Institut für Parallele und Verteilte Systeme
Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diplomarbeit Nr. 3133

Entwicklung eines Kostenmodells für den Optimierer einer nativen ortsbasierten RDF-Datenbank

Björn Dick

Studiengang:	Informatik
Prüfer:	Prof. Dr.-Ing. habil. Bernhard Mitschang
Betreuer:	Dipl.-Inf. Andreas Brodt
begonnen am:	17. Januar 2011
beendet am:	19. Juli 2011
CR-Klassifikation:	H.2.4, H.2.8

Kurzfassung

Technologien des Semantic Web, wie insbesondere RDF, eignen sich hervorragend zur Darstellung und Verarbeitung semistrukturierter Daten. Dies gilt ebenso, falls die Daten Ortsinformationen beinhalten. Systeme zur Verarbeitung derartiger, ortsbasierter Daten auf Basis von RDF stellen somit erstrebenswerte Werkzeuge dar. Im Rahmen der Entwicklung eines solchen Systems, sieht man sich jedoch mit der Aufgabe konfrontiert, geeignete Ausführungspläne zu optimieren. Diese Arbeit befasst sich daher mit der Bereitstellung adäquater Entscheidungshilfen für den Optimierer eines nativen, ortsbasierten RDF-Datenbankverwaltungssystems. Aufgrund möglicher Korrelationen zwischen ortsbasierten und gewöhnlichen Attributen von Entitäten ist hierbei insbesondere ein Mechanismus zur Kardinalitätsabschätzung entsprechender Anfragen notwendig, denn die Werte einer solchen Abschätzung bestimmen maßgeblich die Anordnung der Joins in einem geeigneten Ausführungsplan, der wiederum ausschlaggebend für die Performanz der Anfragebearbeitung ist.
Da bislang keine geeigneten Ansätze zur Lösung dieser Aufgabe zu finden sind, werden im Rahmen dieser Arbeit hierfür eigene Konzepte entwickelt. Das erste dieser Konzepte unterteilt den betrachteten, geographischen Bereich in disjunkte Kacheln, um damit eine Kardinalitätsabschätzung für die in einem Anfragefenster enthaltenen Objekte zu ermöglichen. Auf dieser Grundlage wird anschließend ein Ansatz entwickelt, der die Kardinalität einer Menge von Teilgraphen - und damit potentiellen Anfrageergebnissen - anhand einer Kombination ihrer enthaltenen Pfade abschätzt. Auf Basis dieser Abschätzungen ist es in der Folge möglich, ein Kriterium zu konzipieren, mit dessen Hilfe geeignete Operatoren zur Realisierung eines ortsbasierten Filters ausgewählt werden können.
Die Evaluation einer Implementierung dieser Konzepte zeigt, dass anhand der Unterteilung in Kacheln lediglich moderate Verbesserungen der Abschätzungsgenauigkeit sowie der daraus resultierenden Ausführungszeiten möglich sind. Durch Hinzunahme der Kardinalitätsabschätzung mittels Pfaden können jedoch deutlich bessere Resultate erzielt werden. Auf deren Basis ist es dem erwähnten Entscheidungskriterium möglich, in gewinnbringender Art und Weise die jeweils optimalen Operatoren zur Realisierung eines ortsbasierten Filters auszuwählen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

Aufgrund der Entwicklung hin zur Informationsgesellschaft erlangen Informationssysteme und insb. das Internet eine zunehmende Bedeutung in quasi allen Lebensbereichen. Hieraus resultiert eine unüberschaubare Menge maschinell gespeicherter Daten, die zudem ständig zunimmt. Im Sinne der Anstrengungen hin zu einem „Semantic Web“ [BLF99] gilt es, diese Daten in geeigneter Weise miteinander zu verknüpfen. Die besondere Herausforderung liegt hierbei in der Tatsache, dass die Daten aus unterschiedlichen Quellen stammen und somit, wenn überhaupt, in aller Regel kein einheitliches Schema aufweisen. Es ist also ein Konzept notwendig, das es erlaubt, semistrukturierte Daten in geeigneter Weise darzustellen. Als mögliche Technologie hierfür wurde das *Resource Description Framework*, kurz RDF [KC04], vorgeschlagen und erfreut sich mittlerweile einer umfangreichen Akzeptanz.
Durch die schnell wachsende Verbreitung GPS-fähiger mobiler Geräte können darüber hinaus zunehmend auch Daten mit Ortsbezug erhoben werden. Hierbei ist ebenfalls kein einheitliches Schema zu erwarten, so dass sich RDF auch zur Darstellung derartiger Daten anbietet.

In RDF wird jegliche Information in Form von Tripeln der Art $(\mathit{Subjekt}, \mathit{Prädikat}, \mathit{Objekt})$ beschrieben. [1] Ein Tripel (s, p, o) drückt hierbei aus, dass Objekt o mit dem Subjekt s in einem Zusammenhang steht, der durch das Prädikat p beschrieben wird. Durch die Beschränkung auf dieses simple und gewissermaßen allen Schemata inhärente Konzept wird es möglich, Daten aus unterschiedlichen Quellen in ein und der selben Weise darzustellen, wobei selbst Metadaten wie die Äquivalenz von Attributen unterschiedlicher Schemata in RDF selbst beschrieben werden können.
Im Rahmen einer Modellierung des Stuttgarter Nahverkehrsnetzes könnte beispielsweise die Tatsache, dass die Haltestelle Feuersee Teil der S-Bahn-Linie S1 ist, wie folgt in RDF dargestellt werden. Mit Hilfe des Prädikats „`a`“ werde dabei der Typ einer Entität spezifiziert:

```
(S1 a Rapid_Transit_Line)
(Feuersee a Station)
(Feuersee partOf S1)
```

[1] Die Begriffe Subjekt, Prädikat und Objekt beschreiben hierbei verschiedene *Rollen* im Kontext von RDF-Tripeln und sollten nicht mit den entsprechenden Begrifflichkeiten aus dem Bereich natürlichsprachlicher Grammatik verwechselt werden!

Eine solche Darstellung kann, wie in Abbildung 1.1 zu sehen ist, auch als Graph interpretiert werden. Die einzelnen Tripel werden dabei durch Kanten repräsentiert, deren Beschriftung sich aus dem jeweiligen Prädikat ergibt. Die Knoten stellen je nach Zusammenhang Subjekte oder Objekte dar. Eine derartige Darstellung der betrachteten Datenbasis wird im weiteren Verlauf dieser Diplomarbeit als *RDF-Graph* bezeichnet.
Um weitere Attribute bzw. Beziehungen auszudrücken können analog weitere Tripel bzw. Kanten hinzugefügt werden.

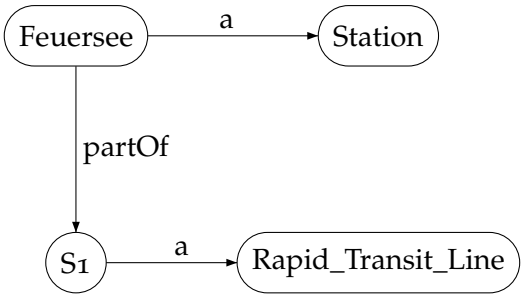


Abbildung 1.1: Graphinterpretation einer Modellierung in RDF

Im Vergleich zum derzeit weit verbreiteten Entity-Relationship-Modell werden also sowohl Attribute („S1 hat den Objekttyp S-Bahn-Linie“) wie auch Beziehungen („Haltestelle Feuersee ist Teil der Linie S1“) in ein und der selben Weise ausgedrückt.

Um aus einer vorhandenen RDF-Datenbank gezielt relevante Informationen extrahieren zu können, wurde die deklarative Anfragesprache SPARQL [Pru07] entwickelt. Die im Kontext dieser Arbeit relevanten SPARQL-Anfragen sind im Wesentlichen von der Form

```
SELECT Variablenliste WHERE {
  Subjekt_1  Prädikat_1  Objekt_1 .
  Subjekt_2  Prädikat_2  Objekt_2 .
  ...
}
```

Die Subjekte, Prädikate und Objekte können hierbei jeweils entweder Konstanten oder Variablen sein. Bei Interpretation der RDF-Daten als Graph (vgl. o.) beschreibt eine solche Anfrage ein *Muster* im RDF-Graph, das durch Kombination beider Tripel entsteht. Alle Teile des Graphen, die diesem Muster entsprechen, können mit Hilfe von Joins über Tripel-Mengen bestimmt werden und stellen das Resultat der Anfrage dar. Hierzu werden die Variablen des Musters gemäß ihrer Entsprechungen in den jeweiligen Teilgraphen gebunden und an den Benutzer zurückgegeben.

Im Kontext des oben skizzierten Anwendungsbeispiels könnten also sämtliche Haltestellen, die von der Haltestelle Universität aus per Direktverbindung erreichbar sind, wie folgt bestimmt werden:

```
SELECT ?x WHERE {
  ?x partOf ?y .
  ?y a Rapid_Transit_Line .
  Universität partOf ?y
}
```

Die Elemente `?x` und `?y` stellen hierbei Variablen dar. Bei den restlichen Elementen handelt es sich um Konstanten.

Wie bereits erwähnt, sollte es über die bislang vorgestellten Modellierungskonzepte hinaus möglich sein, Daten mit Ortsbezug in einem integrierten RDF-System verwalten zu können. Dadurch wird es möglich, Daten in ein und der selben Anfrage sowohl über ortsbezogene, als auch über nicht-ortsbezogene Prädikate auszuwählen. Um derartige Daten *zweckdienlich* in RDF integrieren zu können, werden jedoch Mechanismen benötigt, die es gestatten, die besonderen Eigenschaften von Positionen, Ausdehnungen, etc. auszunutzen (Bestimmung räumlicher Nähe, Schnitt von Objekten, etc.). Eine Verwaltung ortsbezogener Daten in der Faktentabelle erscheint vor diesem Hintergrund wenig ratsam, da diese räumliche Beziehungen in keinster Weise reflektiert. Vielmehr bietet sich hierfür eine Speicherung der zugehörigen Koordinaten in speziellen Indexstrukturen an. Derartige Strukturen werden im Folgenden als Geoindexe bezeichnet und kommen bereits in anderen DBVS gewinnbringend zum Einsatz. Ortsbezogene Daten werden dabei mit Hilfe geographischer oder geometrischer Objekte in einem Koordinatensystem repräsentiert. Der Geoindex bietet hierbei die Möglichkeit, Objekte anhand unterschiedlicher geographischer und geometrischer Eigenschaften innerhalb eines durch die jeweilige Anfrage spezifizierten Bereichs auszuwählen.
Durch diese gesonderte Behandlung ortsbezogener Daten können jedoch die bestehenden Methoden zur Join-Bildung sowie das entsprechende Kostenmodell, nicht ohne Weiteres für ortsbezogene Daten genutzt werden. In [BNM10] wurde gezeigt, wie die vorhandene RDF-Datenbank RDF-3X [NW10] um einen Geoindex zur Integration ortsbezogener Daten erweitert werden kann (die entsprechende Implementierung wird im Folgenden als GeoRDF-3X bezeichnet). Hierbei wurden geeignete Mechanismen entwickelt, um Daten gleichzeitig anhand ortsbezogener wie auch nicht-ortsbezogener Eigenschaften abfragen zu können.
Wie üblich ist auch hier die Ausführungsreihenfolge bei mehreren notwendigen Joins von entscheidender Bedeutung für eine performante Beantwortung der Anfrage. Leider stehen dem Optimierer von GeoRDF-3X bislang keine Informationen zur Verfügung, um Joins über ortsbezogenen Daten so im Operatorbaum platzieren zu können, dass daraus eine Minimierung der globalen Ausführungskosten resultiert. Vielmehr werden Joins mit dem Geoindex stets an unterster Position im Operatorbaum ausgeführt, was in vielen Fällen nicht das Optimum darstellt. Beispielsweise ist im Rahmen der exemplarisch betrachteten Modellierung des Stuttgarter Nahverkehrsnetzes eine Einschränkung der Lage von Haltestellen auf Orte innerhalb des Stadtgebiets wenig selektiv. Vielmehr sollten die qualifizierten Haltestellen sinnvollerweise zunächst anhand anderer Prädikate ausgewählt, d.h. andere Joins ausgeführt werden. Da eine frühzeitige, möglichst restriktive Selektion qualifizierter Tupel jedoch wünschenswert ist, sollte eine Möglichkeit geschaffen werden, auch Joins mit dem Geoindex adaptiv, und möglichst ideal, im Operatorbaum zu platzieren.

Da sich RDF über die reine Wissensrepräsentation hinaus auch zur Anwendung in anderen Fachgebieten (Biologie, Analyse sogenannter „Social Networks“ [BBP+08], etc.) eignet und viele dieser Anwendungen ebenfalls Ortsbezug aufweisen, könnten auch sie von einer Performanz-Steigerung bei ortsbezogenen Anfragen profitieren.

1.2 Aufgabenstellung

Ziel dieser Diplomarbeit ist es, die Performanz nativer RDF-Datenbanken für Anfragen mit ortsbasierten Anteilen zu verbessern. Die zur Beantwortung einer solchen Anfrage notwendigen Joins sind gemäß einer im Datenbank-Bereich üblichen Heuristik hierzu in einer Reihenfolge auszuführen, die die Menge der resultierenden Tupel schon frühzeitig möglichst weitgehend einschränkt. Zur Bestimmung einer derartigen Reihenfolge ist es also unter anderem notwendig, abschätzen zu können, wie viele Tupel aus einem Join mit dem Geoindex resultieren. Hierzu muss zunächst einmal die Anzahl der innerhalb des Geoindex qualifizierten Objekte, d.h. die Kardinalität einer der beiden *Eingabemengen* eines solchen Joins, abgeschätzt werden. Die Entwicklung hierfür geeigneter Konzepte und Methoden stellt den ersten Teil der Aufgabenstellung dar. Anschließend kann anhand besagter Kardinalität, der Kardinalität des Join-Partners sowie einer im Folgenden näher zu bestimmenden Selektivität des Joins die Kardinalität der Ergebnismenge abgeschätzt werden. Im zweiten und gleichzeitig Hauptteil der Arbeit sollen daher Konzepte und Methoden entwickelt werden, um ebendiese *Join-Selektivität* möglichst genau abschätzen zu können.
Zur Lösung der Aufgabe soll zunächst analysiert werden, wie bestehende relationale und auch andere DBVS vergleichbare Problemstellungen lösen bzw. welche Ansätze zur Vorhersage der zu bestimmenden Kenngrößen bereits vorgeschlagen wurden. Auf dieser Basis sind anschließend ein für den RDF-Kontext geeignetes Kostenmodell für Joins mit dem Geoindex, sowie die zur Anwendung dieses Kostenmodells notwendigen Statistiken zu entwickeln. In einem weiteren Schritt ist dieses Modell dann in eine vorhandene native RDF-Datenbank zu integrieren, die durch Erweiterung um einen Geoindex aus dem System RDF-3X hervorging [BNM10]. Das resultierende System soll abschließend anhand von Messungen mit geeigneten Testdaten einer ausführlichen Evaluation unterzogen werden.

1.3 Aufbau der Arbeit

Diese Diplomarbeit ist im Weiteren wie folgt gegliedert:
Kapitel 2 führt zunächst die zum Verständnis notwendigen Grundlagen mitsamt der zugehörigen Terminologie ein. Hierauf aufbauend werden in Kapitel 3 verwandte Arbeiten auf dem Gebiet der Kardinalitäts- und Selektivitätsbestimmung diskutiert, welche als Ausgangspunkt für den in Kapitel 4 entwickelten Lösungsansatz dienen. Kapitel 5 widmet sich anschließend der Implementierung dieses Ansatzes auf Basis der um einen Geoindex erweiterten Datenbank RDF-3X [BNM10]. Die Leistungsfähigkeit des daraus resultierenden Systems wird in Kapitel 6 durch umfangreiche Messungen untersucht und mit der

Leistungsfähigkeit der in [BNM10] beschriebenen, bestehenden Variante von RDF-3X verglichen, um den praktischen Nutzen des Lösungsansatzes abschätzen zu können. Kapitel 7 fasst die Diplomarbeit abschließend noch einmal zusammen und gibt einen Ausblick auf mögliche weitere Forschungsanstrengungen in diesem Bereich.

1.4 Danksagungen

Ich danke Andreas Brodt für die kompetente Betreuung und Prof. Dr.-Ing. habil. Bernhard Mitschang für die Möglichkeit, diese Arbeit in seiner Abteilung anfertigen zu können. Des Weiteren danke ich Oliver Schiller für diverse spontane Fachdiskussionen und daraus resultierende Anregungen. Ebenso zu Dank verpflichtet bin ich Bastian Reitschuster und Tim Waizenegger für die Durchführung der Messungen sowie Thomas Neumann für die Bereitstellung seines Artikels über die Characteristic Sets noch vor der offiziellen Publikation. Zu guter Letzt danke ich insbesondere auch meiner gesamten Familie für die jahrelange finanzielle und moralische Unterstützung!

2 Grundlagen

Dieses Kapitel führt die zum Verständnis der weiteren Arbeit notwendigen Grundlagen ein. Abschnitt 2.1 beschreibt dazu noch einmal in detaillierter Weise die relevanten Eigenschaften des zugrund liegenden Datenmodells RDF sowie der zugehörigen Anfragesprache SPARQL. Abschnitt 2.2 widmet sich anschließend den möglichen Ansätzen zur Implementierung RDF-fähiger DBVS. Ein derartiges System ist RDF-3X. Es ist darüber hinaus Grundlage dieser Arbeit und wird daher in 2.3 ausführlich beschrieben. Abschnitt 2.4 erläutert im Anschluss daran die in [BNM10] zum Zwecke der Unterstützung ortsbasierter RDF-Daten vorgenommenen Erweiterungen des Systems. Abschließend widmet sich Abschnitt 2.5 der Optimierung von Zugriffsplänen für RDF-Daten.

2.1 RDF und SPARQL

Zunächst sollen noch einmal die grundlegenden Technologien RDF und SPARQL dargestellt werden. Dabei wird auch auf Details eingegangen, die zum Verständnis der nachfolgenden Kapitel notwendig sind. Es sei an dieser Stelle jedoch darauf hingewiesen, dass die ausgewählten Aspekte keinesfalls den gesamten Leistungsumfang von RDF und SPARQL widerspiegeln!

2.1.1 RDF - Das Datenmodell

RDF [KC04] ist ein vom W3C spezifiziertes Datenmodell, das vorwiegend zur Strukturierung, Beschreibung und Verknüpfung der im Internet verfügbaren Informationen gedacht ist. Hierzu wird jegliche Information in Form von Tripeln $(Subjekt, Prädikat, Objekt)$ ausgedrückt, wodurch ein Zusammenhang zwischen *Subjekt* und *Objekt* dargestellt wird, der sich durch *Prädikat* beschreiben lässt.
Die Darstellung in Form von Tripeln dient hierbei sowohl der Repräsentation von Attributen betrachteter Entitäten, als auch der Repräsentation von Beziehungen zwischen diesen. Somit entspricht das *Prädikat* in gewisser Weise der Spaltenbezeichnung in einer äquivalenten relationalen Darstellung. Jedoch mit dem Unterschied, dass das Prädikat (und damit die Spaltenbezeichnung) in RDF sehr viel freier für jede Entität individuell gewählt werden kann. Dies führt zu einer weitaus flexibleren Modellierung, die sich nicht der Form eines vorgegebenen Datenbankschemas unterordnen muss. Vielmehr ermöglichen Modellierungen in RDF die langfristige Herausarbeitung eines Schemas, wobei Verwandtschaftsbeziehungen zwischen Attributen und Beziehungen („a und b sind unterschiedliche Bezeichnungen für ein und den selben Zusammenhang“) wiederum in RDF selbst dargestellt werden

können! Das Datenmodell unterscheidet somit nicht zwischen der Modellierung der eigentlichen Information und der Darstellung von Metadaten. Aus diesem Grund ist RDF zur Integration semi-strukturierter Daten aus einer Vielzahl unterschiedlicher Quellen geradezu prädestiniert.
Durch die Beschränkung auf das Tripel-Konzept werden Daten in RDF zwangsläufig in einer sehr feingranularen Struktur beschrieben und es werden nicht etwa wie im relationalen Modell mehrere Attribute in einer Tabelle zusammengefasst. Diese Eigenschaft hat, wie in Abschnitt 2.2 genauer erläutert wird, schwerwiegende Auswirkungen hinsichtlich der Verarbeitung in RDF modellierter Daten.

Die Tripel-Bestandteile *Subjekt* und *Prädikat* enthalten als Werte grundsätzlich Ressourcen im Sinne unterscheidbarer Datenelemente mit zugehörigem Bezeichner, die mittels URIs spezifiziert werden. Das *Objekt* kann sowohl Ressourcen wie auch Literale als Wert annehmen. Zur Vereinfachung der Darstellung lassen sich dabei für URIs mit der Syntax `<URI> ::= <Präfix>:<ID>` zuvor definierte Präfixe verwenden, innerhalb derer die jeweilige ID eindeutig ist. Durch die Nutzung von URIs wird es möglich, ein und die selbe Entität, Eigenschaft oder Beziehung in voneinander unabhängigen Aussagen immer wieder so zu verwenden, dass die resultierenden Tripel stets die selbe Ressource referenzieren. Literale hingegen werden überall dort verwendet, wo eine solche Verknüpfung nicht notwendig ist oder keine geeignete URI existiert. Sie werden in doppelte Anführungszeichen eingeschlossen und standardmäßig als Zeichenketten interpretiert. Mit Hilfe der Syntax `<Literal> ::= "<Wert>"^^<Typ>` ist es jedoch auch möglich, Literale zu typisieren (z. B. als Float, Vektor, etc.), so dass die Wert-Strings in geeigneter Weise interpretiert und verarbeitet werden können. Zur Angabe des Typs werden hierbei wiederum URIs verwendet.
Die Interpretation von RDF-Daten als Graph (s. beispielsweise Abbildung 1.1) wurde bereits in der Einleitung umfassend eingeführt. Aus diesem Grund wird hier lediglich darauf verwiesen.

