Wissensextraktion im Rahmen des Grids

Steffen Metzger
Max-Planck-Institut für Informatik
Campus E1 4
Saarbrücken, Deutschland
smetzger@mpi-inf.mpg.de

Zusammenfassung

Moderne Netzwerke stellen immense Datensammlungen bereit, so dass oft weniger das Beschaffen von Informationen, als vielmehr das Herausfiltern der relevanten Essenz, den größten Aufwand darstellt. Hier können Methoden der Wissensextraktion helfen, essentielle Inhalte aus vorliegenden Daten als höherwertiges Wissen zu extrahieren und damit konkrete Fragestellungen einfacher beantwortbar zu machen. Grid Computing hat zum Ziel verteilte Ressourcen für alle Beteiligten effizient nutzbar zu machen. Bisherige Umsetzungen haben sich dabei primär auf Rechenleistung als Ressource konzentriert. Um auch die im Grid vorliegenden Daten sinnvoll nutzbar zu machen, bietet es sich an, das enthaltene Wissen zu extrahieren und in eine abstrakte Form zu übertragen. Extraktionstechnologien können hierbei von der Gridinfrastruktur profitieren und so eine bessere Effizienz erreichen. Innerhalb dieser Arbeit wird ein generisches Extraktionsframework vorgestellt, sich ergebende Probleme bei der Übertragung in ein Gridumfeld aufgezeigt und erste Schritte zur Anpassung dargelegt.

1. EINLEITUNG

Mit der Entstehung von Computernetzwerken entstand auch der Bedarf entfernte Ressourcen, wie z.B. Rechenleistung und Speicher, aber auch vorhandene Daten und Dienste, in den lokalen Arbeitsprozess einzubinden. Dieses Bestreben brachte das *Grid Computing* hervor, das darauf abzielt, die vorhandenen Ressourcen in sogenannten *virtuellen Organisationen* für die jeweiligen Mitglieder zugänglich zu machen. Bisher konzentrieren sich diese Bemühungen primär auf verschiedene Aspekte zur verteilten Berechnung von Anwendungen, wie z.B. Scheduling, Workflow Management (UNICORE[19], Condor[24]) und Zugriffssicherheit (Globus[13]). Allerdings bietet sich, zur Nutzung von Synergien, eine Zusammenarbeit auch auf Ebene der eigentlichen Inhalte an. Dazu gehört z.B. eine einheitliche globale Suche nach diesen Inhalten. Eine Schlüsselwortsuche auf den Daten selbst stellt hierbei nur den kleinsten gemeinsamen Nenner dar. Die enthaltenen Informationen werden dabei vollständig abgebildet, können aber auch nur dann gefunden werden, wenn explizit mit dem richtigen Schlüsselwort nach ihnen gesucht wird. Beispielsweise wird eine Schlüsselwortsuche nach dem Geburtsort von Albert Einstein zu der Anfrage "Einstein Geburtsort" ein Dokument das lediglich den Satz "Einstein wurde in Ulm geboren" nicht als Ergebnis liefern. Eine abstrakte Wissensrepräsentation dagegen kann solche unterschiedlichen Ausdrucksweisen vereinheitlichen und die Suche nach den eigentlichen Inhalten so erleichtern. Zudem fällt die Suche nach *Beziehungen* über eine Schlüsselwortbasierte Suche schwer. Sind z.B. alle Naturwissenschaftler gesucht, die in Ulm geboren wurden, lässt sich diese Beziehung schwerlich als Schlüsselwortsuche formulieren. Zur Beantwortuntg einer solchen Suche bedarf es Hintergrundwissen zur Identifikation von Naturwissenschaftlern und ihren Geburtsorten. Liegt solches Wissen z.B. in Ontologien vor, kann die Fragestellung dagegen recht einfach als SPARQL Anfrage formuliert werden: "?x hatTyp Naturwissenschaftler . ?x geborenIn Ulm".

Auf der anderen Seite zeigt sich zwar, dass Verfahren der Wissensextraktion auf Texten durch Einsatz komplexerer Sprachanalysemethoden deutlich verbesserte Ergebnisse erzielen können (z.B. Deep Syntactic Analysis in LEILA[21]), dies geht aber in der Regel auf Kosten der Laufzeit([20]), da diese Verfahren einen deutlich höheren Aufwand erfordern als beispielsweise relativ einfache Regeln auf Basis der Zeichenabfolge (z.B. Reguläre Ausdrücke).

