten Weltanschauung und RDF-Datenmodellierung:

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

- Klassenzugehörigkeit und Vererbung - In dem meisten populären OO-Sprachen ist ein Objekt exakt einer bestimmten Klasse zugeordnet. Dessen Zugehörigkeit wird während der Instanziierung festgebunden. Ferner kann meist nur von einer Klasse vererbt werden. Im RDF Schema kann eine Ressource zu mehreren Klassen gehören. Zusätzlich ist diese Bindung nicht fest. Sie wird durch die spezielle Eigenschaft rdf:type, sowie die Menge der aktuellen weiteren Eigenschaften, welche zu einem bestimmten Zeitpunkt zu der Ressourse gehören, festgelegt. Zusätzlich ist Mehrfachvererbung (inklusive zyklischer Vererbung) zugelassen
- Attribute - In OO-Sprachen werden Attribute lokal innerhalb ihrer Klasse festgelegt und haben üblicherweise einen einzigen Datentyp. Im Gegensatz dazu werden bei RDF Attribute als einzelne Eigenschaften angesehen, wodurch sich eigenständige Entitäten sind, welche von mehreren Ressourcen verwendet werden können. Zusätzlich können ihre Werte einen unterschiedlichen Datentyp aufweisen
- Flexibilität - OO-Systeme erlauben es typischerweise nicht, dass sich Klassendefinition zur Laufzeit verändern, während RDF für Integration mit heterogen Daten von variierenden Datenquellen konzepiert ist, wo sowohl Schema als auch Daten sich zur Laufzeit weiterentwickeln.

[siehe Oren u. a. (2007), Kap 2]

Lösung

Der Lösungsansatz von DERI konzentriert sich auf dynamisch-typisierte Scriptsprachen, wobei Ruby als favorisierte Plattform für die Implementierung ausgewählt wurde. Dieser Ansatz bedient sich der Technik zur Erstellung einer sog. Domain Specific Language (DSL). Hierbei werden die Schranken der zur Verfügung stehenden Programmiersprache durch Metaprogrammierung und Laufzeitintrospektion dahingehend aufgelöst, sodass eine Anpassung an die aktuelle Problematik (Domain) stattfindet.

Ruby erlaubt es die oben angeführten Probleme aufgrund einer Reihe optionaler Eigenschaften zu lösen. So ist es beispielsweise erlaubt instanzspezifisches Verhalten für einzelne Objekte zu definieren und somit von ihrer ursprünglichen Klassendefinition abzuweichen. Ferner besteht die Möglichkeit die einst definierte Klassenhierarchie zu überschreiben. Es kann demnach eine Veränderung der Sprache und der Daten zur Laufzeit stattfinden.

Das Mapping zwischen RDF und dem OO-Paradigma wird in ActiveRDF auf den ersten Blick intuitiv vorgenommen, wobei für Klassen, Objekte und Attribute entsprechende Gegenüber im RDF gefunden werden. Im Detail wird jedoch nur die Illusion echter Objekte, mit gekapselten Daten, aufrechterhalten. Tatsächlich wird eine virtuelle Schnittstelle zum Zugriff auf die Daten erzeugt. Die vermeintlichen ActiveRDF Objekte besitzen tatsächlich keine echten Attribute oder Methoden, sondern fangen die Aufrufe an sich immer mit einem 'MethodeNotFoundError' ab. Folgt der Aufrufer einer bestimmten, der erwarteten Daten entsprechenden Konvention, so wird der Aufruf in ein entsprechend gültiges Query an den RDF-Store übersetzt.