2.1.2 SPARQL - Die Anfragesprache

SPARQL [Pru07] ist eine ebenfalls durch das W3C standardisierte Anfragesprache zur gezielten Extraktion von Daten aus einer vorhandenen RDF-Datenbank. SPARQL-Anfragen führen in deklarativer Weise eine Art Pattern-Matching auf dem RDF-Graphen durch, der sich als Interpretation der Gesamtheit aller Tripel der Datenbank ergibt. Hierzu wird mittels „query by example“ ein Muster spezifiziert und es werden sämtliche Abschnitte des RDF-Graphen gesucht, die diesem Muster entsprechen. In einer etwas detaillierteren Sichtweise als in Kapitel 1 hat eine SPARQL-Anfrage im Wesentlichen die Form

```
1  SELECT Variablenliste WHERE {
2    Subjekt_1  Praedikat_1  Objekt_1 .
3    Subjekt_2  Praediakt_2  Objekt_2 .
4    ...
5    FILTER Filter_URI (Argumentliste) .
6    ...
7  }
```

Die Zeilen 2 bis 4 stellen hierbei das eigentliche Muster dar. Es besteht in diesem Fall aus zwei Tripel-Mustern (den Zeilen 2 und 3), kann jedoch durch Hinzufügen zusätzlicher Tripel-Muster beliebig erweitert werden, was durch die Punke in Zeile 4 ausgedrückt werden soll. Die einzelnen Tripel-Muster beschreiben Mengen von RDF-Tripeln bzw. Kanten im RDF-Graph. Die in ihnen enthaltenen Elemente *Subjekt*, *Prädikat* und *Objekt* können jeweils Konstanten oder aber Variablen sein. Konstanten werden dabei durch den jeweiligen Wert ausgedrückt, Variablen durch einen eindeutigen Bezeichner mit vorangestelltem Fragezeichen. Als Konstanten-Werte bzw. Variablen-Bindungen kommen gemäß der Einschränkungen von RDF (vgl. Abschnitt 2.1.1) für *Subjekt* und *Prädikat* lediglich URIs in Frage. Das *Objekt* kann darüber hinaus auch ein Literal sein.
Im Falle von Konstanten ist das Muster an der jeweiligen Stelle klar bestimmt. Im Falle von Variablen werden diese an die jeweils korrespondierenden Werte in den sich als Anfrageresultat qualifizierenden Abschnitten des RDF-Graphen bzw. den zugrunde liegenden RDF-Tripeln gebunden.

Die Punkte am Ende der Zeilen 2 und 3 drücken Konjunktionen aus. Dadurch qualifizieren sich sämtliche *Kombinationen* von Tripeln, bei denen das erste Tripel dem Muster aus Zeile 2 *und* das zweite Tripel dem Muster aus Zeile 3 entspricht. Zur Bestimmung dieser Kombinationen muss also - zumindest konzeptionell - das Kreuzprodukt der jeweils qualifizierten Tripel-Mengen berechnet werden. Tritt in verschiedenen Zeilen des Musters die selbe Variable auf, so bedeutet dies eine weitere Einschränkung der Menge qualifizierter Kombinationen, da nur Kombinationen mit *identischen* Werten an den entsprechenden Positionen als Resultat der Anfrage zurückgegeben werden. Eine derartige Selektion auf einem Kreuzprodukt wird üblicherweise als Join-Operationen zusammengefasst und stellt einen der wichtigsten Operatoren in relationalen DBVS dar.

Die Menge qualifizierter Tripel-Kombinationen kann durch die Angabe von Filtern wie in Zeile 5 weiter eingeschränkt werden. Das intendierte Verhalten des Filters kann dabei mittels des Parameters `Filter_URI` in Form einer URI angegeben werden. Die Argumentliste kann beliebige Variablen des Musters sowie frei wählbare Konstanten enthalten.
Aus der Menge der gebundenen Variablen kann mittels `Variablenliste` am Ende eine beliebige Teilmenge ausgewählt werden, um die Werte dieser Variablen für sämtliche qualifizierten Abschnitte des Graphen als Resultat der Anfrage zurückzugeben. Dabei kann mittels der voranzustellenden Schlüsselwörter `COUNT` und `DISTINCT` optional die Anzahl solcher Abschnitte bestimmt bzw. eine Duplikateliminierung vorgenommen werden.
Zum Zwecke einer übersichtlicheren Darstellung ist eine alternative Schreibweise möglich, wenn sich die nachfolgenden Tripel-Muster auf das selbe Subjekt beziehen: In diesem Fall wird der Punkt am Ende der Zeile durch ein Semikolon ersetzt und das Subjekt in der darauf folgenden Zeile weggelassen.

2.2 DBVS für RDF-Daten: Triple-Stores

Nachdem in den Abschnitten 2.1.1 und 2.1.2 das Datenmodell RDF sowie die darauf aufbauende Anfragesprache SPARQL eingeführt wurden, sollen nun DBVS betrachtet werden,

die den Zugriff auf eine RDF-Datenbasis mit Hilfe einer SPARQL-Schnittstelle ermöglichen. Aufgrund der Tripel-Struktur von RDF-Daten werden derartige Systeme auch als RDF-Triple-Stores bezeichnet.

Typische Anfragen an einen Triple-Store selektieren eine oder mehrere Entitäten anhand ihrer Eigenschaften sowie einer oder mehrerer Beziehungen zwischen den Entitäten. Aufgrund der feingranularen Datenmodellierung in RDF resultieren hieraus in beiden Fällen Anfrage-Muster mit einer Vielzahl enthaltener Tripel-Muster. Wie in Abschnitt 2.1.2 erläutert, erfordert dies bei der Anfrageauswertung wiederum mehrere, je nach Anfrage sogar sehr viele Join-Operationen. Da Joins jedoch bei einer Eingabe der Länge n bis zu $\mathcal{O}(n^2)$ Objekte betrachten müssen, stellen sie eine extrem teure Operation dar. Gleichzeitig kommen RDF und SPARQL primär in Anwendungsszenarien mit einer umfangreichen Datenbasis zum Einsatz. Um trotzdem eine ausreichende Performanz des Systems gewährleisten zu können ist es somit notwendig, diese Joins in effizienter Weise zu berechnen. Darüber hinaus werden Mechanismen benötigt, um die Eingabemengen der Joins ressourcensparend bereitzustellen, so dass nicht für jedes Tripel-Muster die Gesamtheit aller Tripel der Datenbasis nach entsprechenden Kandidaten durchsucht werden muss.

Die derzeit verfügbaren RDF-Triple-Stores lösen diese Aufgaben in unterschiedlicher Weise. Ein verbreiteter Ansatz zur Beherrschung der Join-Problematik besteht darin, die Tripel gemäß ihrer jeweiligen Prädikate zu partitionieren und in verschiedenen Tabellen, sogenannten Property-Tables, abzulegen. Hierdurch können den Joins bei feststehenden Prädikaten *gezielt* die Einträge der entsprechenden, verhältnismäßig kleinen, Tabellen als Eingabe zugeführt werden. Um sich bei der Implementierung von Triple-Stores auf die RDF-spezifischen Besonderheiten der Verarbeitung konzentrieren zu können, bauen viele derartige Systeme auf vorhandenen relationalen DBVS auf.
Zu dieser Klasse gehören unter anderem die frühen Ansätze Jena [WSKR03] und Sesame [BKH01]. Insbesondere bei Jena liegt das Hauptaugenmerk der Verarbeitung jedoch noch auf der grundsätzlichen Bereitstellung entsprechender Funktionalität und nicht auf der Fähigkeit zur Verarbeitung großer Datenmengen. Jena baut dabei auf einer beliebigen relationalen Datenbank auf; Sesame verwendet PostgreSQL [Posa] als Grundlage. In eine ähnliche Richtung geht die in [CDES05] beschriebene Implementierung, die jedoch auf einem kommerziellen relationalen DBVS aufsetzt. Mit dem 2007 publizierten und auf C-Store [SAB+05] basierenden System SW-Store [AMMH07] kam erstmalig eine praktische Nutzbarkeit der RDF-Technologie für große Datenmengen in Betracht.

Leider tritt der oben beschriebene positive Effekt einer vertikalen Partitionierung in Property-Tables auf die Join-Bearbeitung nur dann in Erscheinung, wenn in der entsprechenden Anfrage überwiegend *Konstanten* als Prädikate auftreten. Wie aus Abschnitt 2.1.2 hervorgeht ist dies im Allgemeinen jedoch nicht gegeben. Wird hingegen das konkrete Prädikat offen gelassen, so sind im Falle vertikaler Partitionierung *sämtliche* Property-Tables als Eingabe für den Join zu verwenden! Für die effiziente Verarbeitung derart umfangreicher Eingabedaten werden in den entsprechenden Systemen jedoch in aller Regel keine Vorkehrungen getroffen, so dass hier ein deutlicher Leistungseinbruch zu erwarten ist.
Um diese Beschränkung der Leistungsfähigkeit auf eine Teilmenge aller theoretisch möglichen Anfragen zu überwinden, werden auch alternative Lösungsansätze in Betracht gezo-

gen. Da die Tripel in RDF nicht anhand eines Datenbankschemas geordnet, sondern vielmehr aus Sicht des DBVS alle gleichförmig sind, liegt es nahe, diese Tatsache auch in der physischen Organisation der Daten zu reflektieren und die Tripel in einer einzigen großen Tabelle zusammenzufassen.
Auf konzeptioneller Ebene ist bei diesem Ansatz zunächst mit einem Leistungseinbruch zu rechnen, da angesichts der nicht vorhandenen Partitionierung stets die gesamte Faktentabelle als Eingabemenge für die Joins verwendet werden muss. Durch umfangreiche Indexierung der Faktentabelle ist diese Problematik jedoch beherrschbar. Mit Hilfe eines passenden Index können dabei gezielt und *ausschließlich* die relevanten Tripel als Eingabe für Joins herangezogen werden.
Einen derartigen Ansatz verfolgen die Systeme Hexastore [WKB08] und RDF-3X [NW10], die darüber hinaus native RDF-Systeme darstellen und somit *nicht* auf relationalen DBVS aufsetzen, was im Allgemeinen einen deutlichen Leistungsschub mit sich bringt. Hexastore wurde jedoch bislang lediglich für Datensätze evaluiert, die vollständig im Hauptspeicher untergebracht werden können. Dies ist für den im Rahmen der vorliegenden Diplomarbeit betrachteten Anwendungskontext allerdings im Allgemein nicht möglich. Des Weiteren handelt es sich bei Hexastore um einen in Python implementierten Prototyp, der erwartungsgemäß weit weniger effizient arbeitet, als das in C++ implementierte RDF-3X. Aus diesem Grunde wurde letzteres als Ausgangspunkt für die Integration von Geo-Funktionalität in eine native RDF-Datenbank verwendet [BNM10]. Das resultierende System GeoRDF-3X soll im Zuge dieser Diplomarbeit weiter verbessert werden.

Der Vollständigkeit halber seien an dieser Stelle auch noch das RDF-Modul von Virtuoso Universal Server [Vir] sowie BitMat [ACZH10] genannt, die beide jedoch gänzlich andere Ansätze verfolgen, indem sie durch massiv komprimierte, vorberechnete Join-Indexe die kritische Join-Operation zu beschleunigen versuchen.

2.3 RDF-3X

RDF-3X ist ein nativer RDF-Triple-Store, d.h. es baut *nicht* auf einem vorhandenen relationalen DBVS auf, sondern implementiert die notwendigen Speicherungsstrukturen selbst in einer für RDF-Daten optimierten Art und Weise. Sämtliche aus der Miniwelt abgeleiteten Tripel werden dabei in einer einzigen großen „Faktentabelle“ gespeichert und somit im Gegensatz zu Jena [WSKR03], SW-Store [AMMH07] und anderen Systemen *nicht* anhand ihrer Prädikat-Werte vertikal partitioniert.
Statt dessen verwendet RDF-3X zur Beschleunigung der kritischen Join-Operation eine massive Indexierung der Faktentabelle in Kombination mit darauf aufbauenden schnellen Merge-Joins.
Es sei darauf hingewiesen, dass RDF-3X zum momentanen Zeitpunkt nur eine begrenzte Teilmenge der von RDF und SPARQL prinzipiell ermöglichten Funktionalität bereitstellt. Vielmehr dient das System derzeit zur Untersuchung zentraler Implementierungsaspekte, die für die praktische Nutzbarkeit von RDF jedoch von entscheidender Bedeutung sind.

Die im Rahmen dieser Diplomarbeit relevanten Implementierungskonzepte von RDF-3X sollen in den nachfolgenden Unterabschnitten genauer erläutert werden. 2.3.1 beschäftigt sich dabei zunächst mit der globalen Architektur des Systems, deren Komponenten daran anschließend im Einzelnen genauer beschrieben werden.

2.3.1 Architektur

Die globale Architektur von RDF-3X ist in Abbildung 2.1 dargestellt.

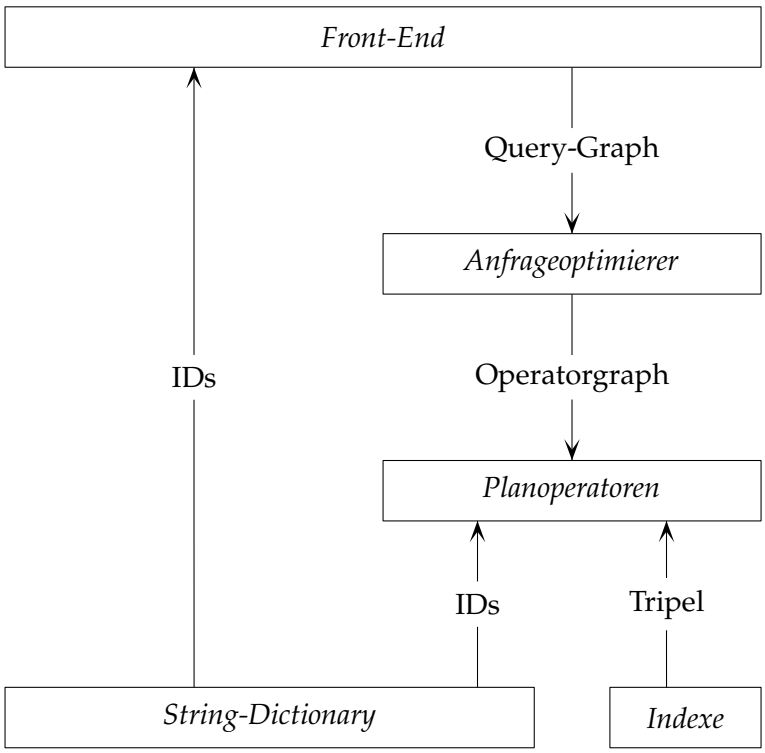


Abbildung 2.1: Architektur von RDF-3X

Das *Front-End* realisiert hierbei die Benutzerschnittstelle zur Entgegennahme von SPARQL-Anfragen. Diese werden anschließend einer lexikalischen, syntaktischen und semantischen Analyse unterzogen, an deren Ende eine Repräsentation der Anfrage in Form eines Anfragegraphen steht. Ein solcher Graph hat im Wesentlichen die in [SSB+08] beschriebene Struktur. Des Weiteren werden die enthaltenen Konstanten, d.h. URIs sowie Literale (vgl. Abschnitt 2.1.1), mit Hilfe des *String-Dictionary* auf jeweils eindeutige Integer-IDs abgebildet. Die Variablen werden einer ähnlichen Abbildung unterzogen, so dass die interne Darstellung der Tripel gänzlich mittels Integern erfolgen kann.
Der *Anfrageoptimierer* erstellt anschließend auf Basis des Anfragegraphen alle zur Beantwortung der Anfrage in Betracht kommenden Operatorgraphen und bewertet deren Kosten anhand eines geeigneten Kostenmodells.

Der bezüglich des verwendeten Kostenmodells günstigste Operatorgraph wir darauf hin mit Hilfe der verfügbaren *Planoperatoren* realisiert. Die jeweiligen Planoperatoren erhalten ihre Eingabedaten in Form von Tripeln aus passenden *Indexen* über der Faktentabelle. Am Ende der Verknüpfung dieser Eingabe-Tripel gemäß des gefundenen Operatorplans werden die in den Ergebnistupeln enthaltenen IDs wiederum mit Hilfe des *String-Dictionary* auf die jeweiligen Konstanten abgebildet und als Resultat an den Benutzer zurückgegeben.

2.3.2 String-Dictionary

Die in RDF-Tripeln und SPARQL-Anfragen verwendeten Konstanten stellen aus Sicht des DBVS - unabhängig von deren Unterteilung in URIs und RDF-Literale - in jedem Fall String-Literale variabler Länge dar. Eine derartige Darstellung impliziert jedoch gewisse Nachteile hinsichtlich der maschinellen Verarbeitung derartiger Literale. Zum einen weisen bereits Strings mittlerer Länge aus informationstheoretischer Sicht einen hohen Grad an Redundanz auf. Zum anderen sind Vergleiche auf Strings verhältnismäßig aufwendig.
Beide Probleme werden in RDF-3X umgangen, indem die Konstanten der Datenbasis beim Laden auf (fortlaufende) Integer-IDs abgebildet werden. Letztere stellen eine sehr viel effizientere Form der Codierung dar *und* ermöglichen Vergleiche in einem Schritt. Wie bereits angedeutet werden die in SPARQL-Anfragen enthaltenen Konstanten auf die selben IDs abgebildet, so dass ein Vergleich mit vorhandenen Daten möglich wird.
Eine solche Umwandlung zur effizienteren internen Verarbeitung findet im Übrigen nicht nur in RDF-3X, sondern in sehr vielen RDF-Triple-Stores statt.

Zur Realisierung der Vorwärts-Abbildung, d.h. der Abbildung von String-Literalen auf Integer-IDs verwendet RDF-3X normale B^+-Bäume. Da die zugeordneten IDs frei wählbar sind können diese zur Adressierung innerhalb eines Arrays verwendet werden, dessen Einträge wiederum auf die Speicheradressen der zugehörigen Strings verweisen. Damit kann die Rückwärts-Abbildung, d.h. die Abbildung von IDs auf String-Literale mittels Direct Mapping (vgl. [EGK95]) erfolgen. Durch die höhere Packdichte der entsprechenden Index-Seiten und die Beschränkung auf maximal zwei Seitenzugriffe kann die Effizienz der Abbildung dadurch gegenüber einem B^+-Baum weiter gesteigert werden.
Eine möglichst effiziente Rückwärts-Abbildung ist notwendig, da insbesondere im Falle umfangreicher Ergebnis-Mengen für die Ausgabe eine Vielzahl an IDs in die entsprechenden String-Literale umzuwandeln sind. Trotz des beschriebenen Mechanismus sind derartige Ausgabe-Operationen relativ aufwendig, so dass RDF-3X für umfangreiche Ergebnis-Mengen nur unzureichend skaliert.

Aufgrund der durch das String-Dictionary ermöglichten Darstellung der Faktentabelle mittels Integer-IDs amortisiert sich der für die beiden Indexe zusätzlich notwendige Speicheraufwand in der Regel bereits für relativ kleine Faktentabellen.

2.3.3 Indexe

SPARQL-Anfragen beinhalten sehr oft Tripel-Muster, die einem Teil der enthaltenen Elemente (Subjekt, Prädikat und Objekt) Konstanten zuweisen, den restlichen Teil jedoch in Form von Variablen offen lassen. Ein naiver Ansatz zur Bestimmung der dadurch beschriebenen Tripel-Menge bestünde darin, die vollständige Faktentabelle zu durchlaufen und jedes Tripel mit dem vorgegebenen Muster zu vergleichen. Eine (umfangreiche) Anwendung dieses Vorgehens in konkreten Systemen verbietet sich jedoch aufgrund des damit verbundenen Aufwands. Aus diesem Grund verwenden DBVS Indexe, mit deren Hilfe die Menge der betrachteten Daten in eine lineare Ordnung gebracht werden kann. Im Fall der betrachteten Tripelmenge bietet sich hierfür die lexikographische Ordnung bzgl. einer der $3! = 6$ Konkatenationen der Tripel-Komponenten an. Bilden die Konstanten einen Präfix dieser Konkatenation, so entspricht die vom zugehörigen Tripel-Muster beschriebene Menge einem Intervall auf besagter Ordnung. Dadurch wird es möglich, die qualifizierten Tripel gezielt aufzusuchen, anstatt die vollständige Faktentabelle durchlaufen zu müssen.
Allerdings sind je nach konkreter Verteilung der Konstanten bzw. Variablen im Muster unterschiedliche Konkatenationen notwendig, um eine solche *zusammenhängende* Aufzählung der qualifizierten Tripel mit Hilfe der linearen Ordnung zu ermöglichen. RDF-3X folgt vor diesem Hintergrund dem Brute-Force-Ansatz, *alle* derartigen Indexe vorzuhalten. Möglich wird dies, durch eine weiter unten beschriebene Komprimierung der Speicherseiten sowie die im RDF-Kontext relativ kleine Zahl von $3! = 6$ (vgl. oben) möglichen Indexen.
Als Datenstruktur für die Indexe wurden geclusterte B^+-Bäume ausgewählt. Die Blattknoten beinhalten also *direkt* die jeweiligen Tripel und es ist somit keine weitere Seitenreferenz notwendig, um die eigentlichen Nutzdaten aufzusuchen. Entsprechend ihrer jeweils zugrunde liegenden Permutation der Tripel-Komponenten Subjekt, Prädikat und Objekt werden die Indexe für vollständig bestimmte Tripel (d.h. alle drei Komponenten sind durch Konstanten festgelegt) nachfolgend mit $\mathcal{I}_{SPO}$, $\mathcal{I}_{SOP}$, $\mathcal{I}_{PSO}$, $\mathcal{I}_{POS}$, $\mathcal{I}_{OSP}$ und $\mathcal{I}_{OPS}$ bezeichnet.

Oftmals ist es zur Beantwortung von Anfragen ausreichend, für ein gegebenes Tripel-Muster nicht die konkreten Variablen-Bindungen qualifizierter Tripel, sondern lediglich deren Anzahl zu kennen. Hierzu werden in RDF-3X weitere, sogenannte aggregierte, Indexe vorgehalten. Die Konstanten des Tripel-Musters bilden auch hier den Schlüssel, ihnen zugeordnet wird die jeweilige Anzahl entsprechender Tripel. Zur Realisierung der Indexe wurden auch in diesem Fall B^+-Bäume verwendet. Angesichts der möglichen Verteilungen einer oder zweier Konstanten auf die drei Komponenten ergeben sich somit $3 + 6 = 9$ weitere Indexe, die mit $\mathcal{I}_S$, $\mathcal{I}_P$, $\mathcal{I}_O$, $\mathcal{I}_{SP}$, $\mathcal{I}_{SO}$, $\mathcal{I}_{PS}$, $\mathcal{I}_{PO}$, $\mathcal{I}_{OS}$ und $\mathcal{I}_{OP}$ bezeichnet werden.

Da benachbarte Einträge in den Blattseiten der Indexe oftmals ähnliche Schlüssel besitzen und die Schlüssel maßgeblicher Bestandteil des Eintrags sind, wird eine Differenz-Codierung benachbarter Einträge vorgenommen, um das Datenvolumen der Indexe zu reduzieren. Die Differenz-Codierung erfolgt jedoch nur innerhalb einzelner Blattseiten, so dass lediglich die jeweilige Seite bereitgestellt und decodiert werden muss, um Inhalte daraus zu extrahieren. Damit ergibt sich ein vertretbares Gesamtdatenvolumen, zumal die aggregierten Indexe angesichts ihrer reduzierten Zahl an Schlüssel-Komponenten nur verhältnismäßig wenige Einträge aufweisen und somit klein sind.

Der dargestellte Ansatz massiver Indexierung führt insb. bei ausgesprochen selektiven Anfragen zu Leistungsvorteilen, da mittels der Indexe gezielt die relevanten Tripel aufgesucht werden können. Bei wenig selektiven Anfragen überwiegt der Aufwand zur Rückwärts-Abbildung der IDs durch das String-Dictionary, so dass RDF-3X hier Leistungseinbußen verzeichnen muss.