In der Kombination beider Technologien bietet sich hier also ein Synergieeffekt an. Einerseits liegen in den Grid-Communitys große Mengen an Daten vor, deren Inhalt auf einem höheren Abstraktionslevel einheitlich und effizient durchsuchbar zur Verfügung gestellt werden soll. Andererseits kann die Wissensextraktion von der verteilten Rechenleistung des Grids profitieren. Für die beteiligten Communitys ergibt sich der Vorteil, dass eine existierende Infrastruktur genutzt wird, um ihre Daten mit effizienten Methoden aufzuwerten und Zugriff auf Wissen anderer Communitys zu erhalten.

Im folgenden wollen wir einen kurzen Einblick in den Stand der Forschung im Bereich der Wissensextraktion und des Grid Computing geben. Anschließend wird in Abschnitt 3 ein generisches Musterbasiertes Verfahren zur Wissensextraktion, das in einem Grid-Projekt (WisNetGrid [14]) Verwendung finden soll, vorgestellt. In Abschnitt 5 besprechen wir letztlich offene Punkte bzw. mögliche Erweiterungen.

GvD Workshop'10, 25.-28.05.2010, Bad Helmstedt, Germany.

2. VERWANDTE ARBEITEN

Zur Realisierung von Gridinfrastrukturen sind verschiedene Middlewarekomponenten entstanden, die sich jeweils Aspekten der verteilten Berechnung von Anwendungen (Jobs) im Grid widmen. Während UNICORE[19] primär den Zugriff auf große Rechencluster über ein Grid anstrebt und hierfür auch ein Job Workflow Management zur Verfügung stellt, widmete sich Condor[24] ursprünglich mehr der Vernetzung von Desktoprechnern, was durch die größere Unbeständigkeit der beteiligten Knoten über ein Job Workflow Management hinaus ein Checkpoint System zur Wiederaufnahme unterbrochener Grid Jobs hervorbrachte. Während beide Systeme den Datentransfer zu einem ausführenden Knoten im Grid und zurück erlauben, bieten Globus[13] und gLite [9] weitergehendere Kontrolle über die Daten im Grid sowie ein z.T. detaillierteres Rechtesystem.
In Deutschland koordiniert die D-Grid-Initiative[8] (D-Grid) Grid-orientierte Projekte mit dem Ziel eine nachhaltige Grid-Infrastruktur zu schaffen. Über die grundlegenden Basisdienste zur Einrichtung des Grids hinaus, sollen hierbei auch höhere Dienste etabliert werden. Die beteiligten akademischen Communitys setzen sich dabei aus unterschiedlichen Disziplinen, wie z.B. Linguistik (TextGrid[15]) und Astrophysik (AstroGrid[10]) zusammen, darüberhinaus sind jedoch auch kommerzielle Projekte (z.B. FinGrid[12]) eingebunden. Durch die Vielfalt der beteiligten Communitys, liegt eine große Datenheterogenität vor. Methoden der Wissensextraktion können genutzt werden, um diese Heterogenität für den Nutzer zu überwinden und eine höherwertige Suche auf dem vorhanden Wissen anzubieten. Allerdings bestehen umgekehrt natürlich auch besondere Anforderungen an die verwendeten Extraktionsverfahren, z.B. kann eine Community darauf bestehen, dass extrahiertes Wissen in ihrem direkten Einflussbereich verbleibt.