[siehe Oren u. a. (2007), Kap 3]

Fazit

Da Semantic Web Konzeption letztendlich das Umgehen mit RDF Daten impliziert, wurde das ActiveRDF Projekt vorgestellt, welches eine unkonventionelle Methodik einführte um diese auf

Objektorientiertes Paradigma zu mappen. Auf den ersten Blick mag diese unkonventionelle Lösung als nicht unbedingt vertrauenswürdig und stabil erscheinen als klar wird, dass in die Kernkomponenten der Programmiersprache eingegriffen wird und diese so verändert wird, dass man ein offenbar unvorhersehbares Verhalten mit Seiteneffekten erzeugt. Dennoch muss man sich vor Augen führen, was die Alternative bedeutet. Ohne eine solche DSL, welche von außen den Eindruck aufrechterhält, man habe immer noch eine objektorientierte Schnittstelle vorliegen, müsste man sich auf die Lowebene begeben und nah an dem RDF Format arbeiten, wobei nur simple Datentypen verwendet werden könnten, was nicht nur eine mühselige Arbeit wäre, sondern auch zu schlechter Verständlichkeit des Quellcodes führen würde.

2.2 Faceted Browsinseg

Eine saubere Lösung bezüglich des Umgehens mit Semantic Web Daten in RDF zu finden, ist bei Weitem nicht die einzige Problematik. Viel interessanter ist die Frage danach, welchen Mehrwert die eigene Applikation aus dieser Technologie ziehen kann. Für geschlossene Systeme liegt diese Antwort meist in semantischer Suche bzw. semantischer Navigation, welche dem Anwender eindeutigere Suchanfragen und treffsicherere Ergebnisse erlauben soll.

Es existieren vier Interfacetypen für Navigation in RDF Daten: Stichwortsuche, explizite Anfragen, Graphenvisualisierung und Faceted Browsing. [Oren u. a. (2006)]geht dabei Näher auf Faceted Browsing ein, da laut Untersuchung weiterer Arbeiten zu den obigen Verfahren die Stichwortsuche sich nur für simples Informationsnachschlagen eignet, explizite Anfragen zu tiefe Schema Kenntnis an den Benutzer fordern und Graph Visualisierung nicht für große Datenmengen skaliert.

Funktionsweise

Unter Faceted Browsing versteht man ein filterbasiertes Navigieren in einer Datenmenge. Jeder Informationsraum kann auf mehrere konzeptionell zusammenhängende Dimensionen aufgeteilt werden. Diese Dimensionen werden Facets genannt und repräsentieren wichtige charakteristische Datenmerkmale. Jedes Facet besitzt wiederum mehrere Restriktionselemente, welche vom Benutzer beim Navigieren ausgewählt werden um die Anzahl der Elemente in dem Informationsraum einzuschränken. Betrachtet man beispielsweise Musikdaten, so sind typische Facets, nach denen gefiltert werden kann, hier - „Genre“, „Interpret“ oder „Album“. Die Restriktionselemente sind dabei Werte, welche diese Attribute annehmen können, e.g. „Rock“, „Pop“ oder „Dance“ für das Facet „Genre“.

In der reinen Form eignet sich Faceted Browsing nur bedingt für graphenbasierte Navigation in RDF Daten. Übliche Implementierungen fokussieren sich immer auf einen Ressourcentyp. So entscheidet der Ersteller von so einem Browser, was die geeigneten Facets für eine bestimmte Domain sind und bietet diese als Filter an. Eine Technik, welche das Faceted Browsing auf Daten im Semantic Web anwenden, muss diese Homogenität selbst erkennen und charakteristische Facets im unbekanntem Datenraum finden.

Übertragung auf RDF

In [Oren u. a. (2006)] wird folgender Weg gewählt um die Faceted Browsing Technologie auf RDF zu übertragen:

Im Informationsraum, bestehend aus der Menge aller RDF Tripel, bilden zunächst die RDF Prädikate die Facets und die Objekte die Restriktionselemente. Das Selektieren der Facets

wird exakt auf entsprechende SPARQL Queries abgebildet. Anschließend werden Facet Ranking Algorithmen eingeführt um besonders navigationsstarke Facets zu finden. Diese Algorithmen basieren darauf, dass Faceted Browsing als gleichzeitiges Erzeugen und Traversieren eines Entscheidungsbaumes, dessen Zweige die Prädikate und die dessen Knoten die Restriktionswerte repräsentieren, angesehen wird (Abb. 4).