2.3.4 Planoperatoren

Wie andere DBVS auch, beinhaltet RDF-3X Planoperatoren für verschiedenste Aufgaben. Im Rahmen dieser Diplomarbeit sind jedoch primär Operatoren zum Scan über die in Abschnitt 2.3.3 vorgestellten Indexe, zur Join-Bildung sowie zur Realisierung von Filter-Bedingungen relevant.

Scan-Operatoren gestatten das schrittweise Auslesen der durch ein Tripel-Muster beschriebenen RDF-Tripel mittels eines geeigneten Index. Die Konstanten des Tripel-Musters bilden dabei einen Präfix des Schlüssels und wählen somit einen Unterbaum des zugrunde liegenden B^+-Baums aus, der komplett durchlaufen wird, um seine Einträge an die nachfolgenden Operatoren im Operatorgraph weiterzureichen. Die übergebenen Tripel sind dabei gemäß der ersten auf den Konstanten-Präfix folgenden Tripel-Komponente sortiert.
Zur Join-Bildung können Operatoren für Merge-Joins und Hash-Joins herangezogen werden. Der Merge-Join-Operator benötigt hierzu lediglich lineare Zeit in der Kardinalität der Eingabemengen. Er setzt jedoch deren Sortierung gemäß der Join-Variablen voraus und kann somit von der massiven Indexierung in RDF-3X profitieren.
Der Hash-Join besitzt asymptotisch ebenfalls linearen Aufwand, benötigt jedoch einen gewissen Mehraufwand zum Aufbau der Hash-Tabelle. Im Gegenzug kommt er jedoch auch mit Eingabedaten zurecht, die nicht gemäß der Join-Variable sortiert vorliegen.
Des Weiteren stehen zur Realisierung der in Abschnitt 2.1.2 beschriebenen `FILTER`-Klausel verschiedene Operatoren zur Verfügung, die sämtliche ihrer Eingabetupel analysieren und nur solche weiterreichen, die dem entsprechenden Filter-Prädikat genügen.

Die genannten Operatoren können allesamt (konzeptionell) parallel ausgeführt werden, so dass die Daten auf Tupel-Basis zwischen den Operatoren weitergereicht werden und eine aufwendige Zwischenspeicherung entfällt.

2.3.5 Sideways Information Passing

Sideways Information Passing, kurz SIP [NW09], ist ein in RDF-3X integrierter Mechanismus, der es erlaubt, bereits statisch optimierte Operatorgraphen zur Laufzeit effizienter auszuführen und dadurch noch weiter zu verbessern. Er beruht auf der Beobachtung, dass Operatoren oftmals Tupel generieren, die durch einen der nachfolgenden Operatoren aufgrund fehlender Join-Partner später verworfen werden. Angesichts der parallelen Ausführung von Operatoren ist es jedoch möglich, vorangehende Operatoren frühzeitig darüber

zu informieren, welche Tupel später keine Verwendung finden können und somit erst überhaupt nicht generiert werden müssen. Sideways Information Passing etabliert hierzu Kommunikationskanäle zwischen den Operatoren eines Operatorgraphen, die hiermit gegenseitig entsprechend hilfreiche Informationen austauschen können.
Es handelt sich somit um einen reinen Laufzeit-Mechanismus, der mit der Aufgabenstellung dieser Diplomarbeit nur am Rande zu tun hat. Es dürfte jedoch interessant sein, inwiefern statisch optimierte Operatorgraphen mit Einbeziehung des Geoindex einen Einfluss auf die Effektivität von SIP haben bzw. inwieweit SIP den (vermeintlichen) Effizienzgewinn durch statische Optimierung zu verfälschen in der Lage ist.

2.4 Unterstützung für ortsbasierte RDF-Daten

Die bislang beschriebenen Konzepte und DBVS erlauben noch keine Verarbeitung ortsbezogener Daten mittels Koordinatendarstellung. Ziel dieser Arbeit ist es jedoch, durch geeignete Statistiken und Verfahren, die Integration ortsbasierter Daten in einen nativen RDF-Triple-Store zu optimieren. Aus diesem Grund gilt es zunächst einmal, zu klären, *wie* solche ortsbasierten Daten überhaupt in ein bestehendes System integriert werden können bzw. wie dies im Rahmen von GeoRDF-3X [BNM10] geschieht.
Hierzu beschreibt der nachfolgende Abschnitt 2.4.1 zunächst einmal, was *genau* unter dem Begriff „ortsbezogene Daten“ zu verstehen ist und welche Teilprobleme zur Integration derartiger Daten somit zu lösen sind. Anschließend widmen sich die Abschnitte 2.4.2 bis 2.4.4 der Frage, wie diese Teilprobleme grundsätzlich gelöst werden können bzw. wie sie in GeoRDF-3X gelöst wurden.

2.4.1 Problemstellung

Wie aus der Einleitung hervorgeht, sind unter dem Begriff „ortsbezogene Daten“ im Rahmen dieser Diplomarbeit geographische sowie geometrische Daten zu verstehen. Hiermit sind *Vektordaten* gemeint, die Objekte auf der Erdoberfläche bzw. in einem beliebigen geometrischen Raum mit Hilfe eines Koordinatensystems beschreiben. Eine derartige Darstellung erlaubt die Ableitung von Aussagen über räumliche Nähe, die Größe von Objekten und vieles andere mehr.
Um solche Aussagen in einem nativen RDF-Triple-Store nutzen zu können sind jedoch geeignete Modifikationen des Systems notwendig. Zunächst einmal muss das Datenmodell so erweitert werden, dass auch räumliche Objekte wie Punkte, Linienzüge und Polygone in adäquater Weise mittels ihrer Koordinaten dargestellt werden können. Abschnitt 2.4.2 widmet sich dieser Aufgabe. Des Weiteren ist ein Mechanismus zu entwickeln, der es erlaubt, spezifische Eigenschaften räumlicher Objekte, wie benachbarte Lage, Schnitt, etc. zur Selektion innerhalb einer SPARQL-Anfrage zu verwenden. Abschnitt 2.4.3 beschreibt einen solchen Mechanismus sowie mögliche Alternativen. Schlussendlich müssen auch noch eine geeignete Datenstruktur zur Verwaltung sowie ein Mechanismus zur Einbindung der Geodaten

in die Anfrageauswertung gefunden werden, mit deren Hilfe entsprechende Anfragen möglichst effizient beantwortet werden können. Abschnitt 2.4.4 befasst sich hiermit.

Von den beschriebenen Vektordaten zu unterscheiden sind sogenannte symbolische Ortsangaben wie die Raumnummer in einem Gebäude, der Name einer Straße oder der Name einer gesamten Stadt. Aus derartigen Daten können die genannten Eigenschaften *nicht* abgeleitet werden. Trotzdem haben auch symbolische Ortsangaben ihre Berechtigung und kommen beispielsweise bei der Lokalisierung von Objekten innerhalb eines Gebäudes bereits erfolgreich zum Einsatz [WHFaG92]. Aus diesem Grund wird in Abschnitt 2.4.2 unter anderem beschrieben, wie mit einer geeigneten Ontologie auch in GeoRDF-3X symbolische Ortsangaben (mit integrierten Vektordaten) verwendet werden können.

2.4.2 Erweiterung des Datenmodells

Ein naiver Ansatz zur Modellierung eines räumlichen Objekts, d.h. eines Punktes, eines Linienzuges oder eines Polygons, bestünde darin, sämtliche Bestandteile des Objekts wie Ecken, Kanten, etc., selbst als Ressourcen im Sinne von RDF zu modellieren. Dadurch könnten mittels herkömmlicher RDF-Tripel Eigenschaften wie Koordinaten zugewiesen bzw. die Bestandteile miteinander in Beziehung gebracht werden. Wird üblicherweise jedoch auf das Objekt *als Ganzes* zugegriffen, so ist diese Form der Modellierung unnötig kompliziert und ineffizient, da das Objekt jedes Mal aufwendig mittels Joins wieder zusammengesetzt werden muss. In [BNM10] wurde daher ein anderer Weg gewählt und ein solches Objekt mit Hilfe eines einzigen Geo-Literals dargestellt. Hierzu wird das von OpenGIS standardisierte Format WKT (Well-Known-Text) [Her06] verwendet und das Literal mit der dafür üblichen Syntax sowie einer geeigneten URI typisiert. Das resultierende Geo-Literal kann anschließend in Form eines Tripels mit beliebigem Prädikat an die Ressource gebunden werden, deren geometrische Eigenschaften modelliert werden sollen. Auf diese Weise wird eine maximale Kompatibilität mit beliebigen Ontologien ermöglicht.

Entsprechend diesem Ansatz kann das Beispiel aus der Einleitung nun wie folgt um eine Ortsangabe mit Breiten- und Längengrad erweitert werden (`geordf` sei ein geeigneter Namensraum):

```
(S1 a Rapid_Transit_Line)
(Feuersee a Station)
(Feuersee partOf S1)
(Feuersee locatedAt "POINT(48.7729 9.1670)"^^geordf:geography)
```

Die Graphinterpretation des Datensatzes hat dann die folgende Form:

De facto ist also überhaupt keine Erweiterung des Datenmodells notwendig. Es werden vielmehr die bestehenden Modellierungskonzepte in geeigneter Weise angewendet.

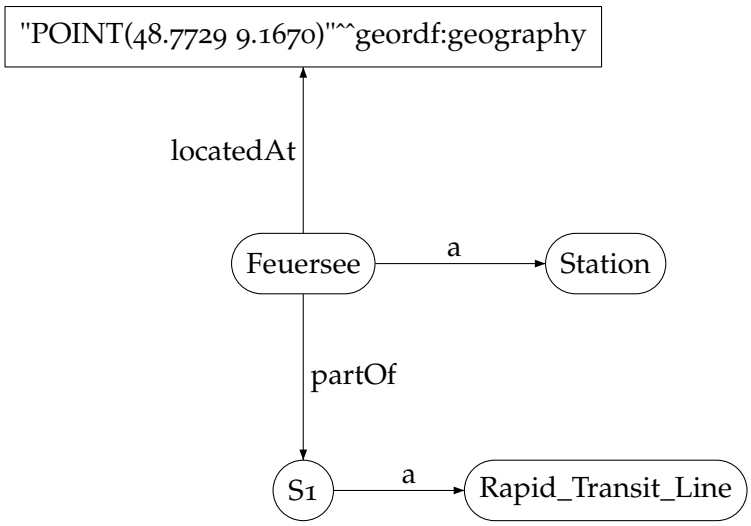


Abbildung 2.2: Graphinterpretation des erweiterten Beispiels

2.4.3 Erweiterung der Anfragesprache

Sollen Ressourcen anhand räumlicher Eigenschaften und Beziehungen selektiert werden können, so besteht wiederum ein naiver Ansatz darin, diese mittels herkömmlicher Tripel zu modellieren. Aufgrund der Vielzahl möglicher Beziehungen zwischen räumlichen Objekten wäre jedoch mit einem massiven Anwachsen der Faktentabelle und daraus resultierenden Effizienzeinschränkungen zu rechnen. Außerdem möchte man oftmals die Eigenschaften räumlicher Objekte mit *Wertebereichen* vergleichen, die erst in der Anfrage spezifiziert werden sollen. Beispielsweise könnte es von Interesse sein, ob ein bestimmtes Objekt innerhalb eines rechteckigen Ausschnitts liegt oder nicht. SPARQL kann jedoch lediglich auf die *Identität* von Werten hin, nicht aber auf Inklusion überprüfen. Daher ist ein von SPARQL unabhängiger Test der jeweiligen Relation anzustreben. GeoRDF-3X löst diese Aufgabe mit Hilfe der `FILTER`-Klausel: Dem Filter wird dabei eine Variable übergeben, die im Muster-Bereich der Anfrage entsprechend der verwendeten Ontologie an die Geo-Literale in Frage kommender Objekte gebunden wird. Des Weiteren können dem Filter Vergleichsparameter ebenfalls in Form von Geo-Literalen übergeben werden. Das auf die Parameter des Filters anzuwendende Prädikat schließlich wird in Form einer Filter-URI spezifiziert. GeoRDF-3X unterstützt verschiedene solcher Prädikate für ortsbasierte Filterungen. Im Rahmen dieser Arbeit soll jedoch lediglich das Prädikat `within` betrachtet werden.

Wiederum kann das Beispiel aus der Einleitung erweitert werden, um sämtliche Haltestellen zu bestimmen, die von der Haltestelle Universität aus per Direktverbindung zu erreichen sind *und* im Bereich der Stuttgarter Innenstadt liegen:

```
SELECT ?x WHERE {
  ?x partOf ?y .
  ?y a Rapid_Transit_Line .
  Universität partOf ?y .
  ?x locatedAt ?geo .
  FILTER geordf:within(?geo, "POLYGON((48.787 9.178, 48.775 9.167,
    48.769 9.182, 48.782 9.192, 48.787 9.178))"^^geordf:geography)
}
```

Auf diese Weise muss auch die Anfragesprache nicht erweitert werden. Statt dessen kommen bereits vorhandene Mechanismen zum Einsatz, so dass die resultierenden SPARQL-Anfragen vollständig kompatibel mit dem gängigen Standard sind.

2.4.4 Operatoren zur Einbindung von Geodaten

Zur Einschränkung der Werte an Geoliterale zu bindender Variablen kommen grundsätzlich zwei Arten von Planoperatoren in Betracht: Zum einen solche, die aus einer gegebenen Menge von Tupeln mit Geoliteralen die jeweils qualifizierten selektieren. Zum anderen solche, die zunächst die Menge der qualifizierten Geoliterale bestimmen und ausschließlich diese in die Konstruktion größerer Tupel einfließen lassen. GeoRDF-3X implementiert beide Arten von Operatoren und bezeichnet ersteren als Spatial-Selection, letzteren als Spatial-Index-Scan.

Der Spatial-Selection-Operator muss zur Beurteilung jedes einzelnen Tupels zunächst die ID des jeweiligen Geoliterals mit Hilfe des String-Dictionary in den dort hinterlegten, zugehörigen Wert auflösen. Für diesen kann anschließend mittels geeigneter Algorithmen bestimmt werden, ob er dem durch den Filter festgelegten Geo-Prädikat genügt, oder nicht. Die Auflösung durch das String-Dictionary erfordert jedoch teure Zugriffe auf mehr oder weniger zufällig verteilte Speicherpositionen und ist insbesondere vom Umfang der untersuchten Tupelmenge abhängig, so dass der Spatial-Selection-Operator in GeoRDF-3X lediglich an der Spitze des Operatorbaums, und damit auf einer tendenziell kleinen Tupelmenge angewendet wird.

Der Spatial-Index benötigt zunächst einmal eine geeignete Datenstruktur zur Bestimmung qualifizierter Geoliterale. Entsprechende Strukturen reflektieren räumliche Eigenschaften wie Position, Nähe, Schnitt, etc. und ermöglichen daher die gezielte Einschränkung der Resultatsmenge anhand entsprechender Kriterien. Aufgrund umfangreicher Forschungen wurden bislang eine ganze Reihe entsprechender Strukturen vorgeschlagen. Gemäß der Problemstellung in Abschnitt 2.4.1 müssen im vorliegenden Fall sowohl Punktobjekte als auch ausgedehnte Objekte wie Linienzüge und Polygone verwaltet werden können. Dies wird insbesondere durch den R-Baum [Gut84] und seine Varianten R^+-Baum bzw. R^*-Baum ermöglicht. Da es in [BNM10] primär darum ging, die grundsätzliche Machbarkeit der Intergration ortsbasierter Daten in einen nativen RDF-Triple-Store nachzuweisen, wurde zur Vereinfachung eine statische Datenbank zugrunde gelegt, die keinen Änderungsdienst beinhaltet. Auf diese Weise kann die Aufteilung der Objekte in einem R-Baum weitgehend

optimiert werden und ist keinerlei Änderungen zur Laufzeit unterworfen. Somit ergibt sich für die komplizierteren Strukturen R^+- und R^*-Baum kein nennenswerter Effizienzgewinn gegenüber dem einfacheren R-Baum. Folgerichtig wurde letzterer als Geoindex zur Integration ortsbasierter Daten in das System RDF-3X herangezogen. Verwendet wurde hierzu die Implementierung eines R-Baums aus der C++-Bibliothek libspatialindex [lsi]. Der Baum speichert dabei nicht nur die Geo-Objekte selbst, sondern auch die zugeordneten Integer-IDs der sie definierenden Geoliterale.

Die Integration des Geoindex zur Laufzeit geschieht durch einen speziellen Planoperator. Zur Beantwortung einer Anfrage ruft dieser gemäß der spezifizierten Filter-URI geeignete Methoden der Bibliothek libspatialindex auf. Als Vergleichsobjekte werden diesen die ebenfalls in der Filter-Klausel spezifizierten Geoliterale übergeben. Der Geoindex bestimmt anschließend anhand dieser Daten mit Hilfe geeigneter Verfahren die qualifizierten Geo-Objekte und liefert deren Integer-IDs an den Planoperator zurück. Dabei kommt das Prinzip „Filter & Refine“ zum Einsatz, d.h. es wird zunächst anhand des kleinsten, achsenparallelen und umschließenden Rechtecks aller verwalteten Geo-Objekte eine Obermenge der qualifizierten Objekte bestimmt. Anschließend wird mittels potentiell aufwändiger Berechnungen für jedes Element dieser Menge entschieden, ob es dem jeweiligen Filter-Prädikat genügt, oder nicht.
Die IDs der resultierenden Geo-Objekte werden im weiteren Verlauf durch spezielle Joins mit sämtlichen Tripel-Mustern der Anfrage verknüpft, in denen die in der Filter-Klausel spezifizierte Variable enthalten ist. Soll hierzu ein Merge-Join Verwendung finden, so sind die IDs vorher zu sortieren, da die Objekte im Geoindex anhand räumlicher Nähe gruppiert sind und die IDs somit unsortiert zurückgeliefert werden. Alternativ steht auch ein spezieller Hash-Join zur Verfügung, der keine vorherige Sortierung erfordert.

Wie bereits erwähnt, wurde zur Integration des Geoindex eine Datenbank mit ausschließlich lesendem Zugriff vorausgesetzt. Des Weiteren wurden keine Mechanismen implementiert, um zur Laufzeit weitere Geo-Objekte einfügen bzw. vorhandene Objekte löschen zu können. Im Zuge weiterer Überlegungen kann also von einem Read-Only-System ausgegangen werden.

2.5 Anfrageoptimierer

Der Anfrageoptimierer ist von zentraler Bedeutung für die Leistungsfähigkeit eines DBVS. In diesem Abschnitt sollen daher einige allgemeine Aspekte hinsichtlich der Implementierung dieser Komponente diskutiert werden. Darauf aufbaubend werden im Speziellen relevante Charakteristika des Anfrageoptimierers von GeoRDF-3X [BNM10] dargestellt, das als Grundlage dieser Diplomarbeit dient.
Abschnitt 2.5.1 beschreibt hierzu zunächst einmal die genaue Problemstellung, mit der der Anfrageoptimierer konfrontiert ist. Eine mögliche Lösung mittels Dynamischer Programmierung wird anschließend in 2.5.2 vorgestellt. 2.5.3 und 2.5.4 diskutieren das dafür notwendige Kostenmodell sowie die Erhebung von Statistiken zur Nutzung durch das Kos-

tenmodell. Abschnitt 2.5.5 beschreibt abschließend eine konkrete derartige Statistik, die in GeoRDF-3X zur Abschätzung der Kardinalität von Zwischenergebnissen verwendet wird.

2.5.1 Problemstellung

Wie in Abschnitt 2.3.1 dargestellt, erhält der Optimierer vom Front-End eine strukturierte Darstellung der Anfrage in Form eines Graphen und muss eine geeignete Kombination der verfügbaren Planoperatoren finden, um damit die von der Anfrage spezifizierten Informationen auf Grundlage der vorhandenen Datenbasis zu berechnen. Eine solche Kombination von Planoperatoren wird auch als Ausführungsplan bezeichnet. Im Allgemeinen kommt eine Vielzahl an Ausführungsplänen in Betracht und es gilt, einen möglichst kostengünstigen auszuwählen. Gemäß [NW08] ist diese Wahl für die Leistungsfähigkeit des DBVS von entscheidender Bedeutung.

Üblicherweise werden im Rahmen einer Anfrage Daten anhand der Eigenschaften von Entitäten der Miniwelt bzw. Beziehungen dieser Entitäten untereinander ausgewählt. Aufgrund der feingranularen Datenmodellierung in RDF ist hierfür gewöhnlich in beiden Fällen eine Vielzahl an Tripel-Mustern in der Anfrage notwendig. Wie in Abschnitt 2.1.2 erläutert, wird zu deren Auswertung entsprechend eine Vielzahl an Verbund-Operationen benötigt. Derartige Operationen sind allgemein bereits ausgesprochen teuer, im RDF-Kontext jedoch ganz besonders, da als Eingabemenge konzeptionell die vollständige und in der Regel sehr umfangreiche Faktentabelle herangezogen werden muss. Durch eine geeignete Architektur des Gesamtsystems kann wie in den Abschnitten 2.2 und 2.3.5 beschrieben, letztere Problematik durch gezielte Bereitstellung der *tatsächlich* benötigten Tripel sowie Sideways Information Passing entschärft werden.
Totzdem besitzt die Join-Operation nach wie vor eine inhärente Komplexität von $\mathcal{O}(n^2)$ in der Länge ihrer Eingabe und ist damit maßgeblich für die Leistungsfähigkeit des DBVS verantwortlich. Es gilt somit, die Eingabemengen der Joins möglichst klein zu halten. Da üblicherweise mehrere Joins miteinander verknüpft werden müssen, wird nicht selten die Ausgabe einer solchen Operation als Eingabe für einen nachfolgenden Join verwendet. Allerdings lässt sich die Ausgabekardinalität eines Joins nicht a priori bestimmen und das Problem der optimalen Verkettung von Verbund-Operationen ist NP-vollständig.

Diese Tatsache motiviert eine strukturierte Herangehensweise, wie sie im nachfolgenden Abschnitt beschrieben wird.

2.5.2 Problemlösung durch Dynamische Programmierung

Auf Basis der in GeoRDF-3X verfügbaren Operatoren ergeben sich oftmals mehrere Möglichkeiten zur Beantwortung einer Anfrage. So können beispielsweise Scan-Operatoren die Faktentabelle in unterschiedlichen Reihenfolgen durchlaufen. Des Weiteren stehen zur Durchführung der Verbund-Operation sowohl Merge- als auch Hash-Joins zur Verfügung (s. jeweils 2.3.4). Schließlich sind auch noch unterschiedliche Reihenfolgen der Verbund-Operationen realisierbar (s. 2.5.1). Aus dieser Vielzahl an Möglichkeiten ergibt sich bereits

für kleine Anfragen ein augesprochen umfangreicher Lösungsraum.
Begnügt man sich mit einem *möglichst guten* Ausführungsplan, so kommt als naive Methode zur Lösungsfindung ein randomisiertes, partielles Durchmustern des Lösungsraumes in Betracht. Jedoch läuft man dabei Gefahr, unter Umständen auch relativ schlechte Ausführungspläne als vermeintlich gute Lösung zu identifizieren.