Verfahren zur Wissensextraktion versuchen aus konkreten Informationen in Dateien abstrakte Fakten zu generieren[17]. Beispielsweise kann aus der Aussage “Einstein kam in Ulm zur Welt” das abstrakte Wissen, dass Albert Einstein in Ulm geboren wurde (als Fakt ausgedrückt *gebore-nIn*(AlbertEinstein,Ulm)) hergeleitet werden. Dazu muss allerdings der mehrdeutige Ausdruck “Einstein” als eine Ausdrucksform der eindeutigen Entität *AlbertEinstein* erkannt werden. Es existiert ein breites Spektrum an Verfahren zur Wissensextraktion, drei der bekanntesten sind Snowball[1], KnowItAll[11] and DIPRE[6]. Sie sind anhand einiger Beispielfakten in der Lage mit statistichen Methoden automatisch textuelle Muster zu finden, die wahrscheinlich die zugehörige Relation ausdrücken. Dieser Ansatz kann weiter verbessert werden, indem bei der Wahl der Muster auch Gegenbeispiele berücksichtigt werden (siehe LEILA[21]). Während diese Verfahren je darauf abzielen für eine gegebene Relation weitere Fakten zu finden, versucht z.B. TextRunner[3] Fakten *aller* Relationen, die im Internet auftauchen, zu extrahieren. Allerdings wird hierbei keine Auflösung von Mehrdeutigkeiten und Umwandlung auf Relationen durchgeführt. Liegen die Daten teilweise strukturiert vor, kann das durch angepasste Extraktionsverfahren ausgenutzt werden. So konzentrieren sich einige Ansätze z.B. auf eine Extraktion aus Wikipedia ([22],[2]). In [18] wurde ein Framework zur deklarativen Extraktion basierend auf Datalog eingeführt, welches zur Erzeugung von Portalen wie DBlife[7] genutzt wurde. Das Modell erlaubt eine regelbasierte Extraktion mit Abgleich gegen existierendes Wissen, z.B. zur Bestimmung von Entitäten. Ein anderer Ansatz ([16]) nutzt Markov Logik um Entitäten auf Basis von Wahrscheinlichkeiten zu identifizieren. Dem Ansatz des *Lifelong Learning* folgend verwaltet ALICE[4] eine Wissensbasis in Form einer Ontologie, die durch extrahierte Fakten regelmäßig erweitert wird. Allerdings gibt es dabei keine Mechanismen, die die Konsistenz der Ontologie sicherstellen. SOFIE[23] vereint Muster-basierte Wissensextraktion mit einer logischen Konsistenzprüfung und eignet sich daher insbesondere zur fortwährenden Erweiterung einer Ontologie. Das angestrebte Extraktionsverfahren für die Anwendung im Grid basiert daher auf der SOFIE Architektur.

3. EXTRAKTIONSVERFAHREN

Auf dem abstrakten Level sind *Muster* generische Verallgemeinerungen bestimmter Ausdrucksformen von Wissen. In Texten sind Muster also beispielsweise parametrisierte Formulierungen die jeweils einen bestimmten Sachverhalt auf unterschiedlichen Objekten ausdrücken. Die Formulierung “Einstein kam in Ulm zur Welt” beispielsweise beschreibt den Sachverhalt, dass Albert Einstein in Ulm geboren wurde. Die beiden Komponenten ‘Einstein’ und ‘Ulm’ können dabei einfach ausgetauscht werden, um denselben Sachverhalt zwischen einer anderen Person und einem anderen Ort auszudrücken. Ersetzen wir diese Komponenten also durch Variablen, erhalten wir das allgemeine Muster $\boxed{X \text{ kam in } Y \text{ zur Welt}}$, welches eine Ausdrucksform der *gebore*n*In* Relation darstellt. Tritt dieses Muster nun mit einer konkreten Variablenbesetzung auf, z.B. in der Form $\boxed{\text{Einstein kam in Ulm zur Welt}}$, bezeichnen wir dies als *Musterinstanz*. Die gleiche Musterinstanz kann in verschiedenen Dokumenten auftreten. Solche Vorkommen mit Bezug zu einer Quelle bezeichnen wir als *Mustervorkommen*.

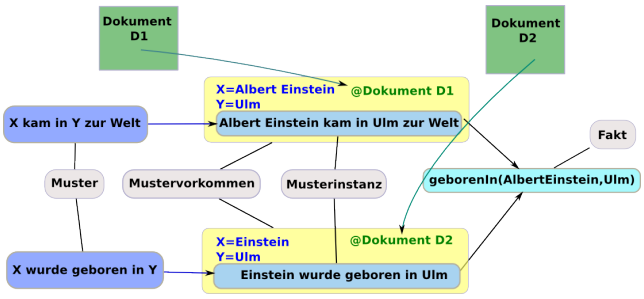

Figure 1: Beispiel der Musterbasierten Extraktion

Dieser Zusammenhang ist in Abbildung 1 skizziert. Die Musterinstanz $\boxed{\text{Albert Einstein kam in Ulm zur Welt}}$ wurde hierbei in Dokument D1 gefunden, während in Dokument D2 eine Instanz des Musters $\boxed{X \text{ wurde geboren in } Y}$ vorliegt. Aus beiden Mustervorkommen lässt sich der Fakt *geborenIn*(AlbertEinstein,Ulm) ableiten.

