Pathfinding

Path Query Processing on Very Large RDF Graphs & Learning Relations by Pathfinding

TECHNISCHE UNIVERSITÄT DARMSTADT

Gliederung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Einführung
- Path Query Processing on Very Large RDF Graphs
  - Kritik
- Learning Relations by Pathfinding
  - Kritik
- Fazit

Einführung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Semantic Web
  - Information in maschinenlesbarer Form bereitstellen
  - (Semantische) Verweise zwischen Seiten nutzbar machen
  - Schlussfolgern ermöglichen
  - Komplexe (nützliche!) Abfragen ermöglichen

Quelle: [3]

- RDF = Ressource Description Framework
  „System zur Beschreibung von Ressourcen“
  - Pattern: <**s**ubject, **p**redicate, **o**bject>
  - Bsp.: Moritz Hof studiert an der TU Darmstadt.

Einführung

TECHNISCHE UNIVERSITÄT DARMSTADT

- RDF-3X
  - experimentelles RDF Speicher- und Anfragesystem

- SPARQL
  - graph-basierte Anfragesprache für RDF
  - Beispiel

```
PREFIX look: <http://example.com/exampleOntology#>
SELECT ?capital ?country
WHERE { ?x abc:cityname ?capital.
        ?y abc:countryname ?country.
        ?x abc:isCapitalOf ?y.
        ?y abc:isInContinent abc:europe. }
```

Gliederung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Einführung

- Path Query Processing on Very Large RDF Graphs
  - Kritik
- Learning Relations by Pathfinding
  - Kritik

- Fazit

Path Query Processing…

Basics

TECHNISCHE UNIVERSITÄT DARMSTADT

…on Very Large RDF Graphs

Aufgabe:	Erreichbarkeit zwischen Knoten & kürzeste Wege in großen RDF Graphen (diskresident) finden
Problem:	Derzeit einzige „reife" Möglichkeit ist mit JENA (eine SPARQL-Erweiterung), die allerdings „sehr" langsam ist
Lösung:	„Optimierung" des Dijkstra Algorithmus, kostenorientierte Anfrageoptimierung
Einschränkung:	Begrenzte Pfade: Subjekt(e) und/oder Objekt(e) müssen angegeben/eingeschränkt werden => keine all-to-all Abfragen

Path Query Processing…

„Test“ Datenbanken

TECHNISCHE UNIVERSITÄT DARMSTADT

UniProt

- “Die Datenbank” über Proteine: Proteinfunktion und -struktur
- viele Informationen werden aus Genom Projekten abgeleitet & angereichert mit Informationen aus wissenschaftl. Literatur
- 57 GB, 845 Millionen einzigartige Tripel

YAGO2

- abgeleitet aus Wikipedia
- lt. Paper: 80 Millionen Tripel enthalten, für die Tests genutzt
  lt. YAGO-Webseite: 460 Millionen Tripel

Path Query Processing…
SPARQL erweitern

TECHNISCHE UNIVERSITÄT DARMSTADT

Neben dem bekannten Pattern <**s**ubject,**p**redicate,**o**bject>
wird das *path triple Pattern* ergänzt: <**s**ubject,**p**ath,**o**bject>

Bsp.:

```
?person ??path ?city.
?city <isLocatedIn> ?country.
?country <isMemberOf> <European_Union>
```

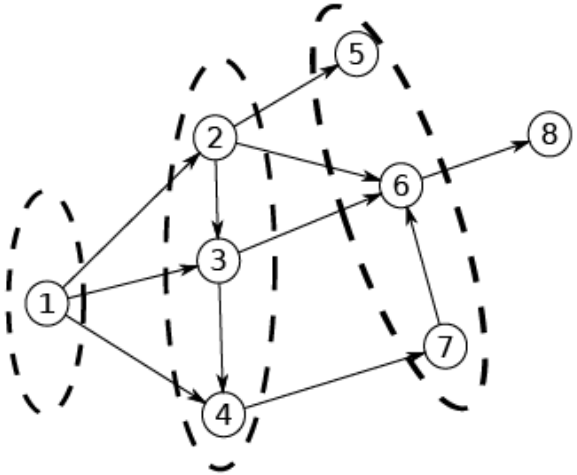
Quelle: [1]