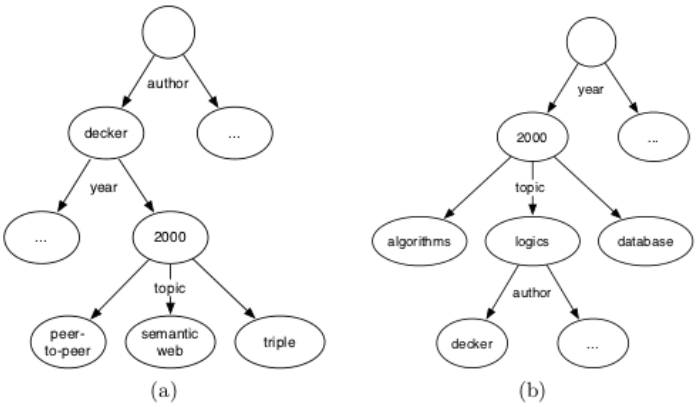

Abbildung 4: Faceted Browsing als Entscheidungsbaum Traversierung [Oren u. a. (2006), Fig.6]

Dabei werden solche Facets als intuitive Facets angenommen, die eine am meisten effiziente Navigation entlang des Entscheidungsbaumes erlauben. Für diese „Navigationsqualität“ definiert [Oren u. a. (2006), Kap 4] folgende Kriterien:

- Predicate balance - Baum Navigation is am effizientesten, wenn der Baum etwa gleich balanciert ist. Ein Prädikat wird demnach als ein effizienterer Facet angesehen, wenn seine bekannten Restriktionselemente etwa gleich oft vorkommen.
- Object cardinality - Als ein intuitives Prädikat wird ein solches Prädikat angesehen, welches eine limitierte Anzahl von Restriktionswerten zur Auswahl hat. Andererseits ergeben sich zu viele Auswahlmöglichkeiten, welche den Benutzer verwirren.
- Predicate frequency - Ein geeignetes Prädikat kommt oft innerhalb der Datenmenge vor. Je mehr unterschiedliche Ressourcen das Prädikat besitzen, desto nützlicher ist es dabei den Informationsraum zu zerteilen. Anders formuliert, wenn ein Prädikat selten vorkommt, betrifft eine Auswahl eines Restriktionswertes für dieses Prädikat, nur eine geringe Untermenge von Ressourcen.

Fazit

Faceted Browsing ist eine Möglichkeit semantische Navigation innerhalb einer Datenmenge zu realisieren, welche für Benutzer sehr intuitiv ist. Letzteres wird durch eine Evaluirung in [Oren u. a. (2006), Kap 6]bekräftigt. Die Effizienz dieses Verfahren hängt maßgeblich davon ab richtige Facets für das aktuelle Datenset aufzudecken. Die hier vorgestellte Methodik zur Aufdeckung der Facets ist absolut plausibel, dennoch ist es offensichtlich, dass sie nicht für jede Situation ideal sein kann. Es ist durchaus vorstellbar, dass Attribute, die zwar ideal für die Navigation im RDF Graphen sind, für bestimmte Benutzer absolut bedeutungslos sein könnten. Ferner ist nicht zu vernachlässigen, dass laut [Oren u. a. (2006), Kap4.3] eine Neuberechnung des Rankings nach jeder Filterauswahl für das gefilterte Ergebnis stattfinden muss, was sich kritisch auf das Echtzeitverhalten dieser Methode ausführt.

2.3 LarkC

Semantische Navigation, welche dem Benutzer erlaubt seine Anfragen präzise zu formulieren, funktioniert am besten, wenn viele Metainformation über die Daten bekannt sind. Es ist also wünschenswert, dass ein System nicht nur solche Daten zu einer Eingabe finden kann, die eine explizite semantische Annotation erhalten haben, sondern auch implizit aus vorher definierten Regeln logische Schlussfolgerungen treffen kann und so weitere Informationen aufdecken kann, die aus diesen Schlussfolgerungen resultieren. In diesem Kontext wurde in 1.2 der Begriff Reasoning in Zusammenhang mit Ontologien eingeführt.