Generell lässt sich das Extraktionsverfahren in vier Schritte unterteilen. Zuerst müssen Quelldateien auf Mustervorkommen untersucht werden. Anschließend werden diese Vorkommen ausgewertet und potentielle Fakten aus den Mustervorkommen generiert. Im dritten Schritt werden diese potentiellen Fakten mit der bestehenden Wissensbasis abgegli-

chen. Daraus ergibt sich eine optimale Menge an potentiellen Fakten, die mit dem bestehenden Wissen in Einklang steht. Diese Fakten werden dann im letzten Schritt zur Wissensbasis hinzugefügt. Parallel zur Extraktion von Fakten, werden auch eigenständig neue Muster für bereits bekannte Relationen erlernt. Ist beispielsweise bereits bekannt, dass Albert Einstein in Ulm geboren wurde, und in einem Dokument beginnt ein Satz mit "Albert Einstein erblickte das Licht der Welt in Ulm ..." kann daraus abgeleitet werden, dass das Muster $\boxed{\underline{X} \text{ erblickte das Licht der Welt in } \underline{Y}}$ ein guter Kandidat für eine Ausdrucksform des Fakts *geborenIn*(X,Y) ist. Beide Lernverfahren werden dabei einer gemeinsamen logischen Evaluierung unterzogen. Basierend auf logischen Regeln und einer Menge von Fakten und Hypothesen, z.B. über neue Muster, wird eine Zuweisung von Wahrheitswerten zu den Hypothesen gesucht, so dass ein Maximum der logischen Regeln erfüllt ist.

Aussagen. Eine *Aussage* besteht aus einer Relation und einer Liste von Entitäten. Jeder Aussage wird ein binärer *Wahrheitswert* zugeordnet, der in eckigen Klammern angegeben wird:

$geborenIn(AlbertEinstein, Ulm)$ [1]

Eine Aussage mit Wahrheitswert 1 bezeichnen wir als *Fakt*, während eine Aussage mit unbekanntem Wahrheitswert eine *Hypothese* darstellt. Ist kein Wahrheitswert angegeben, handelt es sich um einen Fakt.

Regeln. Die logischen Regeln basieren auf *Literalen*. Ein Literal ist eine Aussage ohne Wahrheitswert, bei der sowohl die Relation als auch die Entitäten durch Variablen ersetzt werden können. Eine Regel ist eine logische Formel erster Stufe basierend auf Literalen. Zur *Instantiierung* einer solchen quantifizierten Regel, wird jede Variable an all ihren Vorkommen durch die gleiche Entität ersetzt. Grundsätzlich müssen Regeln auf allen möglichen Instantiierungen gelten, um gültig zu sein. Üblicherweise wird eine Person beispielsweise nur an einem Ort geboren. Dies lässt sich vereinfacht wie folgt als logische Regel ausdrücken:

$geborenIn(X,Y) \Rightarrow \neg geborenIn(X,Z)$

Basisregeln. Im folgenden wollen wir zwei stark vereinfachte Basisregeln vorstellen, die dem Erlernen neuer Fakten und möglicher neuer Muster entsprechen. Hierzu bedienen wir uns zweier spezieller Relationen:

1. $patternOcc(P,A,B)$ besagt, dass das Muster P mit den Instanzen A und B (z.B. P= $\boxed{\underline{X} \text{ wurde geboren in } \underline{Y}}$ mit X=A="Einstein" und X=B="Ulm") vorgekommen ist. Korrekterweise müssten wir hier noch auf das Dokument, in dem die Musterinstanz gefunden wurde, eingehen. Darauf verzichten wir an dieser Stelle.
2. $expresses(P,R)$ besagt, dass das Muster P eine Ausdrucksform der Relation R (also z.B. *geborenIn*) ist.

Das Erlernen neuer Muster bzw. ihrer Bedeutung kann dann vereinfacht wie folgt als logische Regel ausgedrückt werden:

R1: $\begin{pmatrix} & patternOcc(\text{P,X,Y}) \\ \wedge & R(\text{X,Y}) \end{pmatrix} \Rightarrow expresses(\text{P,R})$

Umgekehrt lässt sich auch das Erlernen neuer Fakten in eine Regel fassen:

R2: $\begin{pmatrix} & patternOcc(\text{P,X,Y}) \\ \wedge & expresses(\text{P,R}) \end{pmatrix} \Rightarrow R(\text{X,Y})$

Bei beiden Regeln ignorieren wir hier, dass je die Entitäten, die X und Y entsprechen, noch eindeutig aus den konkreten Worten identifiziert werden müssten. Dies kann ebenfalls als Teil der Regel formuliert werden. Ebenso können weitere Regeln domainspezifisches Wissen kodieren, z.B. dass Personen nur an einem Ort geboren werden können.