Filtermöglichkeiten wurden ergänzt:

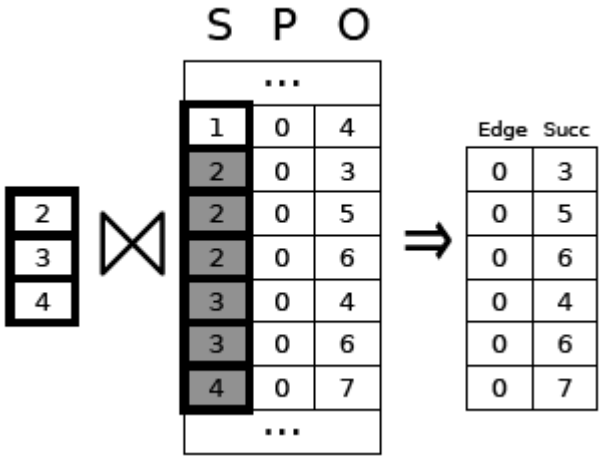
- containsAny(path, element),
- containsOnly(path, element),
- length(path)

Path Query Processing…

Dijkstra optimieren - Idee



Quelle: [1]



Quelle: [1]

Path Query Processing…

Join-based Dijkstra

Algorithm 1: JOIN-BASED DIJKSTRA'S ALGORITHM

```
Input: s, d - start and destination nodes in the graph
Result: path p from s to d
1  begin
2      Q ← {s}
3      V_CUR ← {s}
4      V_PREV ← ∅
5      while Q is not empty do
6          node ← Q.dequeue
7          node.status ← processed
8          if node = d then
9              p ← reconstruct path from s to d
10         if V_PREV is empty then
11             S ← GETNEIGHBORS(V_CUR)
12             V_PREV ← V_CUR
13             clear V_CUR
14         Remove node from V_PREV
15         Relax nodes from S[node]
16         foreach n ∈ S[node], n.status ≠ processed do
17             Add node to V_CUR
```

Quelle: [1]

1. Sammeln von besuchten Knoten in V_{cur} und Aufruf damit von getNeighbors(). Zuvor genutzte Knoten werden in V_{prev} verschoben.

2. Wenn V_{prev} leer ist wird nach neuen Nachfolgern gesucht.

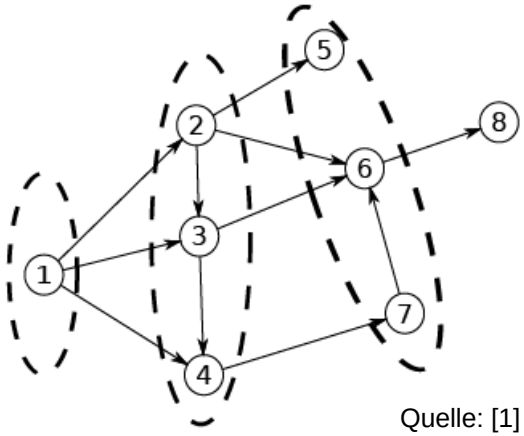
Path Query Processing…

Internal IDs & Dictionary

- RDF-3X ordnet jeder URI und jeder Konstanten eine eindeutige Integer ID zu & arbeitet auf *Tripeln von Integerwerten*
- Die Verzeichnisse sind nach den IDs sortiert. Die Einsortierung erfolgt nach dem „first-come, first serve“ Prinzip

- Problem: IDs könnten weit auseinander liegen im Verzeichnis =>ineffizient

- Lösung: Ändern der IDs mit Hilfe einer „Breitensuche“, startend mit allen Wurzelknoten & auf Basis des temporären Verzeichnisses, vor dem Laden der Strings und Erstellen von Indizes



Quelle: [1]