Verständlicher Weise sind solche logischen Operationen außerordentlich teuer. [Fensel u. a. (2008)] behauptet, dass einige sehr fortgeschrittene Szenarien Reasoning über 10 Milliarden RDF Tripel in weniger als 100ms erfordern. Aktuelle logikbasierte Reasoning Systeme können mit dieser Anforderung nicht mithalten. Diese Diskrepanz resultiert aus den zugrundeliegenden Annahmen der Rechenlogik, welche unter anderem auf Vollständigkeit und Korrektheit der Inferenzregeln beruhen. Es wird also angenommen, dass Axiome die absolute Wahrheit wiederspiegeln und anhand derer wird versucht das komplette Datenset aufzudecken, auf welches die Axiome zutreffen. Im Webkontext sind Informationen unzuverlässig von Anfang an, was sogar einer korrekten Inferenzengine nicht immer erlaubt die absolute Wahrheit zu erschließen.

Ansatz

Die LarKC[2] Plattform positioniert sich zwischen traditionellem Suchen (Information Retrieval) und traditionellem Reasoning. In Information Retrieval(IR) Kontext misst man den Erfolg des Suchergebnisses mit Recall und Precision. Der Recall gibt an, wie viele der potentiell relevanten Ergebnisse tatsächlich gefunden wurden, während Presicion angibt, wie viele der tatsächlich gefundenen Ergebnisse relevant zu der Anfrage waren. Höhere Precision führt zu niedrigerem Recall und umgekehrt, wodurch eine bestimmte Erfolgsgrenze entsteht, über die IR Verfahren nicht hinauskommen(Abbildung 5, grau).

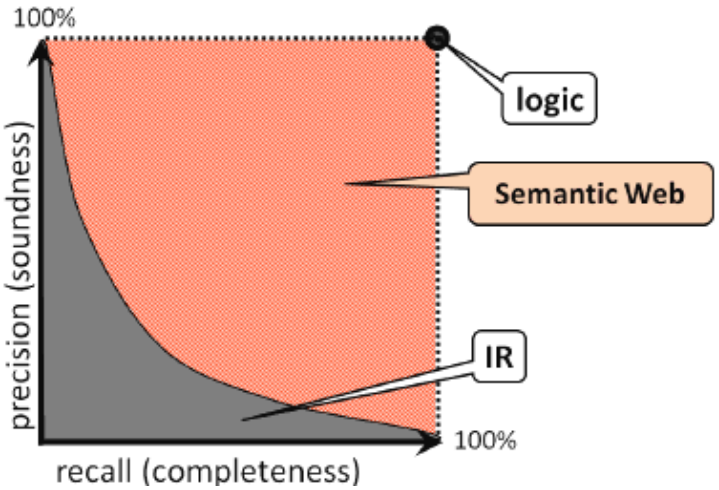

Abbildung 5: Positionierung von LarKC zwischen IR und Logiksuche [Krummenacher u. a. (2011)], Figure 1

Traditionelles Reasonig, welches auf Rechenlogik basiert, fordert wie bereits erläutert, 100%-ige Korrektheit und Vollständigkeit, was im IR Kontext 100% Precision und Reacall bedeutend würde.

[2] http://www.larkc.eu/

In LarKC wird der Reasoningprozess mit Information Retrieval(IR) Methodiken der Keywordsuche derart verknüpft, dass kleinere Datenmengen der verfügbaren Information, die dennoch ein akzeptables Maß an Übereinstimmung zu dem gestellten Problem liefern, als Resultat selektiert werden. Das Ergebnis soll dadurch zwar oberhalb der Precision/Recall Kurve positioniert werden (Abbildung 5, rot), jedoch unter der absoluten Idealgrenze.