Werden Quelldaten bearbeitet, erzeugt jedes gefundene Mustervorkommen einen entsprechenden *patternOcc* Fakt. Zudem können wir davon ausgehen, dass bereits einige Fakten gesuchter Relationen in der Ontologie vorhanden sind, z.B. weil sie bereits extrahiert wurden oder weil es sich um die anfangs gegebenen Beispiele handelt. Weiterhin könnten auch bereits Instanzen der *expresses* Relation bekannt sein. Wann immer nun die Voraussetzungen einer Regel erfüllt sind, erzeugt diese Regel eine Hypothese aus der als Folge angegebenen Aussage.
Ist z.B. $patternOcc(\boxed{\underline{X} \text{ kam in } \underline{Y} \text{ zur Welt}}, \text{"Einstein"}, \text{"Ulm"})$ und $geborenIn(AlbertEinstein, Ulm)$ gegeben, wird die Hypothese $expresses(\boxed{\underline{X} \text{ kam in } \underline{Y} \text{ zur Welt}}, geborenIn)$ [?] durch Regel R1 erzeugt.
Ist die Bearbeitung der Quelldaten abgeschlossen, muss entschieden werden, welche Hypothesen als wahre Fakten betrachtet und somit in die Ontologie aufgenommen werden sollen. Hierzu wird eine Wahrheitszuweisung an die Hypothesen ermittelt, die ein Maximum an Regelinstanzen erfüllt. D.h. einzelne Regeln dürfen grundsätzlich gebrochen werden. Wenn z.B. bereits in der Ontologie bekannt ist, dass eine generierte Hypothese nicht wahr sein kann, weil das Gegenteil bereits wahr ist, muss die entsprechende Regel gebrochen werden können, ohne dass deshalb der komplette Vorgang fehlschlägt.
Um eine solche Wahrheitszuweisung zu finden, kann die Problemstellung auf ein *maximum satisfiability* (MAX-SAT) Problem ([5]) reduziert werden. Das MAX-SAT Problem basiert auf Klauseln:

DEFINITION 3.1 (KLAUSEL). *Eine Klausel ist eine Disjunktion $X_1 \vee ... \vee X_n$ wobei für alle $X_{i \in \{1,...,n\}}$ gilt: X_i ist entweder eine Variable oder eine negierte Variable.*

DEFINITION 3.2 (MAX-SAT PROBLEM). *Gegeben eine Menge von Klauseln $C_1, ..., C_n$ über m Variablen. Finde eine Wahrheitszuweisung für die Variablen, so dass die Anzahl erfüllter Klauseln maximal ist.*

Das MAX-SAT Problem ist ein bekanntes NP-vollständiges Problem, für das exakte Algorithmen sowie Annäherungsverfahren zur Lösung existieren ([5]). Zur Übertragung in ein MAX-SAT Problem müssen die Regeln per Rewriting in Klauseln umgewandelt werden und die enthaltenen Aussagen bzw. ihre Wahrheitswerte bilden die Variablen.

Gewichtete Regeln. In dem vorgestellten Ansatz werden alle Regeln gleichbehandelt. Allerdings können einige Regeln

wichtiger als andere sein. Dies spielt z.B. eine Rolle, wenn unterschiedlich wahrscheinliche Zuordnungen von Wörtern zu einzelnen Entitäten über Regeln abgebildet werden. Zudem kann so existentes Wissen stärker gewichtet werden als neu erlerntes. Das Problem der Zuordnung von Wahrheitswerten zu den Hypothesen unter Beachtung einer Regelgewichtung kann nicht in ein MAX-SAT Problem überführt werden. Stattdessen kann es aber in ein *gewichtetes* MAX-SAT Problem ([5]) überführt werden. Die Übertragung in ein gewichtetes MAX-SAT Problem, sowie ein effizienter Algorithmus zur Lösung eines solchen Problems wird in [23] diskutiert. Auf beides gehen wir hier nicht weiter ein. Im Grunde kann die Hypothesenbewertung mit einem beliebigen Ansatz gelöst werden, es muss lediglich eine Zuweisung von Wahrheitswerten für die aufgestellten Hypothesen gefunden werden, die möglichst viele Regeln erfüllt und dabei eine Gewichtung erlaubt. Markov Logik böte z.B. einen anderen Ansatz.