GeoRDF-3x verwendet daher die alternative Strategie der dynamischen Programmierung, um alle Elemente des Lösungsraums aufzuzählen und geht dabei „bottom-up“ vor. Es interpretiert hierzu die optimale Verknüpfung von n Tripel-Mustern als (Teil-)Problem der Länge n. Begonnen wird mit Tupel-Mengen, die allein durch die von einzelnen Tripel-Mustern bestimmte Selektion auf der Faktentabelle entstehen, d.h. Teilproblemen der Länge 1. Entsprechend dem typischen Vorgehen bei dynamischer Programmierung ergeben sich die möglichen Lösungen für Teilprobleme der Länge $k+1$ aus allen in Frage kommenden Verknüpfungen, d.h. in diesem Fall Joins, der Teilprobleme mit Länge k. Die dadurch gefundenen Lösungen für Teilprobleme der Länge $k+1$ werden gemäß eines Kostenmodells bewertet und aussortiert, sofern bereits eine billigere Lösung für das selbe Teilproblem (d.h. die Verknüpfung der selben Menge von Tripel-Mustern) bekannt ist.

Der Optimierer von GeoRDF-3X weist hierbei einige Besonderheiten auf, die nachfolgend dargestellt werden sollen. Neben den von der jeweiligen Lösung bereits berücksichtigten Tripel-Mustern sowie ihren Kosten wird auch eine möglicherweise vorhandene Sortierung der Tupel (die beispielsweise durch Anwendung des sortierungserhaltenden Merge-Joins entsteht) vermerkt. Dies ist sinnvoll, da derartige Sortierungen bei der Lösung größerer Teilprobleme eventuell gewinnbringend eingesetzt werden können. Aus dem selben Grund werden auch nicht-kostenoptimale Lösungen, die eine später unter Umständen nützliche Sortierung mit sich bringen, weiterverarbeitet, sofern keine billigere Lösung mit der entsprechenden Sortierung bekannt ist.
Zur Bestimmung der Teilprobleme mit Länge 1 können in GeoRDF-3X die vorhandenen Indexe herangezogen werden (s. Abschnitt 2.3.4). Sie liefern gemäß ihrer ersten Tripel-Komponente nach dem Konstanten-Präfix (s. ebenfalls 2.3.4) die initialen, später unter Umständen verwendeten Sortierungen. Entsprechend obiger Beschreibung werden hierbei auch Indexe berücksichtigt, die eine nicht unmittelbar im Anschluss nutzbare Sortierung produzieren.

2.5.3 Kostenmodell für Joins

Wie in Abschnitt 2.5.2 bereits erwähnt, benötigt das dort beschriebene Verfahren ein Kostenmodell für die Join-Operation. Kostenmodelle in DBVS bestehen üblicherweise [SAC+79] aus einer Gewichtung der Kosten für die notwendigen Berechnungen sowie der Kosten für den Speicherzugriff. Beide Anteile sind gemäß 2.5.1 im Falle der Join-Operation maßgeblich vom Umfang der Eingabedaten des Verbunds abhängig. Es genügt also, ausschließlich die Kardinalität der Eingabedaten in das gesuchte Kostenmodell einfließen zu lassen.
Die Kosten eines konkreten Joins lassen sich damit bereits auf einer ausreichend präzisen Abstraktionsebene bestimmen, da lediglich *relative* Kosten, d.h. ein *Vergleichswert* bezüglich unterschiedlicher Ausführungspläne benötigt wird. Besagte Kardinalität ergibt sich

aus den Charakteristika der Vorgängeroperation im Ausführungsplan. Handelt es sich hierbei um einen Scan, so kann die gesuchte Größe in einfacher Weise mit Hilfe des entsprechenden aggregierten Indexes bestimmt werden. Für Filter und Joins wurden in [SAC+79] bzw. [Dem80] geeignete Abschätzungen mittels Filter- und Join-Selektivitätsfaktoren vorgeschlagen. Letztere beschreiben dabei den Anteil des kartesischen Produkts, der den Join-Bedingungen genügt und somit als Ausgabemenge weitergereicht wird. Diese Modelle sind jedoch recht einfacher Natur und berücksichtigen - zum Zwecke der Vereinfachung - nicht die beiden folgenden, grundlegenden Probleme:

- *Ungleichverteilung* der Werte einzelner Attribute
- *stochastische Abhängigkeit* der Werte verschiedener Attribute

Derartige Vereinfachungen führen jedoch zu unpräzisen Abschätzungen der Ausgabekardinalität des jeweiligen Operators. Entsprechend resultieren hieraus unpräzise Abschätzungen der Kosten für die nachfolgende Verbund-Operation. Aus diesem Grund ist es üblich, Statistiken über die tatsächliche Werteverteilung zu erheben und zur Abschätzung der Kardinalitäten zu verwenden. Derartige Statistiken sollen im nachfolgenden Abschnitt kurz vorgestellt werden.

2.5.4 Statistikerhebung

Statistiken fassen in ihren unterschiedlichen Formen stets die *tatsächlich* vorhandene Werteverteilung einer Datenbasis zusammen und stellen somit Metadaten dar. Sind sie geschickt gewählt, so kann mit ihrer Hilfe oftmals der *tatsächliche* (Join-) Selektivitätsfaktor sehr viel genauer bestimmt werden, als mit allgemeinen, wertunabhängigen Modellen.

Histogramme stellen eine beliebte Technik für derartige Statistiken dar. Dabei wird mittels einer geeigneten Gruppierung der vorkommenden Werte die tatsächliche - möglicherweise von der Gleichverteilung abweichende - Häufigkeitsverteilung der Attributwerte beschrieben. Histogramme können in zwei Klassen unterteilt werden, die sich in der Art und Weise dieser Gruppierung unterscheiden. Equi-Width-Histogramme fassen eine *konstante* Anzahl paarweise verschiedener Attributwerte zusammen und weisen jeder dieser Gruppen die Anzahl der hiervon abgedeckten, beobachteten Messwerte zu. Equi-Depth-Histogramme hingegen fassen eine *variable* Anzahl paarweise verschiedener Attributwerte derart zusammen, dass die resultierenden Gruppen alle die selbe Größe besitzen. Histogramme können also prinzipiell auch über mehrdimensionalen Attributwerten wie beispielsweise Koordinaten erstellt werden. Die Gruppen sind dann Objekte der jeweiligen Dimension.
Mit Hilfe einer approximierenden Beschreibung der tatsächlichen Werteverteilung wird das Problem der Ungleichverteilung somit durch Histogramme weitestgehend gelöst; nicht jedoch das Problem der stoachastischen Abhängigkeit.

Statistiken werden in speziellen Segmenten der Datenbank abgelegt und implizieren somit zusätzlichen Ein-/Ausgabe-Aufwand. Darüber hinaus sollten Statistikdaten in möglichst umfangreichem Maße im Datenbank-Puffer gehalten werden, so dass der Zugriff hierauf effizient möglich ist und damit der Optimierungs-Overhead minimiert werden kann. Um

vor diesem Hintergrund die Skalierbarkeit des Systems sicherzustellen, müssen Statistiken weitgehend unabhängig vom Volumen der durch sie zusammengefassten Daten eine mehr oder weniger konstante Größe aufweisen. Selbstverständlich geschieht dies auf Kosten der Präzision.
Des Weiteren ist angesichts des Umfangs der Datenbasis oftmals eine Statistik-Generierung unter Einbeziehung *sämtlicher* enthaltener Daten nicht in überschaubarer Zeit möglich. Sampling stellt einen naheliegenden Ansatz zur Lösung dieser Problematik dar. Dabei werden den Daten in geeigneter, möglichst repräsentativer Weise Stichproben entnommen und anschließende Statistikerhebungen auf diesen Stichproben anstelle der eigentlichen Daten vorgenommen.

Nachfolgend soll exemplarisch eine in GeoRDF-3X zur Verbesserung der Abschätzungen durch den Optimierer verwendete Statistik vorgestellt werden.

2.5.5 Join-Selektivitäts-Abschätzung in RDF-3X

In [NW09] wird ein Verfahren beschrieben, mit dessen Hilfe eine relativ präzise Bestimmung der Join-Selektivitäten in RDF-3X möglich ist. Hierzu werden Selektivitäts-Abschätzungen für *alle* denkbaren binären Verbund-Operationen vorberechnet und im Statistik-Segment der Datenbank materialisiert.

Grundsätzlich sind mehrere Fälle binärer Joins zu unterscheiden. Im Folgenden sollen diese anhand der Anzahl ihrer im ersten beteiligten Tripel-Muster enthaltenen Variablen gruppiert und getrennt voneinander betrachtet werden. Enthält ein solches Muster lediglich eine Variable, so kann nur diese zur Join-Bildung herangezogen werden. Um alle theoretisch möglichen Fälle abzudecken sind jedoch trotzdem noch für sämtliche in Frage kommenden Konstanten-Werte und mögliche Positionen der Join-Variablen in den Tripel-Mustern die jeweiligen Selektivitäten zu bestimmen. Seien nun c_1, c_2, c_3 und c_4 festgehaltene Konstanten sowie S_1, P_1, O_1 und S_2, P_2, O_2 Variablen. Betrachtet werde o.B.d.A. ein Join der Tripel-Muster (c_1, c_2, O_1) und (S_2, c_3, c_4) über den Variablen O_1 und S_2.

Mit Hilfe der in Abbildung 2.3 dargestellten relationenalgebraischen Umformung des Joins können die in RDF-3X ohnehin vorhandenen aggregierten Indexe, wie weiter unten beschrieben, zur Abschätzung der Join-Selektivität verwendet werden.

Auf Basis des Operatorbaums in Abbildung 2.3 c) wird zunächst die Selektivität des Joins *ohne* Berücksichtigung der Selektion an der Wurzel berechnet. Die gesuchte Selektivität der vollständigen Operation ergibt sich dann als Produkt dieses Wertes mit der Selektivität besagter Selektion. Sind die beiden Tripel-Muster (c_1, c_2, O_1) und (S_2, c_3, c_4) stochastisch unabhängig, so liefert der Ansatz einen exakten Wert für die Selektivität des ursprünglichen Joins. Sind sie jedoch stochastisch abhängig, so wird diese Abhängigkeit im beschriebenen Berechnungsverfahren nicht berücksichtigt und das erhaltene Resultat ist eine unpräzise Abschätzung.

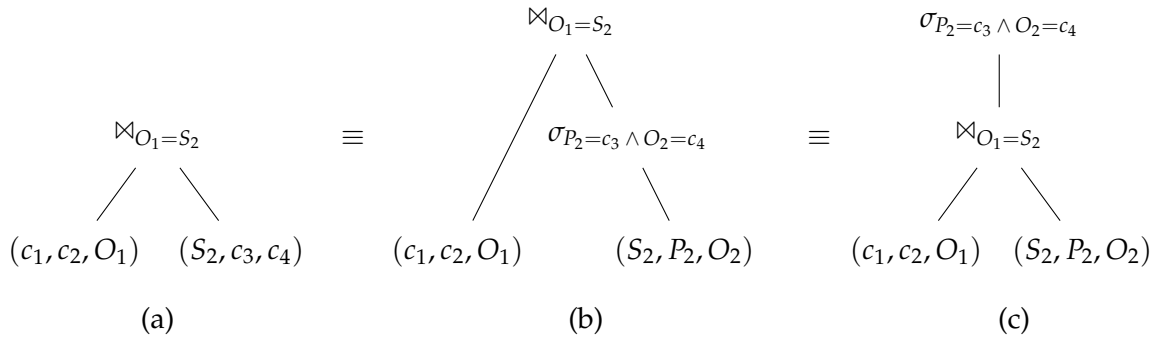


Abbildung 2.3: relationenalgebraische Umformung der Verbund-Operation, aus [NW09]

Die Bestandteile des Produkts können wie folgt bestimmt werden: Die Selektivität des Joins ohne Berücksichtigung der Selektion an der Wurzel ergibt sich gemäß Abbildung 2.3 c) zu (vgl. [NW09]):

$$\begin{aligned} sel((c_1, c_2, O_1) \bowtie_{O_1=S_2} (S_2, c_3, c_4)) &:= \frac{|(c_1, c_2, O_1) \bowtie_{O_1=S_2} (S_2, P_2, O_2)|}{|(c_1, c_2, O_1)|\,|(S_2, P_2, O_2)|} \\ &= \frac{\sum_{x \in \pi_{O_1}(c_1, c_2, O_1)} |(x, P_2, O_2)|}{|(c_1, c_2, O_1)|\,|(S_2, P_2, O_2)|} \end{aligned}$$

$\pi_{O_1}(c_1, c_2, O_1)$ sei hierbei die Projektion der durch (c_1, c_2, O_1) beschriebenen Tupelmenge auf das Objekt. Die $x \in \Pi_{O_1}(c_1, c_2, O_1)$ können mit Hilfe des Index $\mathcal{I}_{SPO}$ (vgl. 2.3.3) sehr effizient berechnet werden. Für jedes derartige x wiederum lässt sich mittels des voll aggregierten Index $\mathcal{I}_S$ ebenso effizient die Anzahl der jeweiligen Join-Partner ($|(x, P_2, O_2)|$) bestimmen. $|(c_1, c_2, O_1)|$ schließlich ergibt sich aus dem aggregierten Index $\mathcal{I}_{SP}$ und $|(S_2, P_2, O_2)|$ entspricht der Gesamtzahl aller Tupel der Faktentabelle, die sicherlich ohne größeren Aufwand mitgeführt werden kann.
Die so erhaltenen Werte werden o.B.d.A. in einem durch die Konstanten c_1 und c_2 indizierten B$^+$-Baum materialisiert. Die Selektivitäten für alle in Frage kommenden Positionen der Join-Variablen im zweiten Tripel-Muster werden dabei unmittelbar benachbart im entsprechenden Blatt des B$^+$-Baumes hinterlegt. Da für die Join-Variable im ersten Tripel-Muster grundsätzlich drei Positionen in Betracht kommen, müssen drei derartige B$^+$-Bäume angelegt werden.

Die Selektivität der Selektion an der Wurzel kann o.B.d.A. unter Verwendung des aggregierten Index $\mathcal{I}_{PO}$ zu

$$sel(\sigma_{P_2=c3 \wedge O_2=c_4}(S_2, P_2, O_2)) := \frac{|(S_2, c_3, c_4)|}{|(S_2, P_2, O_2)|}$$

bestimmt werden. Hierzu ist lediglich ein einziger B$^+$-Baum-Zugriff für die Bestimmung von $|(S_2, c_3, c_4)|$ zur Laufzeit notwendig, so dass diese Werte *nicht* vorab materialisiert werden.

Der Fall mit zwei Variablen im ersten Tripel-Muster kann analog unter Verwendung jeweils geeigneter Indexe behandelt werden. Hier ergeben sich 3 Möglichkeiten für die Wahl der Konstanten und damit auch der Variablen. Jede der jeweils zwei Variablen kann als Join-Variable verwendet werden. Entsprechend ergeben sich für diesen Fall $2 \cdot 3 = 6$ weitere B$^+$-Bäume.

Für den Fall mit drei Variablen kann analog zu oben die Selektivität des Joins ohne Berücksichtigung der Selektion an der Wurzel wie folgt berechnet werden:

$$sel((S_1, P_1, O_1) \bowtie_{O_1=S_2} (S_2, c_3, c_4)) := \frac{|(S_1, P_1, O_1) \bowtie_{O_1=S_2} (S_2, P_2, O_2)|}{|(S_1, P_1, O_1)|\,|(S_2, P_2, O_2)|}$$
$$= \frac{\sum_{x \in \pi_{O_1}(S_1, P_1, O_1)} |(S_1, P_1, x)|\,|(x, P_2, O_2)|}{|(S_1, P_1, O_1)|\,|(S_2, P_2, O_2)|}$$

Die $|(S_1, P_1, x)|$ bzw. $|(x, P_2, O_2)|$ können dabei mittels schritthaltender Scans über die voll aggregierten Indexe $\mathcal{I}_O$ und $\mathcal{I}_S$ effizient bereitgestellt, multipliziert und aufaddiert werden. Da die Tripel-Muster in diesen Fällen keine Konstanten enthalten und je 3 Möglichkeiten für die Wahl der Join-Variablen im ersten bzw. zweiten Tripel-Muster vorhanden sind, können lediglich 9 derartige Joins spezifiziert werden. Die entsprechenden Selektivitäten werden daher vorab berechnet und im Datenbank-Katalog hinterlegt.

Die (approximierte) Selektivität des ursprünglichen Joins kann also in jedem der drei Fälle sehr effizient bestimmt werden. Dies ermöglicht die bereits genannte Vorausberechnung der Selektivitäten *sämtlicher* denkbarer Verbund-Operationen. Die so berechneten Metadaten beschreiben die Werteverteilung in der Datenbasis und beanspruchen somit weit weniger Platz als die Daten selbst. Darüber hinaus werden die Seiten der B$^+$-Bäume einer geeigneten Komprimierung unterzogen, so dass die zusätzlichen Indexe mit einem vertretbaren Mehraufwand an Speicher materialisiert werden können.
Allerdings geht das Verfahren von der stochastischen Unabhängigkeit der involvierten Tripelmengen aus und liefert nur unter dieser Voraussetzung *exakte* Abschätzungen der Join-Selektivität. Damit löst eine derartige Statistik zwar das Problem der Ungleichverteilung, indem die tatsächliche Verteilung in geeigneter Weise zusammengefasst wird; sie löst jedoch nicht das Problem der stochastischen Abhängigkeit! Gleichwohl konnte in [NW09] der praktische Nutzen eines derartigen Ansatzes nachgewiesen werden.

2.5.6 Optimierung ortsbezogener Anfragen

Zur Beantwortung ortsbezogener Anfragen in GeoRDF-3X ist einer der beiden in 2.4.4 beschriebenen Operatoren zu verwenden. Bislang ist diesbezüglich lediglich eine pauschale, *manuelle* Auswahl zwischen Spatial-Selection und Spatial-Index möglich.

Wie in 2.4.4 bereits dargestellt, wird der Selektionsoperator dabei stets an der Wurzel des Operatorbaums platziert.
Da bislang keinerlei Statistiken für die resultierenden Selektivitäten von Joins mit dem Geoindex vorliegen, wird hierfür generell ein extrem geringer Wert angenommen. Dadurch liefert der in 2.5.2 beschriebene Optimierer stets Ausführungspläne, die den Geoindex gleich zu Beginn im untersten Join des Baums einbinden.
Dies ist im Allgemeinen jedoch nicht sinnvoll. Daher soll als Ziel dieser Diplomarbeit ein Mechanismus zur Abschätzung der Selektivität von Joins mit dem Geoindex inklusive der dafür notwendigen Statistiken erarbeitet werden. Dieser Mechanismus soll darüber hinaus einen Vergleich der resultierenden Kosten mit denen eines Selektionsoperators an der Wurzel ermöglichen, so dass adaptiv zwischen den beiden verfügbaren Planoperatoren ausgewählt werden kann.

3 Verwandte Arbeiten

3.1 Vorüberlegungen

Gemäß Abschnitt 1.2 besteht die Aufgabenstellung dieser Arbeit im Kern darin, Verfahren zur Abschätzungen der Kardinalität von Tupelmengen zu entwickeln, die durch Teile einer SPARQL-Anfrage mit ortsbasierter Filterung spezifiziert wurden. Um eine effiziente Suche nach hierfür in Frage kommenden Ansätzen zu ermöglichen, sollen einige Aspekte dieser Aufgabenstellung zunächst einmal näher beleuchtet werden.
Wie bereits dargestellt beziehen sich die betrachteten Kardinalitätsabschätzungen genau genommen auf Tupelmengen. Zum Zwecke einer einfacheren Darstellung sollen diese jedoch im Folgenden auch direkt mit der dazugehörigen Anfrage assoziiert werden können. Die „Kardinalität einer Anfrage“ meint somit stets die Kardinalität der durch die Anfrage spezifizierten Tupelmenge.
Des Weiteren ist der Optimierer eines Datenbankverwaltungssystems für RDF-Daten selbstverständlich primär an Abschätzungen für echte *Teilmengen* der Tripelmuster einer Anfrage interessiert, da diese zur Generierung einer optimalen Join-Reihenfolge maßgeblich sind. Eine solche Teilmenge kann jedoch stets auch als vollständige Anfrage geringeren Umfangs aufgefasst werden. Entsprechend ist es völlig ausreichend, das Problem der Kardinalitätsabschätzung für vollständige Anfragen zu betrachten.
Aufgrund der in Abschnitt 2.5.2 beschriebenen Vorgehensweise des Optimierers von RDF-3X bevorzugt dieser Abschätzungen der *Selektivität* einzelner Join-Operationen. Da diese jedoch lediglich dazu dienen, innerhalb des Optimierers Abschätzungen für die *Kardinalität* von Teilanfragen zu generieren, ist es grundsätzlich auch denkbar, direkt entsprechende Kardinalitätsabschätzungen zu berechnen und einem entsprechend modifizierten Optimierer zur weiteren Verarbeitung zur Verfügung zu stellen.
Präzise Selektivitäts- bzw. Kardinalitätsabschätzungen sind im Allgemeinen nur möglich, wenn hierfür Meta-Daten der konkreten, betrachteten Daten herangezogen werden. Im hier betrachteten Fall wird es daher notwendig sein, in irgendeiner Weise Meta-Daten über Verteilung und Anzahl der vorhandenen Geo-Literale sowie die daran angrenzenden Teile des RDF-Graphen zu erheben, da das konkrete Verhalten der Joins und damit auch die Kardinalitäten ihrer Resultatsmengen hiervon abhängen. Es sei an dieser Stelle darauf hingewiesen, dass statistische Abhängigkeiten auch zwischen verschiedenen Joins bzw. ihren Bestandteilen existieren können. Aus diesem Grund genügt es im Allgemeinen *nicht*, ausschließlich die direkt an Geo-Literale angrenzenden Kanten zu betrachten. Vielmehr müssen *datenabhängig* unter Umständen auch weitere Teile des RDF-Graphen in die Betrachtung einbezogen werden. Im Folgenden soll daher erörtert werden, welche bestehenden Konzepte zur

Auswahl, Aufbereitung und Speicherung dieser Meta-Daten möglicherweise zur Lösung der vorliegenden Aufgabenstellung herangezogen werden können.