Die Plattform verfolgt dabei eine bestimmte Strategie, welche ein Grundgerüstalgorithmus aus fünf verschiedenen Phasen definiert, welche nacheinander ablaufen. Die einzelnen Phasen können durch pluginbasierte Komponenten realisiert werden. Die Kernplattform sorgt dafür dieser Algorithmus koordiniert auf multiple Rechnerknoten zu parallelen Berechnung verteilt werden kann, wodurch eine Skalierbarkeit für überdimensional große Datenmengen erreicht werden soll. [siehe Krummenacher u. a. (2011), Kap 1]und [Fensel u. a. (2008) Kap 2.3]

Das algorithmische Schema sieht wie folgt aus:

- Retrieve - Frage rohen Inhalt mit Assertions ab überall wo ein Beitrag zur Lösung vermutet wird.
- Abstract - Abstrahiere in die für das heterogen Reasoning benötigte Form.
- Select - Wähale den am meisten versprechenden Ansatz zu Inferenz zuerst.
- Reason - Führe ein Reasoning mit mehreren Mitteln durch
- Decide - Entscheide ob genügend akkurate und präzise Lösungen gefunden wurden, versuche gegebenenfalls weiter falls das Ergebnis nicht zufriedenstellend war und die Zeit es zulässt.

[siehe Fensel u. a. (2008) Kap 2.2]

WebPIE

Da LarKC lediglich ein Grundgerüst für eine skalierbare Architektur liefert, wird die eigentliche Aufgabe durch die Engines übernommen, welche an die einzelnen Phasen des Algorithmus einschließen. WebPIE[3] ist eine solche Engine, welche sich des bekannten MapReduce Frameworks Hadoop[4] bedient um die Reasoning Tasks auf mehrere Cluster zu verteilen.

Inferenz Regeln werden typischerweise rekursiv angewendet, wobei die neu erschlossenen Statements, wieder zu der Eingabe hinzugefügt werden und erst angehalten wird, wenn keine weiteren Schlussfolgerungen abgeleitet werden können. Eine Beispielregel ist:

```
if (<?p isA transitiveProperty>, <?a ?p ?b>, <?b ?p ?c>) then
<?a ?p ?c>
```

Um diese Regeln anzuwenden muss ein Join zwischen allen Statements, welche auf die Pattern der linken Seite der Regel matchen, durchgeführt werden. Beim Eintreten dieses Falls wird das Statement auf der rechten Seite generiert. Ein Ontolgie Set kann unzählige dieser Regeln enthalten. Traditionelle Inferenzengines versuchen dieses Problem auf einer einzelnen Maschine zu lösen. Beim Versuch das Ganze zu parallelisieren entstehen diverse Probleme, die weit von trivial sind und durch WebPIE angegangen werden:

- Load Balancing: Parallele Joins werden dadurch erreicht, dass die Regeln nach bestimmten Kriterien in separate MapReduce Job's umverteilt werden.

[3] http://www.few.vu.nl/ jui200/webpie.html
[4] http://hadoop.apache.org/

- Doppelte Inferenzen: Aussagen können auf Basis mehrerer Inputs und Regeln getroffen werden, sodass die gleiche Aussage auf mehrere Weisen erschlossen werden kann. Dieses Problem wurdn durch die Grouppierung der Statements durch die potentiell inferenzierten Statements angegangen.
- Rekursives Anwenden der Regeln: Die Reihenfolge der Regelanwendung wird optimiert um die Iterationsanzahl zu reduzieren.

[siehe Urbani u. a. (2010a), Kap. 2]

Fazit

LarKC und vor allem WebPIE nehmen sich zum Ziel Reasoning durch parallele Verarbeitung in großen Datensets zu ermöglichen. Natürlich kann jede Aufgabe in kürzerer Zeit gelöst werden, wenn man Sie parallelisiert, nun bleibt dennoch bedenklich wie weit man diese Parallelisierung treiben kann. Bekannte Webapplattformen, welche eine hohe Besucheranzahl vorweisen tragen schon heute Kosten in Millionenhöhe. Wenn es also tatsächlich stimmt das einige Inferenzszenarien, wie in den obigen Quellen angeführt ist, Reasoning über mehrere Millionen Tripel benötigen, stellt sich im Kern die Frage ob Reasoning, bedenklich der Tatsache, dass solche Anfragen ebenfalls parallel durch mehrere Benutzer gestellt werden könnten, überhaupt tragbar ist.