4. UMSETZUNG IM GRIDUMFELD

Aus dem Einsatz im Grid ergeben sich besondere Anforderungen an die Wissensexktraktion. Auf einige wollen wir im folgenden kurz eingehen.

Heterogene Daten und Domänen. Durch die Zielsetzung völlig unterschiedliche Communitys über eine gemeinsame Gridarchitektur zu verknüpfen und darüber gegenseitigen Zugriff auf in vorhandenen Daten enthaltenes Wissen zu ermöglichen, ergibt sich eine heterogene Menge an Daten aus denen abstraktes Wissen extrahiert werden soll. Das bedeutet einerseits dasss unterschiedliche Formatierungen strukturierter Inhalte, wie z.B. Tabellen, vorliegen, aber auch dass ganz unterschiedliches Hintergrundwissen vonnöten ist, um sinnvoll abstrakte Fakten aus den Daten verschiedener Communitys zu generieren. Hier bietet das besprochene Extraktionsframework gute Anpassungsmöglichkeiten. Einerseits können Muster generisch anhand von Beispielwissen automatisch erlernt werden, so dass sich die Extraktion inkrementell gestalten lässt. Andererseits können Nutzer aber auch mit entsprechenden Tools direkt Muster vorgeben oder bei entsprechend modularer Implementierung einzelne Komponenten des Frameworks für ihre Belange austauschen. Dadurch kann z.B. ein alternatives Verfahren zur Verarbeitung domainspezifischer strukturierter Daten hinzugefügt werden, dessen Muster sich auf die spezielle Struktur beziehen. Ebenso können auf diese Weise gänzlich andere Inhalte, wie z.B. Audiodaten oder grafische Darstellungen, untersucht werden. Unterschiedliche Mustertypen für verschiedene Datenanalysemodule lassen sich parallel verwalten, es muss lediglich die Information mitgeführt werden, welches Muster zu welchem Analysemodul gehört. Ebenso kann domainspezifisches Vorwissen in Form von logischen Regeln übergeben werden. Alternativ kann Vorwissen in Form von bestehenden Ontologien zur Verfügung gestellt werden, die dann in den Reasoningprozess eingebunden werden. Die Datenheterogenität im Grid beschränkt sich zudem nicht nur auf die Dateninhalte und Formate, sondern auch auf die Sprache der Texte. D.h. das Extraktionsverfahren muss an verschiedene Sprachen, zum Einsatz im D-Grid insbesondere an Deutsch, angepasst werden bzw. einfach an weitere Sprachen anpassbar sein. Einen breiten Basisansatz kann hier eine einfache leicht anpassbare Textanalyse mit weitgehend sprachunabhängigen Methoden bilden, die bei Sprachen für die noch kein höherwertiges Analysemodul vorliegt, zum Einsatz kommt.

Verteilte Daten - Aufspaltung des Frameworks. Eine Grundeigenschaft des Grids ist die verteilte Datenlagerung. Extrahiertes Wissen dagegen sollte zentral in einer Ontologie vorgehalten werden, um eine effiziente Suche zu erlauben und die Konsistenz bei (manuellen) Änderungen sicherstellen zu können. Dabei können einzelne Communitys eigene Ontologien unabhängig voneinander verwalten wollen, um die Kontrolle über 'ihr Wissen' zu behalten. Es wäre nun sehr ineffizient alle Daten zur Extraktion zu der jeweiligen zentralen Stelle, an der die entsprechende Ontologie vorgehalten wird, zu übertragen. Einerseits sollte also die Analyse der vorliegenden Daten möglichst lokal am Lagerort der Daten geschehen. Andererseits muss die Extraktionskomponente das extrahierte Wissen in die Ontologie übertragen und dabei sicherstellen, dass die Konsistenz gewahrt bleibt. Hier bietet sich eine Zweiteilung des Extraktionsverfahrens in einen *Extraktionsclient* und einen *Extrakionsmaster* an. Ersterer übernimmt das lokale Analysieren von Daten und extrahiert Mustervorkommen, letzterer aggregiert die gefundenen Mustervorkommen, generiert daraus Hypothesen und fügt letztlich neue Erkenntnisse zentral in die Ontologie ein. So müssen nicht die eigentlichen Daten, sondern lediglich erkannte Mustervorkommen übertragen werden. Dabei kann die Extraktion auch über das Grid verteilt erfolgen. So können z.B. zur Textanalyse auch rechenintensivere komplexe Sprachanalysemethoden eingesetzt werden.