Die im Folgenden untersuchten Ansätze zur Modellierung der zu erwartenden Korrelationen zwischen dem Ort und weiteren Attributen von Entitäten lassen sich grob in zwei Klassen einteilen. Die erstere der beiden Klassen umfasst Verfahren zur Beschreibung statistischer Abhängigkeiten mit Hilfe von Histogrammen und wird in Abschnitt 3.3 untersucht. Die zweite hingegen behandelt Verfahren zur Abschätzung der Anzahl in einem Graph enthaltener Teilgraphen, die einem festgelegten Muster gehorchen. Verfahren, die dieser Klasse angehören werden in Abschnitt 3.4 dargestellt. Zunächst jedoch sollen mögliche Kardinalitätsabschätzungen und Kostenmodelle für den Spatial Index diskutiert werden.

3.2 Kardinalitätsabschätzungen und Kostenmodelle für den Spatial Index

Im Zuge der Kardinalitätsabschätzung für die betrachteten Anfragen ist es unter anderem auch notwendig, unter Vernachlässigung möglicher weiterer Attribute die Anzahl der durch das Anfragefenster spezifizierten Objekte zu bestimmen. Adäquate Verfahren hierfür sind seit langem bekannt. Die praxistauglichsten und effizientesten davon dürften sich daher zwischenzeitlich in bestehenden Systemen mit Unterstützung ortsbasierter Daten niedergeschlagen haben. Es wurde somit versucht, entsprechende Systeme dahingehend zu untersuchen. Leider finden sich hierzu jedoch kaum Publikationen bzw. Code-Dokumentationen.

3.2.1 PostGIS

PostGIS [Posb] verwendet einen vergleichsweise simplen Ansatz, der zunächst das von Geo-Objekten aufgespannte Fenster berechnet. Anschließend können optional die Grenzen dieses Fensters um einen konstanten Anteil der Standardabweichung bzgl. den Koordination der Objekte korrigiert werden. Geo-Objekte, die dadurch außerhalb des Fensters zu liegen kommen werden im Folgenden ignoriert. Anschließend wird besagtes Fenster *gleichmäßig* in eine vorgegebene Anzahl von Kacheln zerlegt und für jede dieser Kacheln die Zahl der in ihr enthaltenen Geo-Objekte vermerkt. Zur Bestimmung der Kardinalität einer durch ein Anfragefenster selektierten Menge von Objekten, ist auf dieser Basis die Summe der Einträge aller überdeckten Kacheln zu bestimmen. Anteilig überdeckte Kacheln werden dabei auch nur entsprechend ihrem überdeckten Anteil gewertet.

3.2.2 Ortsbasierte Ingres-Erweiterung

Yan et al. beschreiben in [YCCP10] eine Erweiterung des relationalen Datenbankverwaltungssystems Ingres [Ing] um die Möglichkeit, ortsbasierte Anfragen zu stellen. Zu deren Verarbeitung ist wie in der vorliegenden Aufgabenstellung auch, eine Kardinalitätsabschätzung für (Zwischen-)Ergebnismengen notwendig. Unter Anderem ist hierzu auch eine

Kardinialitätsabschätzungen der durch das Anfragefenster qualifizierten Objekte notwendig. Die besagte Implementierung verwendet dafür einen unkonventionellen Histogramm-Ansatz, dessen Buckets die Objektverteilung für den gesamten Bereich zwischen Bucket und Koordinatenursprung und nicht nur innerhalb des Buckets selbst wiederspiegeln. Dabei wird dieser Ansatz um die Möglichkeit beliebig positionierter Fenstergrenzen erweitert, indem partiell enthaltene Buckets auch nur entsprechend partiell gewertet werden.
In seiner Grundform reflektiert ein solcher Ansatz jedoch nicht eventuelle - bzw. oftmals wahrscheinliche - Korrelationen zwischen der Position eines Objekts und seinen darüber hinausgehenden Eigenschaften. Zwar wäre eine Interpretation besagter Eigenschaften als zusätzliche Dimensionen und eine dementsprechende Erweiterung des Ansatzes auf mehrere Dimensionen denkbar, doch ergeben sich daraus Problemstellungen, die später im Zusammenhang mit Histogramm-basierten Ansätzen genauer diskutiert werden. Desweiteren treten auch noch die in [GKTD05] beschriebenen Probleme bei Betrachtung partiell enthaltener Buckets in höherdimensionalen Datensätzen zu Tage. Darüber hinaus wäre zu klären, inwieweit Buckets, die mehr als den von ihnen selbst überdeckten Bereich beschreiben, im Falle der beschriebenen Erweiterung zusätzlichen Speicherplatz benötigen, um eine gleichbleibende Genauigkeit der Beschreibung zu garantieren.

3.2.3 Hierarchischer Ansatz im Microsoft SQL Server 2008

Fang et al. beschäftigen sich in [FFN+08] mit der Indexierung zweidimensionaler Objekte im Rahmen eines relationalen Datenbankverwaltungssystems. Der betrachtete Raum wird dabei für jedes Objekt individuell mit Hilfe eines hierarchischen, adaptiv verfeinerten Gitters in Zellen zerlegt. Dem zu indexierenden Objekt werden anschließend sämtliche von ihm in diesem Gitter überdeckten Zellen zugeordnet. Mittels geschickter Codierung der Zellen-IDs werden die so ermittelten Zuordnungen zwischen Zellen und Objekten unter Berücksichtigung räumlicher Nähe in einem vorhandenen, eindimensionalen B^+-Baum abgelegt. Dieser B^+-Baum dient nachfolgend als Index für ortsbezogene Anfragen. Das im Rahmen einer solchen Anfrage spezifizierte Fenster wird ebenfalls wie oben dargestellt zerlegt. Die daraus resultierenden Zellen werden anschließend durch einen Join mit den korrespondierenden Zellen des Indexes verknüpft, um eine Obermenge der enthaltenen Objekte zu berechnen. Angesichts der hierarchischen Unterteilung sind dabei nicht ausschließlich identische Zellen, sondern auch solche zu berücksichtigen, die sich gegenseitig enthalten. Fang et al. geben hierfür geeignete Verfahren an. Für einen Teil der so bestimmten Objekte lässt sich bereits in diesem Schritt feststellen, ob sie im Anfragefenster enthalten sind. Auf die übrigen wird in einem weiteren Schritt ein rechenintensiver, aber exakter Filter-Operator angewandt.
Der Ansatz liefert also ein Verfahren zur Organisation *ausgedehnter*, räumlicher Objekte. In Ermangelung entsprechender Testdaten und um das Hauptaugenmerk auf die Behandlung von Korrelationen zwischen Positionsdaten und weiteren Attributen richten zu können, soll die Betrachtung jedoch auf Punktobjekte beschränkt bleiben. Der beschriebene Ansatz, sowie sein ebenfalls in [FFN+08] vorgestelltes Kostenmodell, sind daher zunächst nicht gewinnbringend nutzbar, dürften jedoch einen vielversprechenden Ausgangspunkt für eine mögliche zukünftige Erweiterung auf ausgedehnte Objekte darstellen.

Leider wird die Problematik korrelierter Attribute in [FFN+08] nicht behandelt, so dass hierfür keine möglicherweise geeigneten Ansätze zu entnehmen sind. Obwohl sich die beschriebene Implementierung analog zu GeoRDF-3X zwischen der Nutzung eines Index- oder Selektions-Operators entscheiden muss, wird nicht genauer dargelegt, auf welcher Basis diese Entscheidung getroffen wird. Auch in dieser Hinsicht liefert [FFN+08] daher leider keine nutzbaren Ansätze.

3.2.4 Kostenmodelle für R-Bäume

Die Kosten für das Aufsuchen, durch eine Anfrage qualifizierter Tupel in einem R-Baum bzw. seinen Variationen werden in [FSR87, KF93, PSTW93, TSS00] untersucht. Die hierbei in Folge wachsender Primärspeicherbereiche zunehmend relevanten Pufferungseffekte werden in [LL00] genauer analysiert und entsprechende Kostenmodelle vorgeschlagen. Obgleich eine derartige Kostenabschätzung im Zuge der Implementierung eines Prototyps notwendig ist, wurde diesem Forschungsbereich bei der Erstellung der vorliegenden Arbeit keine allzu große Aufmerksamkeit geschenkt, da die entsprechenden Konzepte deren Kernproblem nicht zu lösen im Stande sind und lediglich Verbesserungen um einen konstanten Faktor bewirken dürften. Zum Zwecke der Vollständigkeit seien sie an dieser Stelle aber trotzdem erwähnt.

3.3 Histogramm-basierte Ansätze zur Selektivitätsabschätzung

3.3.1 Selektivitätsabschätzung konventioneller SPARQL-Anfragen

Stocker et al. untersuchen in [SSB+08] das Problem der Selektivitätsabschätzung für SPARQL-Anfragen. Zwar beschäftigen sie sich dabei nur mit „reinen“ RDF-Daten, d.h. ohne gesondert zu behandelnde Orts-Literale, doch könnte ein adäquater Ansatz zur Behandlung statistischer Abhängigkeiten möglicherweise für den erweiterten, vorliegenden Fall angepasst werden. Als Grundlage ihrer Untersuchungen verwenden die Autoren dabei Jena ARQ und weisen dezidiert darauf hin, sich nur mit Systemen befasst zu haben, die den RDF-Graphen vollständig im Primärspeicher vorhalten. Da es sich bei der Selektivitätsabschätzung jedoch um ein Konzept-inhärentes Problem der zunächst abstrakten Anfragesprache SPARQL handelt, sollte dessen Lösung zunächst einmal von der konkreten, zu Grunde liegenden Implementierung unabhängig und damit auch im vorliegenden Fall grundsätzlich einsetzbar sein. Im Gegensatz zur Herangehensweise mittels dynamischer Programmierung in [NW08] verwendet die in [SSB+08] vorgestellte Implementierung einen Ansatz, der sukzessive den selektivsten, bislang noch nicht berücksichtigten Join auswählt, um einen „optimalen“ logischen Operatorbaum zu bestimmen. Die Schnittstelle des Optimierers zur Selektivitätsabschätzung ist aber auch hier, wie in [NW08], die Abfrage der Selektivitäten einzelner bzw. mittels Join verbundener Tripelmuster, so dass auch diesbezüglich nichts gegen eine Verwendung im vorliegenden Fall spricht.

Zur Abschätzung der Selektivität einzelner Tripelmuster betrachtet die in [SSB+08] beschriebene Implementierung zunächst die Selektivitäten der Tripel-Bestandteile Subjekt, Prädikat und Objekt. Variablen wird hierbei die Selektivität $1,0$ zugeordnet, da sie keine einschränkende Semantik besitzen. Sei $|\mathcal{S}|$ die Anzahl paarweise verschiedener Subjekte. Dann wird als Selektivität des Subjekts der Wert $sel_S = \frac{1}{|\mathcal{S}|}$ angenommen. Seien weiterhin $|\mathcal{T}|$ die Gesamtzahl vorhandener Tripel und $|p_i|$ die Anzahl der Tripel mit Prädikat p_i. Dann wird $sel_P(p_i) = \frac{|p_i|}{|\mathcal{T}|}$ als Selektivität des Prädikats p_i verwendet. Zur Bestimmung der Selektivität sel_O des Objekts werden für sämtliche Prädikate Histogramme der zugehörigen Objektwerte erstellt. Die Selektivität des Tripels als Ganzes berechnet sich aus den genannten Werten gemäß $sel_{SPO} = sel_S \cdot sel_P \cdot sel_O$. Es wird also statistische Abhängigkeit der Tripel-Bestandteile angenommen.
Zur Bestimmung der Selektivität zweier durch einen Join über den Subjekten verbundener Tripel-Muster wird für alle Kombinationen von Prädikaten p_1 und p_2 die Kardinalität $\left|J_{p_1,p_2}\right|$ der Anfrage

```
SELECT ?x ?y WHERE {
  ?s p1 ?x .
  ?s p2 ?y
}
```

bestimmt und in der Statistik hinterlegt. Die Selektivität eines Joins $(S_1, p_1, O_1) \bowtie_{S_1=S_2} (S_2, p_2, O_2)$ mit den Variablen S_1, O_1, S_2, O_2 wird damit zu $sel((S_1, p_1, O_1) \bowtie_{S_1=S_2} (S_2, p_2, O_2)) = \frac{\left|J_{p_1,p_2}\right|}{\mathcal{T}^2}$ berechnet. Treten anstelle der Variablen O_1 bzw. O_2 Konstanten auf, so kommen die jeweiligen Objekt-Selektivitäten sel_O als multiplikative Faktoren hinzu. Auch hier wird also statistische Unabhängigkeit angenommen. Analog werden Statistiken für sämtliche weiteren Kombinationen von Join-Variablen ($S_1, O2$, $S_2, O1$, $O_1, O2$) bestimmt und verwendet. Ist eine Ontologie der Daten vorhanden, so kann die Betrachtung von Prädikat-Kombinationen auf die gemäß der Ontologie tatsächlich auftretenden Kombinationen beschränkt und damit unter Umständen drastisch reduziert werden. Im vorliegenen Fall kann eine solche Ontologie jedoch nicht vorausgesetzt werden!
Mit Hilfe der bereits bestehenden aggregierten und voll-aggregierten Index kann RDF-3X für die einzelnen Tripel-Bestandteile Schätzungen angeben, die mindestens si genau wie die oben genannten sind. Für Kombinationen sind mit Hilfe besagter Indexe sogar genauere Schätzungen möglich, da statistische Abhängigkeiten innerhalb des Tripels berücksichtigt werden. Desweiteren sind damit Abschätzungen für *alle* denkbaren Kombinationen von Subjekt, Prädikat und Objekt möglich. Statistische Abhängigkeiten zwischen Verbundpartnern werden durch obige Abschätzungen ebenso wenig berücksichtigt wie durch das in [NW09] beschriebene Verfahren. Selbiges gilt für Abhängigkeiten zwischen verschiedenen Join-Operationen, da lediglich gebundene Werte des direkten Join-Partners, nicht aber die seiner weiteren Partner in die Abschätzung einbezogen werden. Lediglich hinsichtlich der selektiven Wirkung durch *Vorhandensein* oder eben *Nicht-Vorhandensein* von Prädikatkombinationen zu einem Subjekt ermöglichen obige Formeln präzisere Aussagen als dies bislang in RDF-3X möglich ist. Dem gegenüber stehen jedoch die beträchtlichen Kosten zur Berechnung sämtlicher Prädikatkombinationen, die für reale Datensätze mit oftmals mehreren

Hundert Prädikaten angesichts des geringen Nutzens kaum zu rechtfertigen sind. [SSB+08] liefert also keine adäquate Grundlage zur Lösung der vorliegenden Aufgabenstellung, so dass über den RDF-Kontext hinaus nach Konzepten zur Behandlung statistischer Abhängigkeiten gesucht werden sollte.

3.3.2 Verschiedene Histogramm-Ansätze

Eindimensionale Histogramme (s. Abschnitt 2.5.4) stellen ein bewährtes Verfahren zur Beschreibung von Werteverteilungen in Datenbankverwaltungssystemen dar. Ihre mehrdimensionalen Erweiterungen sind daher in der Lage, statistische Abhängigkeiten zwischen mehreren Attributen zu erfassen, wie dies zur Lösung der vorliegenden Aufgabenstellung notwendig sein wird. Allerdings existieren im mehrdimensionalen Fall eine Vielzahl an Möglichkeiten zur Aufteilung des Datenraumes auf die Buckets eines Histogramms. Poosala und Ioannidis vergleichen daher in [PI97] mehrere Verfahren bzw. Heuristiken zur Wahl einer solchen Aufteilung.
Das erste dieser Verfahren zählt die vorhandenen Datenpunkte entlang einer Hilbert-Nummerierung [Jag90] auf, und unterteilt die entstehende lineare Ordnung gemäß einem der in [PHIS96] definierten Partitionierungskriterien.
Ein weiteres Verfahren ist der von den Autoren entwickelte Ansatz PHASED. Dabei wird der Datenraum im ersten Schritt entlang einer zuvor gewählten Dimension aufgeteilt. In jedem weiteren Schritt werden die zuletzt erzeugten Bereich entlang einer gemeinsamen Dimension weiter verfeinert. Angesichts der a priori gewählten Dimensionen zur Aufteilung in den verschiedenen Schritten ist das Verfahren jedoch nicht sonderlich flexibel und arbeitet im weitesten Sinne datenunabhängig.
Poosala und Ioannidis schlagen daher eine Verbesserung vor und nennen diese MHIST-p. Dabei wird im Zuge der Behandlung zuvor erzeugter Bereiche stets *datenabhängig* dasjenige Paar aus Bereich und Dimension ausgewählt, für das die Notwendigkeit einer Unterteilung den größten Wert annimmt. Das Maß hierfür ist abhängig vom jeweils verwendeten Histogramm-Typ gemäß der Taxonomie in [PHIS96]. Anschließend wird eine Unterteilung in p Teilbereiche vorgenommen, wobei der jeweils für p verwendete Wert Teil der Verfahrensbezeichnung ist.
Als zusätzlicher Vergleichspartner dient ein Verfahren, das einen gänzlich anderen Ansatz verfolgt. Hierzu wird die Häufigkeitsverteilung zweidimensionaler Wertekombinationen als Matrix J interpretiert und diese gemäß $J = UDV^T$ faktorisiert. D sei hierbei eine Diagonalmatrix. Es sind Faktorisierungen möglich, so dass die Einträge der Diagonalen von D positiv und absteigend sortiert sind. Seien d_i der i-te Eintrag dieser Diagonalen und $R_M(i)$ bzw. $C_M(i)$ Vektoren, die der i-ten Zeile bzw. Spalte einer Matrix M entsprechen. Dann lässt sich J auch darstellen als $J = \sum_{k=1}^{N} d_k C_U(k) R_V(k)$. Die Matrix J kann also approximiert werden, indem lediglich die ersten $n < N$ Summanden, d.h. die ersten d_i, $C_U(k)$ und $R_V(k)$ betrachtet und als Teil der Statistik hinterlegt werden. Die Vektoren $C_U(k)$ und $R_V(k)$ können hierbei in Form gewöhnlicher, eindimensionaler Histogramme dargestellt werden. Ein immenser Nachteil dieses Verfahrens besteht jedoch in der Beschränkung auf maximal zwei Dimensionen, so dass es zur Lösung der vorliegenden Aufgabenstellung nicht in Frage kommt.

In einer ausführlichen Evaluation der vorgestellten Verfahren erweist sich der MHIST-Ansatz mit $p = 2$ in vielerlei Hinsicht als beste Methode zur Beschreibung statistisch abhängiger, mehrdimensionaler Datensätze.
Leider löst er aber, wie sämtliche anderen in [PI97] vorgestellten Ansätze ebenfalls, nicht das folgende grundsätzliche Problem in Bezug auf eine mögliche Anwendung im RDF-Kontext: Wie zu Beginn des Kapitels bereits erwähnt, müssen zum Zwecke der Selektivitätsabschätzung Meta-Daten über die vorhandenen Geo-Literale samt der an sie angrenzenden Teile des RDF-Graphen erhoben und in adäquater Weise hinterlegt werden. Eine geeignet gewählte Teilmenge der Subjekte, Prädikate und Objekte der entsprechenden RDF-Tripel sowie die Geo-Variable dienen dabei als Dimensionen der erhobenen Daten. Da RDF für die Modellierung semistrukturierter Daten konzipiert ist, gehorchen derartige Datensätze im Allgemeinen jedoch keinem festen Schema. Entsprechend kann über Struktur und Größe der an Geo-Literale angrenzenden Teile des RDF-Graphen keine allgemeingültige Aussage getroffen werden. In der Folge ist es auch nicht möglich, eine allgemeingültige Menge von Subjekten, Prädikaten und Objekten als Dimensionen auszuwählen. Sämtliche der in [PI97] betrachteten Ansätze gehen jedoch von einer festen Menge an Dimensionen der zu Grunde liegenden Daten aus. Eine theoretisch mögliche Lösung dieser „Fehlanpassung“ bestünde darin, die Vereinigungsmenge aller als Dimensionen in Frage kommenden Tripelbestandteile zu wählen. Da gemäß [GKTD05] für quasi alle Verfahren der Speicher- und Zeitbedarf mit wachsender Zahl an Dimensionen exponentiell zunimmt, während die Genauigkeit der Abschätzung abnimmt, dürfte eine derartige Lösung mit den zur Verfügung stehenden Konzepten jedoch selbst für Datensätze moderater Größe nicht mehr effizient zu beherrschen sein.

3.3.3 Selektivitätsabschätzung für reellwertige Attribute

Gunopulos et al. untersuchen in [GKTD05] mehrere Verfahren zur Selektivitätsabschätzung von Anfragen über mehrdimensionalen, statistisch abhängigen Datensätzen. Ein besonderes Augenmerk wird dabei auf die Möglichkeit gelegt, Fest- oder Gleitkommawerte für die Attribute des Datensatzes verwenden zu können, was auch für die Repräsentation von Geo-Koordination notwendig ist. Dabei werden neben Random Sampling, Histogramm-Ansätzen wie MHIST-2 [PI97] (vgl. oben) und Min-Skew [APR99], der auf die Verarbeitung von Ortsdaten hin ausgelegt ist, auch die Wavelet-Transformation [VWI98] als mögliche Kandidaten betrachtet. Ergänzt werden diese Ansätze durch zwei im Rahmen der Arbeit neu entwickelte Methoden, die im Folgenden kurz dargestellt werden sollen.

Die erste dieser beiden Methoden versucht, Histogramm-Ansätze zu verbessern, indem auch überlappende Buckets zugelassen werden. Klassische Histogramm-Ansätze können mit ihren b paarweise disjunkten Buckets lediglich eine Unterteilung des Datensatzes in b Bereiche unterschiedlicher Objektdichte vornehmen. Eine detailliertere Unterteilung erfordert zwangsweise eine Reduktion der Bucket-Größe. Eine derartige Reduktion um den Faktor $\frac{1}{\alpha}$ erfordert jedoch bei n-dimensionalen Datensätzen ein Wachstum von b um den Faktor α^n, was für entsprechende Werte von n beträchtlich werden kann! Außerdem wächst mit einer entsprechenden Vergrößerung von b die Zahl der durch eine Anfrage unter Umständen

partiell beinhalteten Buckets exponentiell. Da für derartige Buckets nur der tatsächlich überdeckte Anteil gewertet und hierzu Gleichverteilung innerhalb des Buckets angenommen wird, wächst damit auch der Abschätzungsfehler, der sich gemäß [GKTD05] additiv aufschaukeln kann. Der in [GKTD05] vorgestellte Ansatz erlaubt daher überlappende Buckets, wobei sich die Abschätzung für mehrfach überdeckte Regionen als Summe aller dort vorhandenen Überdeckungen ergibt. Auf diese Weise werden mit b Buckets mehr als b Bereiche unterschiedlicher Objektdichte möglich. Die Berechnung einer *optimalen* derartigen Aufteilung ist überaus aufwendig, so dass in [GKTD05] ein heuristischer Linearzeit-Algorithmus angegeben wird, der versucht, eine *möglichst gute* Aufteilung zu erzielen. Hierzu wird der Datenbereich mit Hilfe eines, zu Beginn verhältnismäßig feinen, gleichmäßigen Gitters unterteilt und für sämtliche Gitterzellen deren Objektdichte berechnet. Liegt diese für eine Gitterzelle über dem Durchschnitt ihrer Nachbarzellen, so werden aus ihr zufällig gewählte Objekte entnommen, bis sich die Objektdichte den Nachbarzellen angeglichen hat. Die entommenen Objekte werden einem neuen Bucket zugeordnet und dieses in die Statistik eingefügt. Durch die Entnahme von Objekten aus überdurchschnittlich belegten Zellen tendiert die Objektverteilung anschließend mehr in Richtung Gleichverteilung, so dass der Vorgang mit einem gröberen Gitter wiederholt werden kann, bis sämtliche Objekte einem Bucket zugeordnet sind. Die derart generierten Buckets können überlappen und bilden in ihrer Gesamtheit das zu erzeugende Histogramm. Bei jeder Änderung des Gitters ist ein erneuter Scan über die Daten notwendig. Da die Anzahl dieser Änderungen quasi konstant ist, arbeitet der Algorithmus in Linearzeit bezogen auf die Anzahl zu verwaltender Objekte.