2.4 iQser GIN Plattform

Semantische Inhalte entstehen nicht von selbst, wodurch ein sehr großer Aufwand an die Redakteure einer semantischen Plattform entsteht. Obwohl einige Beziehungen aus einem relationalem Schema in RDF Verknüpfungen umgewandelt werden könnten, so liegen in den meisten Fällen die Hauptinformationsquellen als Texte vor. Die Textmining Disziplin beschäftigt sich mit der Aufgabe semantisches Wissen aus lexikalischen Inhalten zu extrahieren. Die Vision Texmining für eine Bottom Up Analyse zu automatisierten Erstellung einer Ontolgie bzw. RDF Schemas zu verwenden ist daher einleuchtend.

Das dies nicht unmöglich ist beweist beispielsweise die iQser GIN Plattform, welche eingesetzt in einer IT Infrastruktur, Inhalte aus diversen Datenbeständen eines Unternehmens (e.g Email, Wikis, ERP-Systeme) extrahiert und ein, wie es in [Wurzer (2010)] heißt „`übergeordnetes Datenmodel, welches als eine flache Ontlogie angesehen kann`"aus diesen Inhalten ableitet. Diese vollautomatische semantische Analyse wird durch [Wurzer (2010)] als ein Alleinstellungsmerkmal der Plattform bezeichnet. Tatsächlich lässt sich kaum eine andere vollständige Plattform für das automatische Ontlogylearning finden, Verfahren dazu werden aber bereits längere Zeit erforscht und sind beispielsweise durch [Buitelaar u. a. (2005)] zusammengefasst.

3 Abgrenzung zu eigenen Arbeitszielen und / oder Methoden

In dieser Ausarbeitung wurden exemplarisch Projekte gezeigt, welche sich den Kernmethodiken und Lösungen der Semantic Web Ansätze widmen. Daraus wird deutlich, dass die Etablierung eines solchen semantischen Ecosystems keineswegs ein wohldefinierter Arbeitsprozess ist. Auch wenn bereits vollständige Fullstackframeworks wie Jena oder Sesame existieren, gehen diese immer davon aus, dass mit deren Hilfe eine neue Applikation vom Grund auf erstellt wird. In der Realität aber, hat man es, wie im Fall von „Morgen in meiner Stadt“ mit bestehenden

Webapplikationen zu tun, welche auf relationale Datenschemas zugreifen. Das Ziel ist es ein solchen bestehenden realen Datensatz mit Hilfe von semantischen Technologien zu annotieren um eine intelligente Datennavigation zu erlauben.

Ebenfalls wird meistens die Frage vernachlässigt, woher die semantischen Beschreibungen entstanden sind und einfach von einem bestehenden RDF Datenset ausgegangen. Bottom Up Analyse, wie sie in 2.4 vorgestellt wird, kann ein hilfreiches Mittel für die Lösung dieses Problems sein. Hinzuziehend der Tatsache, dass die Metadaten auf diese Weise völlig automatisiert extrahiert würden, kann das Resultat eines solchen Verfahren erstaunlich gut sein. Realistisch gesehen jedoch kann auf diese Weise auf keinen Fall das exakte Wissen aus den zu Grunde liegenden Textinhalten extrahiert werden. Somit kann auf ein Interface zu Annotation der Textinhalte durch Fachexperten nicht verzichtet werden und bildet eine weiteres Ziel der zukünftigen Ausarbeitung. Hier gilt es ein solches Interface zu finden, welches für die Fachexperten die technischen Details von dem semantischen Technologien transparent macht, aber dennoch eine Verknüpfung zu einer existierenden Ontologie bzw. bereits erstellten Metatdaten bietet, so dass bereits existierende Begriffe in den Annotationen wiederverwendet werden können.