Konfidenzen. Aufgrund der generischen Gestaltung sowie fehlerhafter Quelldokumente, kann es vorkommen, dass Fakten extrahiert werden, die nicht als wahr anzusehen sind. Verschiedene Muster bzw. Mustertypen können als unterschiedlich sicher betrachtet werden. Beispielsweise ist ein von einer Community manuell vorgegebenes Muster für ihre strukturierten Daten, z.B. ein Mapping von Spalten- und Zeilenattribut in einer Tabelle auf eine Relation, im Allgemeinen als sicherer anzusehen als ein selbsterlerntes textuelles Muster. Da die Nutzer im D-Grid aus unterschiedlichen Fachrichtungen kommen und daher bei Communityübergreifenden Nachforschungen mit Fakten aus ihnen fremden Fachbereichen umgehen müssen, sollte der Grad der Sicherheit einzelner Fakten ersichtlich sein. Hierzu wird ein Konfidenzwert für jeden Fakt berechnet, der das Vertrauen darin repräsentiert, dass der Fakt korrekt extrahiert wurde. Dieser Wert basiert auf dem Vertrauen in die Qualität der Quelle und in die Genauigkeit des benutzten Musters. D.h. es existieren Konfidenzwerte für Muster, Quellen und Fakten. Ihre Abhängigkeiten werden in Abbildung 2 skizziert. Generell basiert der Konfidenzwert eines Fakts auf allen Mustervorkommen, die ihn bestätigen, bzw. dem durch sie generierten Vertrauen. Ein Mustervorkommen eines exakten und sehr wohl erprobten Musters in einer vertrauenswürdigen Quelle lässt den Fakt glaubwürdig erscheinen. Ist der einzige Beleg für den Fakt jedoch ein als unsicher eingestuftes Muster, das auf einer unglaubwürdigen Quelle gefunden wurde, so wird der Fakt wenig glaubwürdig erscheinen. Findet sich beispielsweise der Satz "Albert Einstein erhielt seinen Nobelpreis für ..." auf der Wikipediaseite über Albert Einstein, so ist dies ein sehr eindeutiger Beleg für den

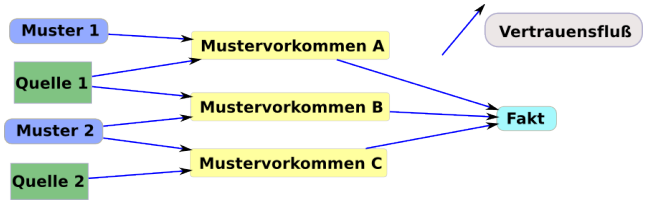

Figure 2: Beispiel: Vertrauensfluß der zum Gesamtvertrauen in einen Fakt führt

Fakt *hatAuszeichnungErhalten*(AlbertEinstein,Nobelpreis). Eng damit verknüpft ist das Einarbeiten von Nutzerfeedback. Wird ein extrahierter Fakt von einem Nutzer als fehlerhaft erkannt, sollte dieses Wissen genutzt werden, um für die Zukunft hinzuzulernen. Das kann beispielsweise über eine Anpassung der Konfidenz in das entsprechende Muster geschehen. Ein weiterer Schritt wäre die Konfidenzwerte auch direkt in das Extraktionsverfahren einzubinden, z.B. zur Regelgewichtung. Hinzu kommt, dass die gegenwärtige Modellierung der Extraktion eingeschränkt ist. So ist es beispielsweise nicht ohne weiteres möglich zeitliche oder örtliche Einschränkungen zu erkennen bzw. mit einem Fakt zu verknüpfen. Ebenso können z.B. kausale Zusammenhänge nicht erkannt werden. Die Modellierung von Meinungen und Wertungen, die sich ja unter Umständen widersprechen können, wird ebenso noch nicht berücksichtigt.