Der zweite im Rahmen von [GKTD05] vorgestellte Ansatz ist eine Erweiterung des „kernel density estimation“-Verfahrens auf mehrere Dimensionen. Es handelt sich dabei um eine verallgemeinerte Form des Random Sampling, bei der die „Energie“ gefundener Samples mittels geeigneter, mehrdimensionaler Verteilungsfunktionen auf die Bereiche rund um das Sample verteilt wird. Im Gegensatz zu ersterem Ansatz kann eine derartige Statistik mit einem einzigen Scan über den Datensatz generiert werden.
Leider löst jedoch auch keines der hier vorgestellten Verfahren die im Zusammenhang mit [PI97] beschriebene grundsätzliche Problematik der Dimensionsauswahl für RDF-Datensätze. Somit kommen auch sie nicht zur Lösung der Aufgabenstellung in Betracht.

3.3.4 Erweiterung des Histogramm-Ansatzes für höherdimensionale Daten

Gemäß [PI97, GKTD05] liefern die Annahme statistischer Unabhängigkeit und darauf aufbauende Verfahren keine hinreichend präzisen Abschätzungen für die Selektivität mehrdimensionaler Bereichsanfragen. Andererseits führen aber auch Verfahren zur Beschreibung *sämtlicher* statistischen Abhängigkeiten eines mehrdimensionalen Datensatzes gemäß [GKTD05] bereits für eine höhere einstellige Zahl an Dimensionen zu deutlichen Abschätzungsfehlern, wenngleich sie für 3,4, oder auch 5 Dimensionen noch brauchbare Abschätzungen liefern. Deshpande et al. verfolgen daher in [DGR01] einen Ansatz, der sich gewissermaßen zwischen diesen beiden Extremen bewegt. Hierzu werden mit Hilfe statistischer

Interaktionsmodelle [BFH+75, DLS80] zunächst die *maßgeblichen* statistischen Abhängigkeiten innerhalb eines Datensatzes identifiziert und diese anschließend mittels vorhandener Ansätze für mehrdimensionale Histogramme, wie beispielsweise MHIST-p, beschrieben. Die verwendeten statistischen Interaktionsmodelle zerlegen Funktionen $f(i_1, \ldots, i_n)$, die die Häufigkeit von Wertekombinationen $(i_1, \ldots, i_n)$ angeben, dabei gemäß

$$\log f(i_1, \ldots, i_n) = u + \sum_j u_j(i_j) + \sum_{j \neq k} u_{j,k}(i_j, i_k) + \cdots + u_{1,\ldots,n}(i_1, \ldots, i_n)$$

in Summanden u_{Index}, die statistische Abhängigkeiten zwischen den in ihrem *Index* genannten Dimensionen reflektieren. Vernachlässigt man dabei weniger relevante Bestandteile höherer Ordnung, d.h. Summanden u_{Index} mit umfangreichem Index, die jedoch nur geringfügig zum Wert der Summe beitragen, so erhält man eine gemäß [DGR01] ausreichend gute Abschätzung der Häufigkeiten bei gleichzeitig reduziertem Berechnungsaufwand. Dementsprechend werden in [DGR01], ausgehend von statistischer Unabhängigkeit, d.h $\log f(i_1, \ldots, i_n) = u$, so lange sukzessive Summanden geringer Ordnung hinzugenommen, die gemäß einer Fehlermetrik maßgeblich zur Genauigkeit der Abschätzung beitragen, bis eine zuvor festgelegte Komplexität der Darstellung erreicht ist. Auf diese Weise werden statistische Abhängigkeiten innerhalb *niedrigdimensionaler* Projektionen des Datenraumes bestimmt, die mit Hilfe mehrdimensionaler Histogramme (s. [PI97, GKTD05]) effizient und effektiv dargestellt werden können. Durch Auswertung der einzelnen Histogramme für entsprechende Werte und deren Kombination gemäß obiger Formel kann damit eine Abschätzung für die Kardinalität (und somit auch Selektivität) mehrdimensionaler Bereichsanfragen bestimmt werden. Deshpande et al. geben - mitunter ausgefeilte - Lösungen für Teilprobleme an, die sich im Zuge einer möglichst effizienten Umsetzung des Ansatzes ergeben. Deren genauere Behandlung würde jedoch den Rahmen dieser Darstellung sprengen. Mit Hilfe eines direkten Vergleiches wird in [DGR01] nachgewiesen, dass der beschriebene Ansatz dem Verfahren MHIST-2 überlegen ist, das zuvor als „state-of-the-art“ galt. Allerdings vermag auch er nicht die bereits dargelegten, inhärenten Probleme von Histogramm-Ansätzen bei der Beschreibung von RDF-Daten zu lösen. Zwar wird ein Verfahren zur effizienten Auswertung von Anfragen angegeben, die in ihrem Auswahlprädikat nur eine kleine Zahl der vorhandenen Dimensionen spezifizieren, doch ist zur Generierung der Statistik stets der vollständige Datenraum mit allen Dimensionen vorzuhalten, da er als Vergleichsbasis der erwähnten Fehlermetrik dient. Bei Anwendung der weiter oben beschriebenen „Lösung“ mittels Vereinigungsmenge aller in Frage kommender Dimensionen, würde dies jedoch einen nicht akzeptablen Speicher- und Berechnungsaufwand bedeuten.

Offenbar ist keiner der hier vorgestellten, Histogramm-basierten Ansätze in der Lage, die vorliegende Aufgabenstellung in adäquater Weise zu lösen. Wie ersichtlich wurde herrscht stets eine Art „Fehlanpassung“ zwischen der im RDF-Kontext gegebenen Situation und der von Histogramm-Ansätzen angenommenen Struktur der Daten. Es ist also mit einiger Sicherheit davon auszugehen, dass Histogramme im Allgemeinen kein geeignetes Hilfsmittel zur Lösung der Aufgabe darstellen und somit nach zweckmäßigeren Strukturen Ausschau gehalten werden sollte. Besagte „Fehlanpassung“ basiert im Kern auf der Diskrepanz zwischen einer überaus flexiblen Modellierung in RDF, die insbesondere eine beliebige Zahl inzidenter Kanten eines Knotens gestattet, und der vergleichsweise starren Betrachtung ei-

ner festen Menge von Dimensionen in Histogramm-basierten Ansätzen. Gesucht sind also Strukturen, die ihrerseits eine gewisse Flexibilität bieten. Hierzu gehören insbesondere Graphen. Da RDF selbst eine Graphinterpretation gestattet, scheinen Graph-basierte Ansätze sogar in doppelter Weise adäquate Verfahren zu sein, und sollen daher im Folgenden genauer untersucht werden.

3.4 Graph-basierte Ansätze zur Selektivitätsabschätzung

3.4.1 Subgraph-Counting für einfache Teilgraphen

Bordino et al. beschreiben in [BDGL08] ein Verfahren zur Klassifizierung von Graphen anhand der relativen Häufigkeit in ihnen enthaltener Teilgraphen. Da hierfür zunächst die absolute Häufigkeit von Teilgraphen, d.h. deren Kardinalität, bestimmt werden muss, könnten Teile des Vorgehens eventuell zur Lösung der vorliegenden Aufgabenstellung verwendet werden. Das Verfahren erwartet als Eingabe einen Datenstrom aller Kanten des Graphen, wobei sämtliche zu einem festgehaltenen Knoten inzidenten Kanten unmittelbar benachbart in diesem Datenstrom auftauchen. Da mittels eines vergleichsweise billigen Scans über die Blätter der Indexstrukturen von RDF-3X ein solcher Datenstrom erzeugt werden kann, qualifiziert sich der Ansatz auch in dieser Hinsicht als grundsätzlicher Kandidat zur Lösung des hier untersuchten Problems und erfordert somit eine genauere Betrachtung.
Das Vorgehen gliedert sich grob in drei Phasen, in denen die Häufigkeit eines einzigen, festgehaltenen Teilgraphen bestimmt wird. In Phase 1 wird anhand unmittelbar aus dem genannten Datenstrom ableitbarer Informationen (benachbarte Knoten, Ausgangsgrad eines Knotens, etc.) die Menge P aller *potentiellen* Vorkommen des Teilgraphen bestimmt, die bereits sämtliche seiner Knoten, nicht aber zwangsläufig alle seiner Kanten beinhalten. Anschließend wird aus P „gleichverteilt“ ein mögliches Vorkommen ausgewählt und hinsichtlich der noch fehlenden Kanten überprüft. Sind diese vorhanden, so sei $\beta = 1$, andernfalls $\beta = 0$. Wird das beschriebene Vorgehen mehrfach wiederholt, so lässt sich die absolute Häufigkeit des gesuchten Teilgraphen aus dem Erwartungswert $E(\beta)$ ableiten.

Zwar ist das Verfahren hochgradig parallelisierbar, könnte - wie erwähnt - die vorhandenen Indexstrukturen effizient nutzen und benötigt bei geeigneter Implementierung sowie $|E|$ Kanten lediglich $\mathcal{O}(|E|)$ Zeitschritte. Da aber kein persistenter Index aufgebaut wird muss es für jede Anfrage erneut vollständig durchlaufen werden! Da P wie beschrieben ausschließlich anhand des Eingabestroms bestimmt wird ist das Verfahren außerdem auf die Erkennung sehr kleiner Teilgraphen mit maximal vier Knoten beschränkt (weicht man vom strikten Modell des Eingabestroms ab, so sind zwar prinzipiell auch Teilgraphen mit mehr als vier Knoten denkbar, allerdings wird dadurch lediglich das zu lösende Problem in die Phase 1 des Verfahrens transferiert). Darüber hinaus dürfte die experimentell bestimmte und mitunter erstaunlich hohe Genauigkeit der Abschätzung massiv auf der Verwendung sehr kleiner Teilgraphen beruhen. Diese kommen naturgemäß häufiger vor als komplexere Gebilde, so dass für das stochastische Verfahren das Gesetz der großen Zahlen zum Tragen kommt. Desweiteren ist ein Großteil aller prinzipiell möglichen Teilgraphen mit maximal

vier Knoten tendenziell eng vermascht, was eher weniger dem für eine RDF-Modellierung realer Daten typischen Szenario entspricht.
Bereits einzelne dieser Kritikpunkte stehen einer Verwendung des Verfahrens im betrachteten Kontext entgegen. In ihrer Gesamtheit schließen sie eine solche Verwendung jedoch gänzlich aus.

3.4.2 MD-Tree und P-Tree

Maduko et al. beschreiben in [MASS08] zwei Verfahren zur Indexierung von Teilgraphen eines größeren Graphen. Als Teil der Einträge wird dabei die Kardinalität des jeweiligen Teilgraphen hinterlegt, so dass sich die Verfahren grundsätzlich auch *direkt* zur Lösung der vorliegenden Aufgabenstellung eignen. Beide Verfahren beruhen dabei auf einer in [YH02] vorgestellten Sequentialisierung der Teilgraphen, die wie folgt berechnet wird: Seien u, v Knoten mit den Beschriftungen $\sigma(u)$ bzw. $\sigma(v)$ und $e = (u, v)$ eine Kante, die diese Knoten verbindet. Die Beschriftung von e werde mit $\tau(e)$ bezeichnet. Seien darüber hinaus $t(u)$ bzw. $t(v)$ die Zeitpunkte, zu denen eine an einem beliebigen, aber festen, Knoten gestartete Tiefensuche die Knoten u bzw. v erstmalig besucht. Dann wird die Kante $e = (u, b)$ im Rahmen der Sequentialisierung durch das Tripel $(t(u), t(v), \sigma(u), \tau(e), \sigma(v))$ beschrieben. Die Elemente $t(u)$ bzw. $t(v)$ sind hierbei notwendig, da von einem allgemeinen Graphmodell ausgegangen wird, das für unterschiedliche Knoten identische Beschriftungen erlaubt. Eine mögliche Sequentialisierung des gesamten Teilgraphen ergibt sich daraus als Sequenz von Kantenbeschreibungen gemäß der Reihenfolge, in der diese bei der zu Grunde liegenden Tiefensuche aufgesucht wurden. Für eine Kante (u, v), die Teil des Tiefensuchbaumes ist, werden dabei sämtliche Kanten (v, w) unmittelbar im Anschluss an (u, v) in der Sequenz aufgezählt. Eine Kante (v, x) steht hierbei vor einer Kante (v, y), falls $t(x) < t(y)$. Die Reihenfolge der Kanten wird also maßgeblich durch die Tiefensuche bestimmt, wobei unterschiedliche Tiefensuchen möglich sind. Als kanonische Sequentialisierung wird daher die lexikographisch kleinste derartige Sequenz S von Kantenbeschreibungen gewählt.

Das erste in [MASS08] vorgestellte Verfahren basiert maßgeblich auf $m \times n$-Matrizen M, deren Einträge m_{ij} die Wahrscheinlichkeit eines Auftretens der Kantenklasse i an der Position j einer Sequentialisierung der indexierten Teilgraphen angeben. Die auftretenden Kanten $(t(u), t(v), \sigma(u), \sigma(e), \sigma(v))$ werden dazu anhand $\sigma(u)$, $\sigma(e)$, $\sigma(v))$ sowie ihrer aus den Werten $t(u)$ und $t(v)$ hervorgehenden Richtung klassifiziert (Kanten mit $t(u) < t(v)$ werden hierbei als Vorwärtskanten bezeichnet, solche mit $t(u) > t(v)$ als Rückwärtskanten und solche mit $t(u) = t(v)$ als Schleifen). Geht man von statistischer Unabhängigkeit der verschiedenen Positionen innerhalb einer Sequentialisierung aus, so lassen sich die zu bestimmenden Kardinalitäten durch Multiplikation der passenden Einträge einer solchen Tabelle ermitteln. Im Falle statistischer Abhängigkeit liefert ein solches Vorgehen jedoch fehlerhafte Werte. Das vorgestellte Verfahren versucht daher, statistische Abhängigkeiten zu identifizieren und deren Auswirkungen mittels *bedingter* Wahrscheinlichkeiten in einer adäquaten Form zu berücksichtigen.
Es partitioniert hierfür zunächst die auftretenden Teilgraphen anhand ihrer Größe und legt für jede derartige Partition eine Matrix wie zuvor beschrieben an. Diese Matrizen werden

im nächsten Schritt mit den Blättern eines trivialen Baumes assoziiert, der lediglich aus Blättern und einer Wurzel besteht. Die Wurzel ist hierbei mit den Blättern durch Kanten verbunden, deren Beschriftung als Kardinalität der mit dem jeweiligen Blatt assoziierten Partition gewählt wird. Die einzelnen Partitionen, d.h. Knoten, werden nun wie folgt getrennt voneinander behandelt: Ist in einem Knoten die Wahrscheinlichkeitsverteilung der Kantenklassen für eine bestimmte Position l in den Sequentialisierungen maßgeblich von einer anderen Position k abhängig, so wird für jede Kantenklasse ein neuer Knoten erzeugt. Die Matrix des ursprünglichen Knotens wird gelöscht und durch Kanten zu den neu erzeugten Kindern ersetzt. Die Wahrscheinlichkeiten, mit denen die jeweiligen Kantenklassen an Position k auftauchen dienen dabei als Kantenbeschriftungen. Desweiteren wird der Wert k im ursprünglichen Knoten vermerkt, um später feststellen zu können für welche Position eine Aufteilung erfolgte. Die neu erzeugten Kinder werden nun ihrerseits mit geeigneten Matrizen assoziiert, die die Wahrscheinlichkeiten des Auftretens der Kantenklassen an den verbleibenden Positionen *unter der Voraussetzung* angeben, dass an Position k eine Kante der jeweiligen Klasse auftritt. Die erzeugten Kinder bzw. deren Matrizen werden anschließend sukzessive in gleicher Weise aufgespalten.

Zur Klärung der Frage, welche Wahrscheinlichkeitsverteilungen *maßgeblich* von einer bestimmten Position abhängen und somit eine Aufteilung nach sich ziehen sollten, geben Maduko et al. ein Verfahren an, das adäquat erscheint, dessen detaillierte Schilderung jedoch den Rahmen dieser Darstellung sprengen würde. Im Wesentlichen identifiziert dieses Verfahren *maximal* abhängige Positionen. Der beschriebene Ansatz wird daher auch als Maximal-Dependency-Tree, kurz MD-Tree, bezeichnet.
Die Anzahl der Vorkommen eines Teilgraphen kann nun anhand seiner kanonischen Sequentialisierung wie folgt ermittelt werden. Zunächst wird in einer externen Struktur (die Autoren äußern sich nicht dazu, wie eine derartige Struktur aussehen könnte bzw. sollte) nachgeschlagen, ob der jeweilige Teilgraph überhaupt im betrachteten Graph vorhanden war. Falls ja, ist wie folgt der zur Sequentialisierung gehörende Pfad durch den Baum zu betrachten: Ausgehend von der Wurzel wird zunächst anhand der Größe des Teilgraphen in die entsprechende Partition verzweigt. Ist der jeweilige Knoten nicht mit einer Matrix assoziiert, so wird anhand der vermerkten Aufteilungsposition die Klasse der zugehörigen Kante in der betrachteten Sequentialisierung bestimmt und entsprechend weiterverzweigt. Dieser Vorgang wird so lange wiederholt, bis ein Blatt und damit eine Matrix erreicht ist. Die Anzahl der Vorkommen eines Teilgraphen ergibt sich dann als Produkt der Kantenbeschriftungen des Pfades und der Matrixeinträge für die bislang nicht betrachteten Positionen der Sequentialisierung.

Das zweite von Maduko et al. vorgeschlagene Verfahren ist konzeptionell sehr viel einfacher. Es besteht im Kern darin, die Sequentialisierungen, d.h. Folgen von Kantenbeschreibungen, in einem Präfixbaum zu verwalten. Dessen Knoten repräsentieren somit Teilgraphen des untersuchten Graphen, deren jeweilige Häufigkeit am entsprechenden Knoten vermerkt werden kann. Um den Speicherbedarf des Verfahrens zu reduzieren werden anschließend weniger relevante Teile dieses Baumes abgeschnitten. Treten für einen Knoten v des Baumes die durch seine Kinder repräsentierten Teilgraphen gemäß einem geeigneten Maß μ annähernd gleichverteilt auf, so können diese zusammengefasst werden. Bei einer späteren Abschätzung wird an dieser Position dann Gleichverteilung zugrunde gelegt. Entsprechend

können die zugehörigen Baumkanten entfernt und etwaige Kinder der Kinder direkt an v angehängt werden. Die Autoren geben ein solches Maß μ an, mit dessen Hilfe der Speicherbedarf des Verfahrens auf einen vorgegebenen Wert reduziert werden kann. Selbstverständlich leidet unter einer derartigen Reduktion die Genauigkeit der Abschätzung. Mittels einer geeigneten Heuristik wird daher versucht, diesen Fehler möglichst minimal zu halten. Eine Sequentialisierung wird in [MASS08] als Pattern bezeichnet und der geschilderte Ansatz somit konsequenterweise als Pattern-Tree, kurz P-Tree.
Die Häufigkeit des Vorkommens eines Teilgraphen im betrachteten Graph lässt sich mit Hilfe eines solchen P-Trees elegant ermitteln: Zunächst wird, wie im Falle der MD-Trees auch, anhand einer externen Struktur überprüft, ob der jeweilige Teilgraph überhaupt vorkommt und somit indexiert wurde. Ist dies der Fall, so wird, ausgehend von der Wurzel, sukzessive die zum nächsten Element der Sequentialisierung gehörende Baumkante verfolgt. Knoten, deren Kinder beim Aufbau des P-Trees zusammengefasst wurden, tragen eine geeignete Markierung. An derartigen Knoten wird das momentan betrachtete Element der Sequentialisierung abgearbeitet, ohne gleichzeitig im Baum voranzuschreiten. Wird auf diese Weise für die *vollständige* Sequentialisierung ein korrespondierender Pfad durch den Baum gefunden, so ist die gesuchte Kardinalität am letzten Knoten des Pfades vermerkt. Wird ein Blatt erreicht, ohne dadurch bereits die vollständige Sequentialisierung abgearbeitet zu haben, so ist wie folgt vorzugehen: Der am jeweiligen Blatt vermerkte Wert dient als Grundlage der Abschätzung und wird unter Annahme von Gleichverteilung für jede noch ausstehende Verzweigung anteilig reduziert.

Leider bieten beide Verfahren keine Mechanismen zur adäquaten Bearbeitung *allgemeiner* SPARQL-Anfragen. Verwendet man die Knotenbeschriftungen des RDF-Graphen für die Abbildung σ, so ist zur Kardinalitätsabschätzung von Teilmustern, die Variablen enthalten, unter Umständen eine Verzweigung des Suchpfades notwendig. Die Zahl der zu verfolgenden Pfade wächst dabei - für eine unter Umständen beträchtliche Basis - exponentiell mit der Anzahl enthaltener Variablen! Ignoriert man hingegen die Knotenbeschriftungen des RDF-Graphen und verwendet stattdessen einen konstanten Wert, so lassen sich für Muster mit variablen Subjekten bzw. Objekten effizient die zugehörigen Kardinalitäten bestimmen. Im Gegenzug ist es dann jedoch nicht mehr möglich, für gebundene Subjekte bzw. Objekte entsprechend präzisere Abschätzungen zu tätigen. Beides erscheint im Allgemeinen nicht akzeptabel, so dass die in [MASS08] vorgestellten Verfahren ebenfalls keine adäquate Lösung der untersuchten Problematik darstellen.

3.4.3 G-Trie

Ribeiro und Silva beschreiben in [RS10] ein Verfahren zur Detektion häufig auftretender Teilgraphen eines größeren Graphen. Hierfür wird zunächst eine Indexstruktur namens G-Trie aufgebaut, mit deren Hilfe anschließend gewissermaßen parallel die Vorkommen der enthaltenen Teilgraphen gezählt werden können, ohne dabei nicht auftretende Teilgraphen betrachten, oder isomorphe Graphen mehrfach im Index vorhalten zu müssen. Die Struktur der G-Tries erinnert dabei stark an die in [MASS08] vorgestellten P-Trees und könnte möglicherweise analog zur Lösung der vorliegenden Aufgabenstellung genutzt werden.