2.2 zeigte, dass selbst wenn man einen RDF Datensatz vorliegen hat, die Erstellung der eigentlichen semantischen Navigation oder Suche immer noch viele unbeantwortete Fragen aufweist. Das richtige Facetranking zu ermitteln ist weiterhin ein nicht ausgeschöpftes Beschäftigunsfeld. In [Oren u. a. (2006)]wird unter anderem die Möglichkeit, das Verhalten der Benutzer zu analysieren uns somit Facets höher zu bewerten, welche oft von Benutzern favorisiert werden, angedeutet aber nicht weiter verfolgt. Unbeantwortet blieb ebenfalls die Frage inwiefern und ob eine solche Navigation mit dem Wissen aus einer Ontologie verknüpft werden kann, so dass zu einer vom Benutzer erstellten Restriktionsabfrage, weitere Daten gemäß der Axiome in der Ontologie durch Inferenz erschlossen werden.

Die Semantic Web Idee ist nicht unumstritten und es wird nicht selten behauptet, dass die Probleme, welche durch diese Techniken angegangen versucht werden, auch durch herkömmliche Verfahren gelöst werden können. Gerade bei einer geschlossen Plattform, die sich keiner dritten RDF Quellen bedient und selbst keine zur Verfügung stellt, wird sich für Kunden einer solchen Plattform die Frage auftun, inwiefern hier überhaupt für Semantic Web gesprochen werden kann. Daher ist es in der zukünftigen Arbeit besonders wichtig, für das gefundene semantische Anwendungsszenario zu ermitteln ob dadurch ein Mehrwert, verglichen mit einer herkömmlichen Realisierung, wie etwa Keywordsuche und simplen Taggingsystemen enstanden ist bzw. zu beweisen oder zu wiederlegen, dass die Realisierung dieses Szenarios mit typischen Mitteln der Webentwicklung sich deutlich schwerer realisieren lässt.

4 Zusammenfassung

In dieser Ausarbeitung wurden vergleichbare Arbeiten aus dem Themengebiet „Semantic Web“ vorgestellt. Dabei lieferte das ActiveRDF Projekt eine Antwort auf die Frage, wie man RDF Daten in OO modelieren kann. Mit Faceted Browsing wurde ein Anwendungsfall für Semantische Navigation vorgestellt. Das Projekt LarKC behandelte das Skalierbarkeitsproblem in bei Inferenz Rückschluss in großen Datenset. Und schließlich wurde auf das Problem der manuellen annotierung hingewiese und auf die iQser Gin Plattform hingewiesen, welche dieses mit Hilfe von Bottom Up Reasoning nutzt.

Literatur

[Buitelaar u. a. 2005] BUITELAAR, P. ; CIMIANO, P. ; MAGNINI, B.: Ontology Learning from Text: An Overview. In: BUITELAAR, P. (Hrsg.) ; CIMMIANO, P. (Hrsg.) ; MAGNINI, B. (Hrsg.): *Ontology Learning from Text: Methods, Evaluation and Applications*. IOS Press, 2005

[Fensel u. a. 2008] FENSEL, Dieter ; HARMELEN, Frank van ; ANDERSSON, Bo: Towards LarKC: A Platform for Web-Scale Reasoning. In: *ICSC*, IEEE Computer Society, 2008, S. 524–529. – URL http://doi.ieeecomputersociety.org/10.1109/ICSC.2008.41

[Geroimenko und Chen 2006] GEROIMENKO, Vladimir (Hrsg.) ; CHEN, Chaomei (Hrsg.): *Visualizing the Semantic Web: XML-Based Internet and Information Visualization*. 2. London : Springer, 2006. – ISBN 978-1-85233-976-0

[Kifer u. a. 2005] KIFER, Michael ; BRUIJN, Jos de ; BOLEY, Harold ; FENSEL, Dieter: A Realistic Architecture for the Semantic Web. In: ADI, Asaf (Hrsg.) ; STOUTENBURG, Suzette (Hrsg.) ; TABET, Said (Hrsg.): *RuleML* Bd. 3791, Springer, 2005, S. 17–29. – ISBN 3-540-29922-X