Path Query Processing…

Cardinality estimation

TECHNISCHE UNIVERSITÄT DARMSTADT

- erlaubt die Aufnahme der Pfadanfrageverarbeitung in das „Kostenmodel“ und die Anfrageoptimierung von RDF-3X
- zusätzlicher B+ Baum mit Laufzeit Schätzungen
  - # Vorgänger und # Nachfolger

- 3 Fälle
  - **S**ubjekt oder **O**bjekt konstant: Anzahl besuchter Knoten berechnen
  - **S**ubjekt und **O**bjekt konstant: | #Nachfolger **s** - #Vorgänger **o** |
  - „alles“ variabel: wegen der Einschränkung nicht für alle Tripel einer Abfrage möglich, es muss also in einem „späteren“ Tripel der Abfrage zumindest eine Konstante geben, die Kardinalität dieses Tripels wird als Schätzung übernommen

Path Query Processing...

Cardinality estimation – estimation error & costs

TECHNISCHE UNIVERSITÄT DARMSTADT

- Fehler am Beispiel YAGO2 (1000 zufällige Tripel mit je einem konstanten Knoten)

$$relative\ error = \frac{\max(real\ cardinality, estimation)}{\min(real\ cardinality, estimation)} - 1$$

Quelle: [1]

  - ∅ Fehler der Schätzung(en) der Kardinalitäten
    - Vorgänger: 16%
    - Nachfolger: 50%

- „Kosten" am Beispiel YAGO2
  - 5 Minuten zum Berechnen des Index (10%)
  - 320Mb Festplattenplatz (12%)

Path Query Processing…

Auswertung I

TECHNISCHE UNIVERSITÄT DARMSTADT

- YAGO2, 1000 zufällige Knoten,
- getNeighbors() mit Nachfolgern der 1000 zufälligen Knoten

	less than 1000 nodes returned		more than 1000 nodes returned	
Dictionary	join	lookup	join	lookup
new dict	4 ms	12 ms	32 ms	201 ms
old dict	18 ms	13 ms	78 ms	228 ms

Quelle: [1]

- Dijkstra Algorithmus

	less than 1000 nodes visited		more than 1000 nodes visited	
Dictionary	join	lookup	join	lookup
new dict	5 ms	8 ms	120 ms	1052 ms
old dict	12 ms	8 ms	458 ms	1169 ms

Quelle: [1]

Path Query Processing...

Auswertung II

TECHNISCHE UNIVERSITÄT DARMSTADT

Q1: select ?loc ??path where {<Ulm> ??path ?loc. pathfilter(containsOnly(??path, <isLocatedIn>))}

Q5: select ?person where {?person <isKnownFor> ?smth. ?smth ??related <wordnet physical phenomenon>. ?place ??loc ?country. ?country <type> <wikicategory European countries>.?country <type> <wikicategory_Mediterranean>. ?person <wasBornIn>?place.pathfilter(containsOnly(??loc, <isLocatedIn>)) }

Quelle: [1]

	YAGO2					
	Q1	Q2	Q3	Q4	Q5	geom. mean
	cold cache (warm cache)					
RDF-3X	0.18 (0.004)	1.09 (0.19)	1.21 (0.09)	1.12 (0.18)	2.49 (1.39)	0.92 (0.11)
Jena	0.45 (0.1)	34.54 (1.31)	28.17 (1.07)	29.98 (0.95)	>30min (>30min)	>29.83 (>2.99)

Quelle: [1]

Path Query Processing...
Auswertung II

TECHNISCHE UNIVERSITÄT DARMSTADT

Direction	Q1	Q2	Q3	Q4	Q5	geom mean
			cold cache			
baseline	0.18	1.09	1.21	1.12	2.49	0.92
only new dict	0.69	4.85	5.20	8.63	11.03	4.40
only join-based Dijkstra	0.18	7.71	3.82	3.86	8.64	2.81
			warm cache			
baseline	0.004	0.19	0.09	0.18	1.39	0.11
only new dict	0.04	0.99	0.75	0.54	5.87	0.62
only join-based Dijkstra	0.004	0.41	0.29	0.36	4.86	0.24

Quelle: [1]

	YAGO2					
	Q1	Q2	Q3	Q4	Q5	geom. mean
	cold cache (warm cache)					
RDF-3X	0.18 (0.004)	1.09 (0.19)	1.21 (0.09)	1.12 (0.18)	2.49 (1.39)	0.92 (0.11)
Jena	0.45 (0.1)	34.54 (1.31)	28.17 (1.07)	29.98 (0.95)	>30min (>30min)	>29.83 (>2.99)

Quelle: [1]

Path Query Processing...