Ebenso wie in [MASS08] werden auch hier Präfixbäume über Graphen aufgebaut, wobei die Teilgraphrelation die Rolle der Präfixrelation einnimmt (Um Verwirrungen zu vermeiden werden die Knoten des zu indizierenden Graphen im Folgenden weiterhin als *Knoten*, die der Indexstruktur als *Baumknoten* bezeichnet). Allerdings entspricht der Übergang von einem Baumknoten zu dessen Kind (und damit der Übergang von einem Teilgraphen zu einem größeren, der ersteren enthält) nicht wie in [MASS08] der Hinzunahme einer Kante, sondern der Hinzunahme eines Knotens des repräsentierten Teilgraphs. Die Verbindungen dieses Knotens zu seinen Nachbarn werden im jeweiligen Baumknoten in Form einer Zeile der Adjazenz-Matrix beschrieben. Zur Konstruktion derartiger Indexstrukturen wird ein, in der Größe der Teilgraphen, quadratischer Algorithmus angegeben.
Beinhaltet eine SPARQL-Anfrage variable Subjekte oder Objekte, so besitzt ihre Graph-Repräsentation korrespondierende *unbeschriftete* Knoten. Um solche Anfragen beantworten zu können müssen somit auch entsprechende Teilgraphen indiziert werden. Erfreulicherweise behandelt das in [RS10] vorgeschlagene Verfahren auch diesen Fall. Sind die Knoten des zu indizierenden Graphen unbeschriftet und somit a priori alle gleichwertig, so können strukturell isomorphe Teilgraphen auftreten (an dieser Stelle soll ein intuitiver Begriff des besagten Isomorphismus genügen; eine formale Definition würde den Rahmen dieser Abhandlung sprengen). Um diese mittels eines gemeinsamen Baumknotens repräsentieren zu können wird daher eine kanonische Beschriftung zuvor unbeschrifteter Knoten in Form der Anzahl, zu einem Knoten inzidenter Kanten, eingeführt.

Leider erweist sich eine derartige Beschriftung bei genauerer Betrachtung als nicht zwangsläufig eindeutig: Für den - nicht unwahrscheinlichen - Fall mehrerer Knoten mit der selben Anzahl inzidenter Kanten ist deren Rolle nicht mehr eindeutig bestimmt und man sieht sich erneut mit dem Problem isomorpher Teilgraphen konfrontiert, die durch unterschiedliche Knoten des G-Trie repräsentiert werden. In der Folge kann es bei einer Suche über dem G-Trie notwendig werden, mehrere von einem Knoten ausgehende Pfade zu verfolgen. Die Anzahl der Pfade entwickelt sich dadurch im schlimmsten Fall exponentiell mit der Tiefe des G-Trie und damit der Größe indizierter Teilgraphen. Da angesichts der feingranularen Datenmodellierung in RDF Knoten nicht selten mehrere Dutzend inzidente Kanten aufweisen und die zu betrachtenden Teilgraphen somit entsprechend groß werden ist ein derartiges Verhalten äußerst ungeschickt. Somit kommt auch das Konzept des G-Trie nicht zur Lösung der vorliegenden Aufgabenstellung in Betracht.

3.4.4 Characteristic Sets

Neumann und Moerkotte befassen sich in [NM11] konkret mit einer möglichen Verbesserung der Kardinalitätsabschätzung in RDF-3X, allerdings ohne dabei die Erweiterung für ortsbasierte Anfragen zu berücksichtigen. Ihr Ansatz beruht auf der Beobachtung, dass angesichts der feingranularen Datenmodellierung in RDF sowohl der RDF-Graph als auch potentielle Anfragegraphen oftmals umfangreiche Sternmuster aufweisen. Dabei bildet der mit einer Entität assoziierte Knoten den Mittelpunkt des Sterns und ist durch Kanten mit denjenigen Knoten verbunden, die die konkreten Attributwerte der Entität darstellen. Die

Kanten beschreiben hierbei das jeweilige Attribut. Wird die Kardinalität von Anfrageergebnissen mit Hilfe der vorhandenen und in Abschnitt 2.5.5 näher beleuchteten Statistik bestimmt, so stehen die resultierenden Abschätzungsfehler insofern mit der beschriebenen Stern-Struktur im Zusammenhang, als dass von statistischer Unabhängigkeit der Prädikate ausgegangen wird, während sich viele hiervon auf das gleiche Subjekt beziehen und damit in engem Zusammenhang stehen (eine Entität vom Typ „S-Bahn“ wird sehr wahrscheinlich auch ein Attribut „Linie“ mit einer geringen Zahl in Frage kommender Objektwerte besitzen; sehr viel häufiger jedenfalls, als dies für die Gesamtheit aller Entitäten der Fall ist). Fasst man die Prädikate der Sternkanten zusammen, so ergibt sich eine Charakterisierung der jeweiligen Entität, die eine Art flexibles Schema darstellt. Die Autoren bezeichnen eine solche Zusammenfassung daher als *Characteristic Set*. Empirische Untersuchungen zeigen gemäß [NM11], dass in realen Datensätzen nur vergleichsweise wenige, unterschiedliche, Characteristic Sets auftreten.

Die Idee des Ansatzes besteht nun darin, eine Statistik über deren Häufigkeit zu ermitteln. Die Kardinalität von Anfragen, welche Entitäten anhand eines Sternmusters selektieren, kann dann als Summe aller Vorkommen von Characteristic Sets bestimmt werden, die eine Obermenge des Sternmusters darstellen. Durch die geringe Zahl möglicher Characteristic Sets bleibt ein solches Vorgehen beherrschbar und liefert in seiner Reinform sogar *exakte* Abschätzungen!
Allerdings nur, wenn zu einem festgehaltenen Subjekt und Prädikat maximal ein Objektwert vorkommt. Um auch den allgemeinen Fall mit Multiattributen behandeln zu können wird daher für alle Characteristic Sets zusätzlich vermerkt, wie oft die enthaltenen Prädikate als Attribute der durch das Characteristic Set abgedeckten Entitäten auftauchen. Damit lässt sich bestimmen, wieviele Objektwerte die Entitäten *durchschnittlich* für das jeweilige Prädikat besitzen und wieviele Variablenbindungen demnach für eine Anfrage möglich sind. Dieser Wert ist angesichts der Mittelwertbildung jedoch nicht mehr zwangsläufig exakt, sondern setzt gewissermaßen Gleichverteilung voraus.
Sind einzelne Objektwerte in Form von Konstanten festgelegt, so wird die Kardinalität des Anfrageergebnisses dadurch reduziert. Um diese Tatsache modellieren zu können wird daher für alle Prädikate eines Characteristic Sets zusätzlich die Anzahl o_i der mit dem jeweiligen Prädikat auftretenden, unterschiedlichen Objektwerte vermerkt. Sei c die Kardinalitätsabschätzung für eine Anfrage mit variablem Objekt. Dann wird $\frac{c}{o_i}$ als Abschätzung für die entsprechende Anfrage mit *festem* Objekt verwendet. Durch die individuelle Erhebung des Wertes o_i für sämtliche Characteristic Sets und Prädikate wird dabei sowohl die statistische Abhängigkeit vom jeweiligen Prädikat, als auch in gewissem Maße die Abhängigkeit von anderen Attributen der Entität reflektiert.

Auch in RDF-Modellierungen existieren oftmals Attribute, die angesichts ihrer Semantik quasi Schlüssel für die jeweilige Entität darstellen. Von diesen Schlüsseln hängen oftmals wiederum die übrigen Attribute der Entität weitgehend funktional ab. Die Annahme statistischer Unabhängigkeit führt somit häufig zu drastischen Unterschätzungen! Der Ansatz von Neumann und Moerkotte verwendet daher zur Selektivitätsabschätzung einer sternförmigen Anfrage lediglich den *minimalen* Selektivitätsfaktor aller auftretenden Prädikate mit festem Objekt. Als Selektivitätsfaktor der übrigen festgehaltenen Objekte hingegen dient der Wert 1.

Während bislang lediglich Sternstrukturen mit gemeinsamem Subjekt betrachtet wurden, sind ebenso auch Strukturen mit gemeinsamem Objekt möglich. Die in [NM11] beschriebene Implementierung verwendet diese beiden Sichten daher parallel, bevorzugt jedoch die Variante mit gemeinsamem Subjekt.

Durch die Abschätzung mittels Characteristic Sets wird nicht mehr die *Selektivität* eines einzelnen Joins bestimmt, sondern vielmehr die *Kardinalität* der durch einen Teil der Anfrage beschriebenen Resultatsmenge, die unter Umständen mehrere Joins und Tripelmuster beinhaltet. Entsprechend ist zur Abschätzung der Gesamtkardinalität ein leicht modifiziertes Vorgehen erforderlich. Dabei wird zunächst versucht, möglichst weite Teile des Anfragegraphen durch passende, vorhandene Characteristic Sets zu „überdecken“. Für die verbleibenden Tripelmuster und Joins sowie die Einbindung der „überdeckten“ Bereiche werden anschließend die bereits vorhandenen Berechnungskonzepte zur Abschätzung verwendet.

Im Zuge einer ausführlichen Evaluation wird der beschriebene Ansatz in [NM11] mit der bisherigen Selektivitätsabschätzung von RDF-3X ([NM11]) sowie weiteren Ansätzen verglichen. Hierzu zählen unter anderem das bereits erläuterte Verfahren von Stocker et al. [SSB+08] sowie der P-Tree-Ansatz aus [MASS08]. Die Characteristic Sets erweisen sich dabei als überlegenes Konzept zur Abschätzung der Kardinalität von SPARQL-Anfragen. Leider werden jedoch lediglich Beispielanfragen mit ausschließlich sternförmigen Anfragegraphen untersucht. Da eine solche Form exakt den indexierten Strukturen entspricht ist naturgemäß ein hervorragendes Verhalten des Algorithmus zu erwarten. Interessant wäre hingegen vielmehr gewesen, wie das beschriebene Verfahren mit Anfragen zurecht kommt, die sternförmige Strukturen lediglich als *Teil* ihres Anfragegraphen beinhalten. Es steht zu erwarten, dass hierfür nur sehr viel ungenauere Abschätzungen möglich sind.
Die einseitige Fokussierung auf sternförmige Strukturen und das Fehlen von Konzepten zur Behandlung *allgemeiner* Graphstrukturen unter Berücksichtigung statistischer Abhängigkeiten sind gleichsam ein mögliches Hindernis für den effizienten Einsatz des Verfahrens zur Lösung der vorliegenden Aufgabenstellung: Zwar erscheint es zunächst einmal naheliegend, die Geo-Literale der betrachteten Datensätze als Spitzen einer Sternstruktur um die zugehörige Entität zu betrachten und entsprechend eine Statistik aufzubauen. Jedoch sollte ein derartiges Verfahren auch mit Modellierungen zurecht kommen, die *partiell* Ortsobjekte verwenden. In einem solchen Fall jedoch degeneriert der betrachtete Stern zu einer Kombination zweier Kanten, die nicht zur Modellierung weiterer Attribute der zugehörigen Entität dienen. Entsprechend reflektiert eine solche Statistik auch nicht mögliche Korrelationen zwischen dem Ort einer Entität und ihren weiteren Attributen. Dies jedoch ist das zentrale Anliegen der Aufgabenstellung. Hierfür wäre es vielmehr notwendig, mittels geeigneter Primitive adaptiv die Umgebung der zugehörigen Entität zu erfassen. Egal, ob diese vom Geo-Literal aus über eine oder zwei Kanten erreicht wird.

3.4.5 Pfadansätze in XML-Datenbankverwaltungssystemen

Da es sich bei XML, analog zu RDF, um ein Konzept zur Modellierung semi-strukturierter Daten handelt, die darüber hinaus eine Graphinterpretation ermöglichen, kommen Verfahren aus entsprechenden Systemen möglicherweise auch als Grundlage zur Lösung der vor-

liegenden Aufgabenstellung in Betracht. Polyzotis und Garofalakis beschreiben in [PG02] ein Verfahren zur Abschätzung der Selektivität von Anfragen über XML-Datenbanken. Allerdings konzentriert sich der Ansatz auf die Behandlung vergleichsweise langer Pfade. Dabei werden außerdem nur *direkt* vom Hauptpfad abzweigende Teilpfade zugelassen. Eine weitere Aufspaltung der Teilpfade ist nicht vorgesehen. Auf diese Weise kann nur ein kleiner Teil der prinzipiell möglichen, allgemeinen Graphstrukturen präzise erfasst werden. Eine solche Einschränkung ist jedoch gewissermaßen konträr zu der in RDF-Modellierungen vorherrschenden Charakteristik vergleichsweise kurzer Pfade, die jedoch eine enge Vermaschung aufweisen.

Moraes Filho und Härder beschäftigen sich in [MFH08] ebenfalls mit der Kardinalitätsabschätzung für Anfragen über XML-Datenbanken. Allerdings werden hier lediglich Pfadstrukturen *ohne* die Möglichkeit von Verzweigungen betrachtet. Da die im Rahmen der vorliegenden Arbeit betrachteten Korrelationen im Allgemeinen jedoch zwischen dem Ort und *mehreren* Attributen der zugehörigen Entität bestehen, erscheinen simple Pfade nicht als adäquates Konstrukt zur Beschreibung besagter Korrelationen, da sie die aus der feingranularen RDF-Modellierung resultierenden Sternstrukturen immer nur partiell abbilden können.
Die in [AMFH08] betrachteten, etwas allgemeineren Pfadstrukturen entsprechen im Wesentlichen denen aus [PG02] und sind daher für eine Verwendung im Rahmen der vorliegenden Arbeit gleichermaßen ungeeignet. Der Ansatz in [AMFHS09] beruht fundamental auf den Achsen-Beziehungen in XML, die so jedoch in RDF nicht gegeben sind. Daher ist auch ein Einsatz dieses Verfahrens zur Lösung der hier betrachteten Aufgabenstellung leider nicht möglich.

4 Lösungsansatz

Da sich die in Kapitel 3 untersuchten Konzepte aus den jeweils genannten Gründen allesamt als untauglich erwiesen haben, um damit die hier betrachtete Aufgabenstellung zu lösen, soll nun ein angepasster und geeigneter Ansatz entwickelt werden. Einige Teilaspekte der zuvor behandelten Konzepte dürften trotzdem gewinnbringend nutzbar sein und werden daher an den entsprechenden Stellen in den besagten Lösungsansatz einfließen.
Ziel dieser Arbeit ist es, die Optimierung von Ausführungsplänen in RDF-3X zu verbessern. Angesichts der zu Grunde liegenden Implementierung dieses Systems sind dazu zweierlei Aufgaben zu lösen. Zum einen sind dem Optimierer adäquate Kardinalitätsabschätzungen für ortsbasierte Anfragen zur Verfügung zu stellen, auf deren Basis er Entscheidungen hinsichtlich der Struktur möglicher Ausführungspläne treffen kann. Diese Problemstellung wird in Abschnitt 4.1 behandelt und mittels Kachelung (Abschnitt 4.1.1) sowie durch den Pfadbündel-Ansatz (Abschnitt 4.1.2) gelöst. Des Weiteren ist zu entscheiden, ob in einem konkreten Fall die ortsbasierte Filterung in Anfragen mittels Spatial-Index- oder Spatial-Selection-Operator realisiert werden soll. Dieser Aspekt wird daher anschließend in Abschnitt 4.2 behandelt.

4.1 Kardinalitätsabschätzung

Um die Problematik der Kardinalitätsabschätzung zu lösen, soll im Folgenden eine Kombination zweier Konzepte zum Einsatz kommen. Das erste hiervon ist eine Unterteilung des betrachteten Gebietes in sogenannte Kacheln. Hierdurch wird zum einen eine Abschätzung der Kardinalität für Anfragen möglich, die *ausschließlich* eine ortsbasierte Auswahl von Geo-Objekten spezifizieren. Zum anderen hat eine solche Unterteilung auch positive Auswirkungen auf die Kardinalitätsabschätzung *allgemeiner* Anfragen. Letztere wird insbesondere durch die Pfadbündel-Statistik möglich, die das zweite der beiden Konzepte darstellt.
Zunächst einmal sollen jedoch Hintergründe und Details der Kachelung genauer betrachtet werden.

4.1.1 Kachelung

Im Zuge der Optimierung ist es, wie in Abschnitt 2.5.2 erläutert, unter anderem notwendig, Kardinalitätsabschätzungen für Anfragen zu tätigen, die ausschließlich eine ortsbasierte Selektion auf dem betrachteten Gebiet darstellen. Mehr oder weniger präzise Verfahren, die nicht von Gleichverteilung ausgehen, benötigen dazu Statistiken über die Verteilung der

Geo-Objekte. Hierfür bietet sich ein Histogramm-Ansatz an, der das betrachtete Gebiet in kleinere Bereiche unterteilt und die Anzahl der darin jeweils enthaltenen Objekte vermerkt. Besagter Anfragetyp kann dann durch Addition der Werte aller überdeckten Bereiche beantwortet werden. Diese können hierbei grundsätzlich eine nahezu beliebige Form aufweisen. Da aus praktischen Gründen jedoch rechteckige Gebilde anzustreben sind, werden sie im Folgenden als Kacheln bezeichnet.
Eine derartige Unterteilung bietet darüber hinaus auch, wie nachfolgend dargestellt, Vorteile hinsichtlich der Kardinalitätsabschätzung *allgemeiner* Anfragen. Grundlage dieser Arbeit ist bekanntlich die Beobachtung bzw. Vermutung, dass der Ort und weitere Attribute von Geo-Objekten keinesfalls statistisch unabhängig sind, sondern dazwischen oftmals eine Korrelation zu erkennen ist. Beispielsweise dürften Objekte vom Typ „Seehafen“ im Bereich der Sahara eher selten anzutreffen sein. Vielmehr sind sie entlang der Küstenlinien zu finden. Eine Statistik, die derartige Korrelationen reflektiert, muss also insbesondere in geeigneter Weise die Ortsinformation von Geo-Objekten berücksichtigen. Eine Indexierung *präziser*, *individueller* Koordinaten der betrachteten Objekte würde jedoch die Skalierbarkeit des Verfahrens verhindern und entspräche nicht der zusammenfassenden Natur einer Statistik. Es erscheint daher sinnvoll, räumlich benachbarte Geo-Objekte als Einheit zusammenzufassen und im Rahmen der Statistik mit einer gemeinsamen Position (bzw. einem gemeinsamen Bereich) zu assoziieren, wie dies im oben beschriebenen Ansatz der Fall ist. Die Genauigkeit der Ortsinformation wird dadurch zwar reduziert, sollte im Rahmen der Kardinalitätsabschätzung jedoch ausreichend sein. Werden neben dem Ort auch weitere Attribute der Geo-Objekte indexiert und für die einzelnen Kacheln individuelle Statistiken geführt, so bilden sich beim Vorliegen ortsbezogener Korrelationen entsprechende Werteverteilungen der jeweiligen Statistik aus, die in der Regel präzisere Kardinalitätsabschätzungen ermöglichen sollten.

Um derartige individuelle Statistiken zu generieren ist das im nachfolgenden Abschnitt beschriebene Verfahren somit für jede Kachel getrennt auszuführen. Auf diese Weise erscheint die Unterteilung für besagtes Verfahren transparent, so dass dort von einem einzigen, nicht weiter unterteilten Gebiet ausgegangen werden kann. Sowohl die Statistikgenerierung als auch ihre Nutzung sind darüber hinaus in Folge der Kachelung hochgradig parallelisierbar. Des Weiteren wird das Verfahren hierdurch skalierbar, indem für größere zu betrachtende Bereiche lediglich eine größere Zahl an Kacheln verwendet, deren Größe jedoch konstant gehalten wird.

Die Unterteilung in Kacheln kann wie bereits erwähnt auf vielfältige Art und Weise geschehen ([YCCP10, GKTD05, PHIS96]). Die zentrale Aufgabe dieser Arbeit besteht jedoch im Finden eines geeigneten Verfahrens zur Berücksichtigung ortsbezogener Korrelationen bei der Kardinalitätsabschätzung. Daher soll im Folgenden eine vergleichsweise simple Variante der Kachelung verwendet werden, die den betrachteten Bereich durch eine gleichmäßige Gitterstruktur unterteilt. Die Anzahl der Kacheln ist hierbei ein Verfahrensparameter, dessen Einfluss im Rahmen der Evaluation in Kapitel 6 genauer untersucht werden soll. Sind Kacheln durch ein konkretes Anfragefenster nur anteilig überdeckt, so wird von Gleichverteilung ausgegangen und die Zahl der enthaltenen Objekte im Rahmen der Kardinalitätsabschätzung auch nur anteilig gewertet.
Ausgefeiltere Ansätze könnten sicherlich weitere Leistungssteigerungen ermöglichen. Es

ist jedoch davon auszugehen, dass diese lediglich eine Verbesserung um einen konstanten Faktor bewirken und daher aus konzeptioneller Sicht vernachlässigt werden können.

4.1.2 Pfadbündel-Statistik

Nachdem in Abschnitt 4.1.1 die Unterteilung in Kacheln als gewinnbringendes Konzept identifiziert wurde, soll nun ein Verfahren entwickelt werden, das das Problem möglicher Korrelationen innerhalb ortsbasierter Anfragen zu lösen vermag. Hierfür ist jedoch zunächst eine Vorüberlegung notwendig.

Vorüberlegung

Die Annahme statistischer Unabhängigkeit der unterschiedlichen Teile einer Anfrage führt bekanntlich auf nicht ausreichend präzise Kardinalitätsabschätzungen, da Korrelationen zwischen den einzelnen Teilen vorliegen können. Derartige Korrelationen sind jedoch nicht nur zwischen Tripelmustern möglich, die unmittelbar durch einen Join verbunden sind, sondern vielmehr ganz allgemein zwischen beliebigen Teilen der Anfrage. Mit der bisherigen Schnittstelle zwischen Optimierer und Kardinalitäts- bzw. Selektivitätsabschätzung ist es jedoch nicht möglich, solche allgemeinen Korrelationen zu erfassen, da dem Optimierer leglich *kontextunabhängig* Kardinalitätsabschätzungen für einzelne Tripelmuster sowie Selektivitätsabschätzungen für die Kombination zweier Tripelmuster zur Verfügung gestellt werden können. Um beliebige Korrelationen berücksichtigen zu können ist es jedoch notwendig, die resultierende Kardinalität größerer Teile der Anfrage *direkt* mit einem geeigneten Verfahren abzuschätzen und dem Optimierer den jeweiligen Wert zur Weiterverarbeitung zu übergeben. Da derartige Kardinalitätsabschätzungen tendenziell genauere Werte liefern dürften als der bisherige Ansatz von RDF-3X, sollten möglichst große Teile der Anfrage auf diese Weise abgeschätzt werden. Zur Berechnung der Gesamtkardinalität kann der Optimierer anschließend die übergebenen Werte mit den bestehenden Mechanismen kombinieren. Im Falle mehrerer ortsbasierter Filter können dem Optimierer Abschätzungen für disjunkte, an die verschiedenen Geo-Variablen angrenzende Teile des Anfragegraphen übergeben und ebenfalls mit besagten Mechanismen kombiniert werden. Die weiteren Überlegungen lassen sich somit auf den Sonderfall eines einzigen ortsbasierten Filters beschränken.
Der geschilderte Gedankengang steht im Einklang mit der korrespondierenden Überlegung aus [NM11]. Die in Kapitel 5 beschriebene Implementierung des Lösungsansatzes realisiert daher eine entsprechende Erweiterung des Optimierers von RDF-3X.