[Kotowski und Bry 2010] KOTOWSKI, Jakub ; BRY, FranÃ§ois: *A Perfect Match for Reasoning, Explanation, and Reason Maintenance: OWL 2 RL and Semantic Wikis*. 2010

[Krummenacher u. a. 2011] KRUMMENACHER, Reto ; TOMA, Ioan ; FENSEL, Dieter ; BREHAR, Raluca ; NEDEVSCHI, Sergiu: Instrumenting and Monitoring the LarKC Reasearch Infrastructure. In: *6th International IST-Africa Conference*, Mai 2011

[Oren und Delbru 2006] OREN, Eyal ; DELBRU, Renaud: ActiveRDF: object-oriented RDF in Ruby, URL http://fparreiras/papers/ActiveRDF.pdf, 2006

[Oren u. a. 2006] OREN, Eyal ; DELBRU, Renaud ; DECKER, Stefan: Extending faceted navigation for RDF data. In: *5th International Semantic Web Conference, Athens, GA, USA, November 5-9, 2006*, 2006

[Oren u. a. 2007] OREN, Eyal ; DELBRU, Renaud ; GERKE, Sebastian ; HALLER, Armin ; DECKER, Stefan: ActiveRDF: object-oriented semantic web programming. In: WILLIAMSON, Carey L. (Hrsg.) ; ZURKO, Mary E. (Hrsg.) ; PATEL-SCHNEIDER, Peter F. (Hrsg.) ; SHENOY, Prashant J. (Hrsg.): *WWW*, ACM, 2007, S. 817–824. – URL http://dblp.uni-trier.de/db/conf/www/www2007.html#OrenDGHD07. – ISBN 978-1-59593-654-7

[Segaran u. a. 2009] SEGARAN, Toby ; EVANS, Colin ; TAYLOR, Jamie: *Programming the Semantic Web*. Beijing : O'Reilly, 2009. – ISBN 978-0-596-15381-6

[Urbani u. a. 2010a] URBANI, Jacopo ; KOTOULAS, Spyros ; MAASSEN, Jason ; DROST, Niels ; SEINSTRA, Frank ; HARMELEN, Frank V. ; BAL, Henri: H.: WebPIE: A Web-Scale Parallel Inference Engine. In: *In: Third IEEE International Scalable Computing Challenge (SCALE2010), held in conjunction with the 10th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid*, 2010

[Urbani u. a. 2010b] URBANI, Jacopo ; KOTOULAS, Spyros ; MAASSEN, Jason ; HARMELEN, Frank van ; BAL, Henri: OWL Reasoning with WebPIE: Calculating the Closure of 100 Billion Triples. In: AROYO, Lora (Hrsg.) ; ANTONIOU, Grigoris (Hrsg.) ; HYVÃ¶NEN, Eero (Hrsg.) ; TEIJE, Annette ten (Hrsg.) ; STUCKENSCHMIDT, Heiner (Hrsg.) ; CABRAL, Liliana (Hrsg.) ; TUDORACHE, Tania (Hrsg.): *Proceedings of the 7th Extended Semantic Web Conference (ESWC)* Bd. 6088. Berlin, Heidelberg : Springer, Mai 2010, S. 213–227

[Urbani u. a. 2009] URBANI, Jacopo ; KOTOULAS, Spyros ; OREN, Eyal ; HARMELEN, Frank van: Scalable Distributed Reasoning Using MapReduce. In: *The Semantic Web - ISWC 2009, Chantilly, VA, USA, October 25-29, 2009* Bd. 5823, Springer, 2009, S. 634–649. – URL http://dx.doi.org/10.1007/978-3-642-04930-9. – ISBN 978-3-642-04929-3

[Wurzer 2010] WURZER, Joerg: iQser GIN Plattform: Intelligente Semantische Middleware. In: *KI – Künstliche Intelligenz* 24 (2010), Nr. 1, S. 83–86. – ISSN 0933-1875

Literaturverzeichnis