Auswertung III

Q1:

select ?a ?mod ??inf where ?a <mnemonic> ?vo. ?a <replacedBy> <P62965>. ?a <type> <Protein>. ?a <modified> ?mod. ?b <modified> "2005-08-30". ?b <replacedBy> <P62964>. ?b <type> <Protein>. ?a <replacedBy> ?ab. ?ab ??inf ?b .

Quelle: [1]

	UniProt			
	Q1	Q2	Q3	geom. mean
RDF-3X	0.52 (0.01)	4.01 (3.21)	11.54 (4.61)	2.88 (0.53)
Jena	>30min (>30min)			

Quelle: [1]

Path Query Processing…
Kritik

TECHNISCHE UNIVERSITÄT DARMSTADT

- Join-based Dijkstra so nicht funktionsfähig
- getNeighbors() nicht vorgestellt
- „reconstruct path“ nicht vorgestellt
- Bei der Schätzung der Kardinalitäten scheint noch Potenzial zu sein
- Vergleichswerte ohne „Cardinality estimation“ fehlen
- Anfragen/Queries an UniProt nicht beschrieben
- Darauf, wie die Laufzeitschätzungen und Abfrageoptimierung in RDF-3X integriert ist, wird nicht eingegangen („cheapest plan“)

Gliederung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Einführung
- Path Query Processing on Very Large RDF Graphs
  - Kritik
- Learning Relations by Pathfinding
  - Kritik
- Fazit

Learning Relations by Pathfinding

First order learning

TECHNISCHE UNIVERSITÄT DARMSTADT

- First-order learning systems
- Local Hill/Plateau Problem

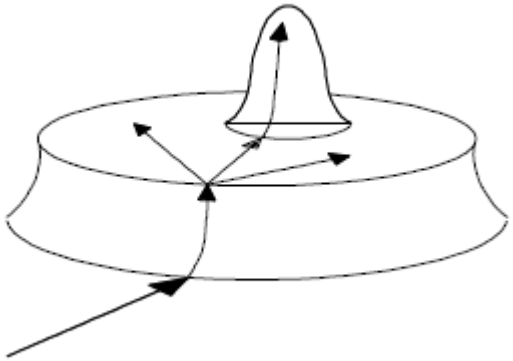
- Top-down
  - FOIL
  - FOCL
  - Forte
- Bottom-up
  - GOLEM, wird nicht betrachtet

- Alle lernen bis zur Abdeckung aller positiver Beispiele. Dabei wird pro Schritt ein Literal ergänzt, was sozusagen die Ursache des Problems ist.