5. ZUSAMMENFASSUNG & AUSBLICK

In der vorliegenden Arbeit wurde ein Überblick über aktuelle Verfahren der Wissensextraktion gegeben und ein generisches Framework zur Wissensextraktion vorgestellt. Darauf aufbauend wurden einige Aspekte diskutiert, die es bei einer Übertragung des Extraktionsverfahrens in ein Gridumfeld zu beachten gilt. Für einige dieser Problemstellungen wurden erste Lösungsansätze entlang des vorgestellten Frameworks aufgezeigt. Im weiteren Verlauf der Forschungsarbeit müssen die noch offenen Problemstellungen weitergehend analysiert und vollständige Lösungen in Form eines lauffähigen Systems umgesetzt werden. Dieses wird dann an die Bedürfnisse zweier Testcommunitys innerhalb von D-Grid angepasst und anschließend in diesem Rahmen evaluiert werden.

6. LITERATUR

[1] E. Agichtein and L. Gravano. Snowball: extracting relations from large plain-text collections. In *DL '00*, pages 85–94, 2000.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In *ISWC*, pages 11–15. Springer, 2007.

[3] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open Information Extraction from the Web. 2007.

[4] M. Banko and O. Etzioni. Strategies for lifelong knowledge extraction from the web. In *K-CAP '07*, pages 95–102, 2007.

[5] B. Borchers and J. Furman. A two-phase exact algorithm for MAX-SAT and weighted MAX-SAT problems. *Journal of Combinatorial Optimization*, 2:299–306, 1997.

[6] S. Brin. Extracting patterns and relations from the world wide web. In *WebDB '98*, pages 172–183, 1999.

[7] P. DeRose, W. Shen, F. Chen, A. Doan, and R. Ramakrishnan. Building structured web community portals: a top-down, compositional, and incremental approach. In *VLDB '07*, pages 399–410, 2007.

[8] D-Grid Initiative. `http://www.d-grid.de/`.

[9] E. Laure et al. Programming the Grid with gLite. In *Computational Methods in Science and Technology*, 2006.

[10] H. Enke, M. Steinmetz, T. Radke, A. Reiser, T. Röblitz, and M. Högqvist. AstroGrid-D: Enhancing Astronomic Science with Grid Technology. 2007.

[11] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Web-scale information extraction in knowitall: (preliminary results). In *WWW '04*, pages 100–110, 2004.

[12] FinGrid - Financial Business Grid. `http://www.fingrid.de/`.

[13] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid - Enabling Scalable Virtual Organizations. *International Journal of Supercomputer Applications*, 15:2001, 2001.

[14] gLite - Lightweight Middleware for Grid Computing. `http://glite.web.cern.ch/glite/`.

[15] M. W. Küster, C. Ludwig, and A. Aschenbrenner. TextGrid: eScholarship und vernetzte Angebote. *it - Information Technology*, 51(4):183–190, 2009.

[16] H. Poon and P. Domingos. Joint inference in information extraction. In *AAAI'07*, pages 913–918. AAAI Press, 2007.

[17] S. Sarawagi. Information extraction. *Foundations and Trends in Databases*, 1(3):261–377, 2008.

[18] W. Shen, A. Doan, J. F. Naughton, and R. Ramakrishnan. Declarative information extraction using datalog with embedded extraction predicates. In *VLDB '07*, pages 1033–1044, 2007.

[19] A. Streit, D. Erwin, D. Mallmann, R. Menday, M. Rambadt, M. Riedel, M. Romberg, B. Schuller, and P. Wieder. UNICORE - From Project Results to Production Grids. In *Grid Computing and New Frontiers of High Performance Processing*, 2005.

[20] F. M. Suchanek. *Automated Construction and Growth of a Large Ontology*. PhD thesis, Saarland University, 2009.

[21] F. M. Suchanek, G. Ifrim, and G. Weikum. Combining linguistic and statistical analysis to extract relations from web documents. In *KDD '06*, pages 712–717. ACM, 2006.

[22] F. M. Suchanek, G. Kasneci, and G. Weikum. YAGO: A Large Ontology from Wikipedia and WordNet. *Web Semant.*, 6(3):203–217, 2008.

[23] F. M. Suchanek, M. Sozio, and G. Weikum. SOFIE: A Self-Organizing Framework for Information Extraction. In *WWW 2009*, 2009.

[24] D. Thain, T. Tannenbaum, and M. Livny. Distributed computing in practice: the Condor experience: Research Articles. *Concurr. Comput. : Pract. Exper.*, 17(2-4):323–356, 2005.