Learning Relations by Pathfinding

Basics

- Problem: lokale Maxima/Plateaus
- Lösung: relational pathfinding



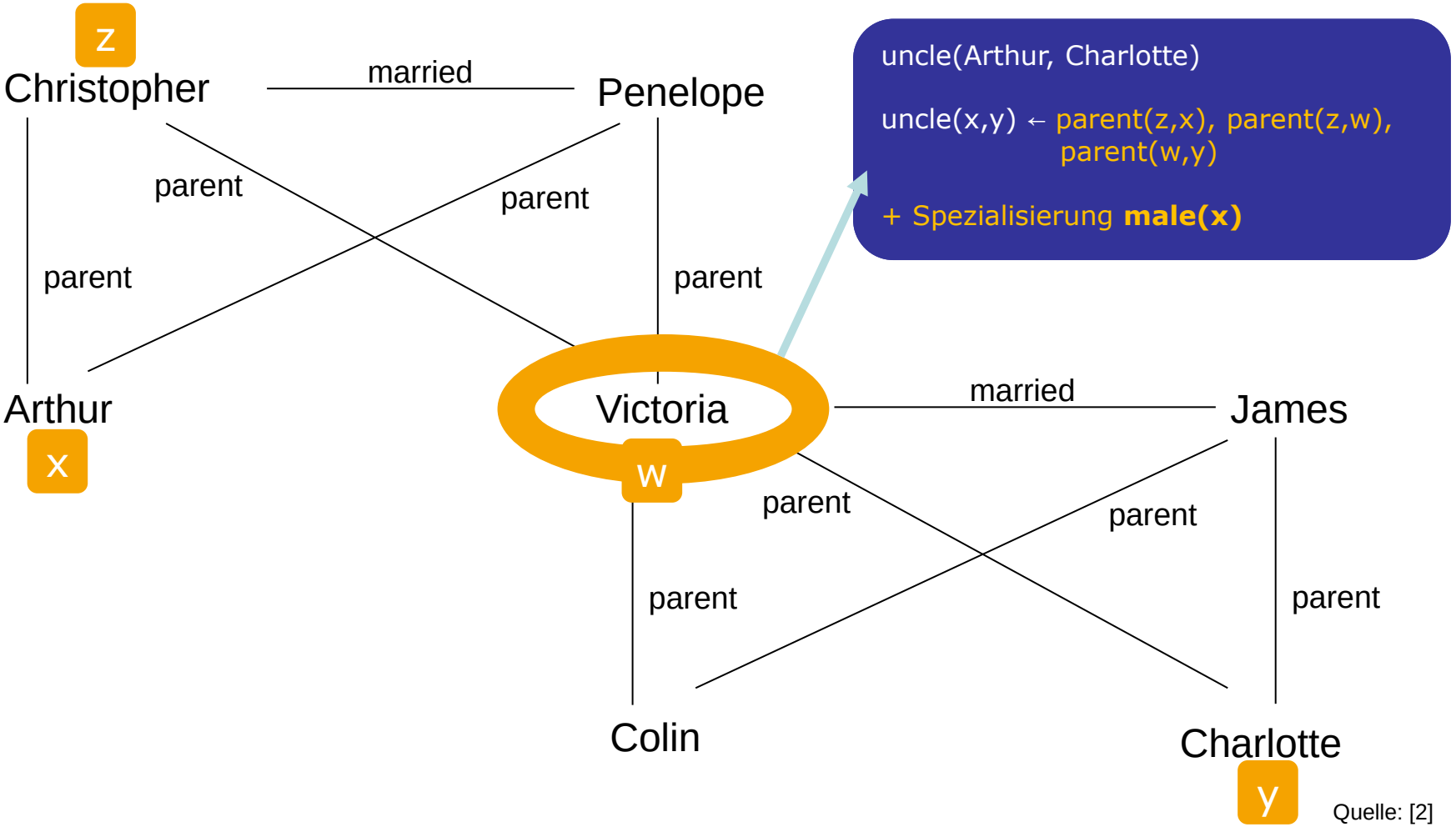
Quelle: [2]

- Es werden nur 3 Domänen untersucht
  - Familienbeziehungen
  - Qualitative Modelbildung
  - Logische Programmierung

- Basisannahme
  - In relationalen Domänen existiert für die Menge der Terme, die das Zielkonzept erfüllen, normalerweise ein Pfad mit fixer Länge
- Idee/Ziel: Finden dieser Pfade

Learning Relations by Pathfinding

Beispiel

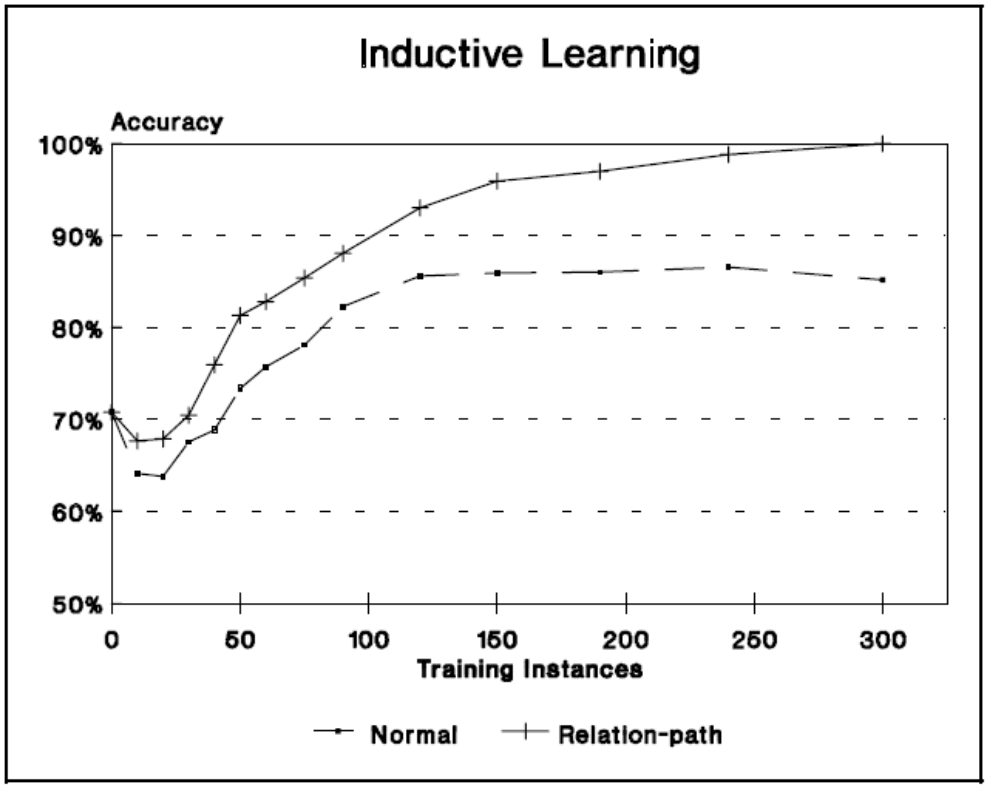


Quelle: [2]

Learning Relations by Pathfinding

Auswertung

- Inductive learning performance in Hinton's family domain
- Deutlich „besserer" Lernerfolg bei beliebiger Anzahl von Trainingsbeispielen



Quelle: [2]

Learning Relations by Pathfinding

Kritik

- Keine Angaben/Vergleiche zum Thema Komplexität oder Laufzeit(en)
- Domain Hintons Family nicht ganz klar, bzw. die Auswertung
- Nur wenig relevantes zum Thema „Semantic Web“

Gliederung

TECHNISCHE UNIVERSITÄT DARMSTADT

- Einführung
- Path Query Processing on Very Large RDF Graphs
  - Kritik
- Learning Relations by Pathfinding
  - Kritik
- Fazit

Fazit

TECHNISCHE UNIVERSITÄT DARMSTADT

- In beiden Papern keine Angaben zur Komplexität
- Auswertung im ersten Paper meines Erachtens gut
- Schnittpunkte
  - Learning relations by pathfinding kann evtl. zur besseren Kardinalitätenschätzung oder grundsätzlich schnelleres finden von Pfaden im Path Query Processing genutzt werden

Vielen Dank für Ihre Aufmerksamkeit.

Quellen

[1] B. L. Richards and R. J. Mooney, "Learning Relations by Pathfinding," in *AAAI*, 1992, pp. 50–55.

[2] A. Gubichev and T. Neumann, "Path Query Processing on Very Large RDF Graphs," *Work,* http://131.159.16.103/research/publications/, 2008.

[3] Vorlesungsfolien WS 2011 von Heiko Paulheim, Semantic Web: http://www.ke.tu-darmstadt.de/lehre/ws-11-12/semantic-web/folien-teil-1

[4] Tagungsband *AAAI*, 1992.