Konzeption der Pfadbündel-Statistik

Wie in Kapitel 3 ausführlich erläutert wurde, sind Histogramm-basierte Verfahren nicht zur Lösung der vorliegenden Aufgabenstellung geeignet. Die Graphinterpretation von RDF legt vielmehr eine Indexierung von Graphstrukturen sowie eine Kardinalitätsabschätzung auf deren Grundlage nahe. Die entsprechenden Verfahren aus Kapitel 3 lassen sich grob in

zwei Klassen aufteilen: Die erste hiervon indexiert vergleichsweise simple Strukturen wie Pfade ([MFH08]), erweiterte Pfade ([AMFH08, PG02]), Characteristic Sets ([NM11]) oder kleine Graphen ([BDGL08]) und ermöglicht damit lediglich Kardinalitätsabschätzungen für eine Teilmenge aller grundsätzlich möglichen Strukturen. Die zweite der beiden Klassen hingegen indexiert vergleichsweise komplexe Strukturen, nämlich allgemeine Graphen ([RS10, MASS08]. Diese werden sequentialisiert und ihre Sequentialisierung unter Ausnutzung gemeinsamer Strukturen in einem Präfixbaum verwaltet. Allerdings können auf diese Weise nur gemeinsame Strukturen bis zum ersten Unterschied der Sequentialisierungen, d.h. bis zur Aufspaltung des gemeinsamen Pfades im Baum, gemeinsam repräsentiert werden. Werden in einer Anfrage, beispielsweise in Form von Variablen, gemeinsame Strukturen spezifiziert, so müssen gegebenenfalls mehrere Zweige des Baumes verfolgt werden. Durch mehrfache derartige Aufspaltungen kann sich eine exponentielle Zahl zu besuchender Zweige ergeben. Auch eine Vorabberechnung und Aggregation löst dieses Problem nicht: Da ein Graph eine exponentielle Zahl an Teilgraphen, und damit möglicherweise spezifizierten Strukturen, beinhaltet müsste eine ebenso große Zahl vorab berechneter Datensätze hinterlegt werden. Das Problem ist also konzeptioneller Natur und resultiert aus der Verwendung von Sequentialisierungen, die jedoch zur Indexierung von Graphen auf eindimensionalem Speicher zwingend erforderlich sind. Schlussendlich folgt das Problem also bereits aus der Wahl von Graphen als zu indexierenden Objekten.
Gleichzeitig ist für den betrachteten Anwendungskontext nicht zwingend eine präzise Vorhersage der Kardinalität erforderlich. Vielmehr muss lediglich die Tendenz der Abschätzung richtig sein, so dass vom Optimierer jeweils die „korrekten“ Entscheidungen getroffen werden. Angesichts dieses Umstandes und der jeweiligen Beschränkungen beider Klassen von Indexierungsverfahren wäre ein Mittelweg erstrebenswert, der auf Kosten der Präzision die jeweiligen Vorteile beider Klassen zu kombinieren versucht, ohne jedoch deren Nachteile in Kauf nehmen zu müssen. Ein solcher Mittelweg ist möglich, wenn zwar für allgemeine Graphen Kardinalitätsabschätzungen getätigt werden, diese jedoch auf einer geeigneten Kombination indexierter *Pfade* beruhen. Ein solcher Ansatz gestattet die Nutzung einer vergleichsweise geringen Zahl indexierter Objekte als gemeinsame Strukturelemente einer unter Umständen variantenreichen und damit großen Menge von Graphen.
Eine geeignete Operation zur Kombination der Pfade ergibt sich durch folgende Überlegung: Gemäß [NM11] sind Überschätzungen der Kardinalität grundsätzlich Unterschätzungen vorzuziehen. Eine derartige Überschätzung ergäbe sich beispielsweise durch die Verwendung einer oberen Schranke der zu erwartenden Kardinalität. Eine solche obere Schranke wiederum korrespondiert jeweils mit einer entsprechenden notwendigen, aber nicht zwingend hinreichenden Bedingung für das Auftreten des gesuchten Graphen. Eine solche notwendige Bedingung wäre ihrerseits im betrachteten Fall durch eine konjunktive Verknüpfung der im Anfragegraph enthaltenen Pfade gegeben und zudem bei geeigneter Implementierung sehr effizient berechenbar.

Der Kern des Lösungsansatzes besteht also in einer Indexierung auftretender Pfade mitsamt der Menge ihrer jeweiligen Vorkommen. Durch Berechnung der Schnittmenge für die in einem Anfragegraphen enthaltenen Pfade ergibt sich daraus eine Obermenge der Vorkommen des Anfragegraphen, deren Kardinalität den Ausgangswert der Abschätzung bildet. Die Schnittmengenbildung kann hierbei als eine Art „Bündelung“ eigenständiger

Pfade aufgefasst werden. Das Verfahren wird daher im Folgenden als Pfadbündel-Statistik bezeichnet.
Nachfolgend soll dieser Ansatz durch Betrachtung einiger Teilaspekte verfeinert und optimiert werden. Im Anschluss werden die dabei getroffenen Design-Entscheidungen in einer formalen Darstellung zusammengefasst und ihre Anwendung anhand eines konkreten Beispiels veranschaulicht.

Einschränkung der Menge indexierter Pfade: Da im vorliegenden Fall ausschließlich an Geo-Literale angrenzende Teilgraphen betrachtet werden, ist es ausreichend, lediglich Pfade zu indexieren, die an einem Geo-Literal beginnen. Sämtliche grundsätzlich möglichen Pfade lassen sich als Teilsequenzen hiervon auffassen.

Beschränkung auf Indexierung der Prädikate: Werden bei der Kardinalitätsabschätzung Pfade *mitsamt* ihrer Knotenbeschriftungen betrachtet und mittels Schnittmengenbildung mögliche Vorkommen des Anfragegraphen identifiziert, so ist diese überaus feingranulare Betrachtung äußerst zeitintensiv und nimmt bereits große Teile der Abarbeitung des Ausführungsplanes vorweg. Da nur eine ungefähre Abschätzung notwendig ist und die Prädikate der Kanten bzw. Tripel die wesentlichen Strukturmerkmale des RDF-Graphen darstellen, sollen daher nur sie in den Index aufgenommen werden. Die Kombination unterschiedlicher Pfade lässt sich dadurch deutlich einfacher berechnen. Pfade mit identischer Prädikatsequenz lassen sich außerdem einer gemeinsamen Klasse zuordnen und durch einen einzigen Eintrag im Index repräsentieren. Im Folgenden wird ein Pfad daher mit seiner Prädikatsequenz assoziiert.

Wie nachfolgend gezeigt wird, ist es trotz dieser grundsätzlichen Beschränkung der Indexierung auf Prädikate ohne deutlichen Mehraufwand möglich, das Ende von Pfaden detaillierter zu beschreiben, um dadurch die Genauigkeit der Abschätzung zu erhöhen. Hierbei sind zur Behandlung variabler und konstanter Pfadenden in Anfragegraphen unterschiedliche Ansätze notwendig. Ebenso sind zur Behandlung mehrerer Pfade eines Typs geeignete Mechanismen zu deren Kombination notwendig, die ebenfalls im Folgenden dargelegt werden.

variables Pfadende: Besitzen modellierte Entitäten Multiattribute in Form mehrerer Kanten, so sind für eine entsprechende Variable mehrere Bindungen und somit auch mehrere Ergebnistupel möglich. Dieser Fall kann im bisherigen Entwurf jedoch noch nicht von einelementigen Attributen unterschieden werden. Die Werte von Multiattributen stellen im RDF-Graphen allerdings stets Endpunkte von Pfaden mit identischer Prädikatsequenz dar. Als erste Annäherung an eine verbesserte Behandlung der Problematik soll daher die mittlere Anzahl f_j der Vorkommen eines Pfades π_j in den bei Geo-Literalen beginnenden Teilgraphen vermerkt werden. Ist der Pfad π_j Teil eines Anfragegraphen, so lässt sich die bisherige Kardinalitätsabschätzung auf dieser Grundlage durch Multiplikation mit dem Wert f_j verbessern. Ein solcher Ansatz reflektiert zwar nicht in jedem Fall die tatsächliche Konstellation

innerhalb der in Frage kommenden Teilgraphen; laut der bereits dargestellten Überlegung genügen jedoch ausreichend präzise Näherungen.
Gemäß der Semantik von SPARQL sind im Falle mehrerer Multiattribute sämtliche Kombinationen ihrer Werte Teil der Lösungsmenge einer Anfrage, die Variablen für besagte Attribute spezifiziert. Entsprechend sind die Werte f_j der zugehörigen Pfade π_j zu multiplizieren.

konstantes Pfadende: Um die selektive Wirkung konstanter Pfadenden in einem Anfragegraph reflektieren zu können, ist die Werteverteilung der Knotenbeschriftungen am Ende aller Pfade einer Klasse in geeigneter Weise zu repräsentieren. Motiviert durch das analoge Vorgehen in [NM11] soll hierfür im Folgenden eine vergleichsweise simple Methode Anwendung finden, die lediglich die *Anzahl* e_j *unterschiedlicher* Knotenbeschriftungen vermerkt und von deren Gleichverteilung ausgeht. Betrifft das Pfadende ein Multiattribut, so selektieren unterschiedliche Konstanten nicht zwingend unterschiedliche Entitäten, da mehrere Attributwerte pro Entität auftauchen können. Eine Verwendung des „kanonischen" Selektivitätsfaktors $\frac{1}{e_j}$ würde daher eine Unterschätzung nach sich ziehen. Legt man Gleichverteilung zu Grund und geht - wie oben - im Mittel von f_j Attributwerten pro Entität aus, so verteilen sich die e_j beobachteten Konstanten auf $\frac{e_j}{f_j}$ Entitäten. Entsprechend ergibt sich der zugehörige Selektivitätsfaktor zu $\frac{f_j}{e_j}$.
Ebenso aus [NM11] entliehen sind die folgende Überlegung sowie die daraus resultierende Design-Entscheidung: Die hier betrachteten Teilgraphen beschreiben im Wesentlichen Geo-Objekte mitsamt ihren jeweiligen Attributen und damit einzelne Entitäten. Diese sind gemäß [NM11] jedoch oftmals durch eine Art Schlüsselattribut bestimmt, von dem die übrigen Attribute quasi funktional abhängen. Die Annahme statistischer Unabhängigkeit führt daher zu einer unter Umständen drastischen Unterschätzung, da die übrigen Attribute nahezu keine selektive Wirkung mehr zeigen, wenn bereits die Auswahl durch das Schlüsselattribut berücksichtigt wurde. Der tatsächliche Selektivitätsfaktor ist somit im Wesentlichen durch das Schlüsselattribut bestimmt. Dieses wiederum stellt üblicherweise das selektivste Attribut einer Entität dar. Entsprechend soll - in Übereinstimmung mit [NM11] - ausschließlich das selektivste konstante Attribut einer Anfrage berücksichtigt werden.

Formale Definition der Pfadbündel-Statistik

Fasst man obige Überlegungen zusammen, so ergibt sich daraus folgender Ansatz.

Sei $\mathcal{T} = \{(S_1, P_1, O_1), (S_2, P_2, O_2), \ldots\}$ die Menge aller RDF-Tripel und $\tilde{\mathcal{T}} = \{(x, p, y) \mid (x, p, y) \in \mathcal{T} \vee (y, p, x) \in \mathcal{T}\}$. Sei darüber hinaus $\mathcal{G} = \{G_1, \ldots, G_l\}$ die Menge aller Geo-Literale. Der Ansatz betrachtet dann Pfade im RDF-Graphen, die bei einem Geo-Literal beginnen und maximal die Länge d besitzen, d.h. Sequenzen

$$((n_1, p_1, n_2), (n_2, p_2, n_3), \ldots, (n_k, p_k, n_{k+1}))$$

mit

$$\begin{aligned} & n_1 \in \mathcal{G} \\ \wedge \quad & (n_i, p_i, n_{i+1}) \in \tilde{\mathcal{T}} \\ \wedge \quad & \forall r, s \in 1, \ldots, k-1 : r \neq s \Rightarrow n_r \neq n_s \\ \wedge \quad & \neg(n_{k+1} = n_{k-1} \wedge p_k = p_{k-1}) \\ \wedge \quad & k \leq d \end{aligned}$$

Die Pfadbündel-Statistik $\mathcal{S}$ ist dann eine Menge von Tupeln (π_j, A_j, e_j, f_j), d.h. $\mathcal{S} = \{(\pi_1, A_1, e_1, f_1), (\pi_2, A_2, e_2, f_2), \ldots\}$ mit

$$\begin{aligned} (\pi_j, A_j, e_j, f_j) \in \mathcal{S} \Longleftrightarrow P_j = \{((n_1, p_1, n_2), (n_2, p_2, n_3), \ldots, (n_k, p_k, n_{k+1})) \mid \\ n_1 \in \mathcal{G} \\ \wedge (n_1, p_1, n_{i+1}) \in \tilde{\mathcal{T}} \\ \wedge \forall r, s \in 1, \ldots, k-1 : r \neq s \Rightarrow n_r \neq n_s \\ \wedge \neg(n_{k+1} = n_{k-1} \wedge p_k = p_{k-1}) \\ \wedge k \leq d\} \\ \wedge \pi_j = (p_1, p_2, \ldots, p_k) \\ \wedge A_j = \{n_1 \mid ((n_1, p_1, n_2), \ldots) \in P_j\} \\ \wedge e_j = \left|\{n_{k+1} \mid ((n_1, p_1, n_2), \ldots, (n_k, p_k, n_{k+1})) \in P_j\}\right| \\ \wedge f_j = \frac{|P_j|}{|A_j|} \end{aligned}$$

P_j ist hierbei die Menge aller Pfade mit der Prädikat-Sequenz $\pi_j = (p_1, p_2, \ldots, p_k)$. Letztere dient gewissermaßen als Schlüssel für die ihr zugeordneten Metadaten A_j, e_j und f_j. A_j ist die Menge aller Geo-Literale, an denen ein entsprechender Pfad beginnt. e_j ist die Anzahl der am Ende dieser Pfade auftretenden Knotenbeschriftungen und f_j gibt an, wieviele solcher Pfade durchschnittlich an einem Geo-Literal beginnen. Ein Eintrag (π_j, A_j, e_j, f_j) beschreibt somit die Menge aller Pfade mit der Prädikat-Sequenz $\pi_j = (p_1, p_2, \ldots, p_k)$.

Die beschriebene Statistik stellt bereits eine deutliche Zusammenfassung der Graph-Struktur dar. Sind in einem konkreten Anwendungsfall auch weniger präzise Abschätzungen ausreichend, so kann ihr Umfang auf Basis der folgenden Überlegung jedoch noch weiter reduziert werden: Fehlt ein Eintrag (π_j, A_j, e_j, f_j) und somit auch die von ihm getragene Information, so kann sich dies bei einer ausschließlich auf der Statistik $\mathcal{S}$ beruhenden Betrachtung der Daten lediglich auf die Beurteilung derjenigen Teilgraphen auswirken, die einen entsprechenden Pfad beinhalten. Vor diesem Hintergrund sollten zur Reduktion des Speichervolumens primär diejenigen Einträge eliminiert werden, die nur wenige Teilgraphen betreffen und deren Fehlen somit nur einen geringen Abschätzungsfehler bewirkt. Entsprechend ergibt sich eine weitere Zusammenfassung $\overline{\mathcal{S}}$ der Statistik $\mathcal{S}$ gemäß

$$\overline{\mathcal{S}} = \{(\pi, A, e, F) \in \mathcal{S} \mid \forall (\pi', A', e', f') \notin \overline{\mathcal{S}} : \left|A'\right| \leq |A|\}$$

mit $\left|\overline{\mathcal{S}}\right| \leq c \cdot |\mathcal{S}|$ maximal. c ist hierbei ein vom Benutzer vorzugebender Reduzierungsfaktor. Wird den verschiedenen Kacheln ein jeweils individueller Wert zugeordnet, so ist auf diese Weise eine Bevorzugung bestimmter Kacheln auf der Basis von Hintergrundwissen möglich. Wird hingegen ein globaler Wert verwendet, so resultiert hieraus ein Verhalten, das *automatisch* sowohl Kacheln mit einer erhöhten Zahl an Geo-Objekten, als auch solche mit vergleichsweise dichten, angrenzenden Teilgraphen bevorzugt, da beide Fälle eine erhöhte Zahl an Pfaden nach sich ziehen.

Beispiel einer Pfadbündel-Statistik

Die Struktur der so definierten Pfadbündel-Statistik soll nun anhand eines konkreten Beispiels illustriert werden. Als Grundlage hierfür dient der in Abbildung 4.1 dargestellte Ausschnitt einer möglichen Modellierung des Stuttgarter ÖPNV-Netzes.

Legt man die in Tabelle 4.1 eingeführten Abkürzungen der Konstanten zu Grunde und betrachtet die bei Geo-Literalen beginnenden Teilgraphen der Tiefe 3, so ergeben sich die in den Abbildungen 4.2 und 4.3 dargestellten, zu indexierenden Teilgraphen. G_1 bis G_4 seien hierbei die Geo-Literale der vier Punktobjekte in der Reihenfolge ihres Auftretens. G_5 bzw. G_6 seien die Geo-Literale des Linienzuges bzw. Polygons. Mittels Erkundung der an G_1 bis G_6 beginnenden Pfade der Länge 2 ergibt sich dafür die in Tabelle 4.2 dargestellte Pfadbündel-Statistik $\mathcal{S}$. Teil der eigentlichen Statistik sind dabei nur die Spalten π_j, A_j, e_j, und f_j. Die Spalten $\{n_{k+1} \mid ((n_1, p_1, n_2), \ldots, (n_k, p_k, n_{k+1})) \in P_j\}$ und $\left|P_j\right|$ dienen lediglich dem Verständnis des Zustandekommens der in den Spalten e_j und f_j eingetragenen Werte. Für den Wert $c = 0,4$ ergibt sich daraus die Zusammenfassung $\overline{\mathcal{S}}$, wobei sämtliche Einträge unterhalb der einfachen, horizontalen Linie in Tabelle 4.2 vernachlässigt werden.

α	locatedAt
β	partOf
γ	a
δ	attachedTo
ε	connectedTo
σ	course
τ	area
A	Stadtmitte
B	Hauptbahnhof
C	Charlottenplatz
D	Staatsgalerie
E	Theodor-Heuss-Str.
F	Arnulf-Klett-Platz
G	Schillerstr.
H	Willy-Brandt-Str.
I	S1
J	S2
K	S3
L	U1
M	U2
N	station
O	road
P	square

Tabelle 4.1: Abkürzungen der Konstanten

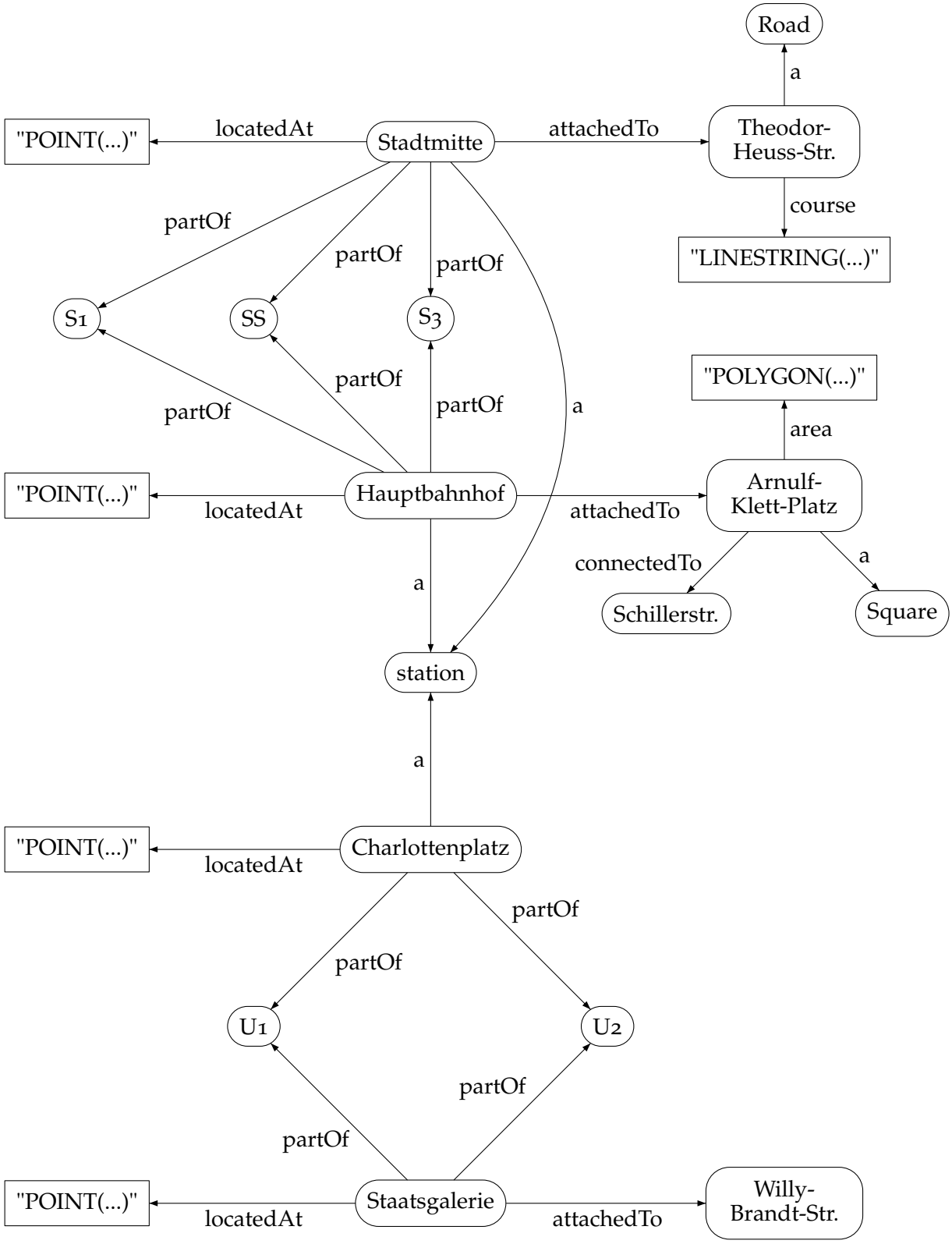


Abbildung 4.1: Ausschnitt einer möglichen Modellierung des Stuttgarter ÖPNV-Netzes

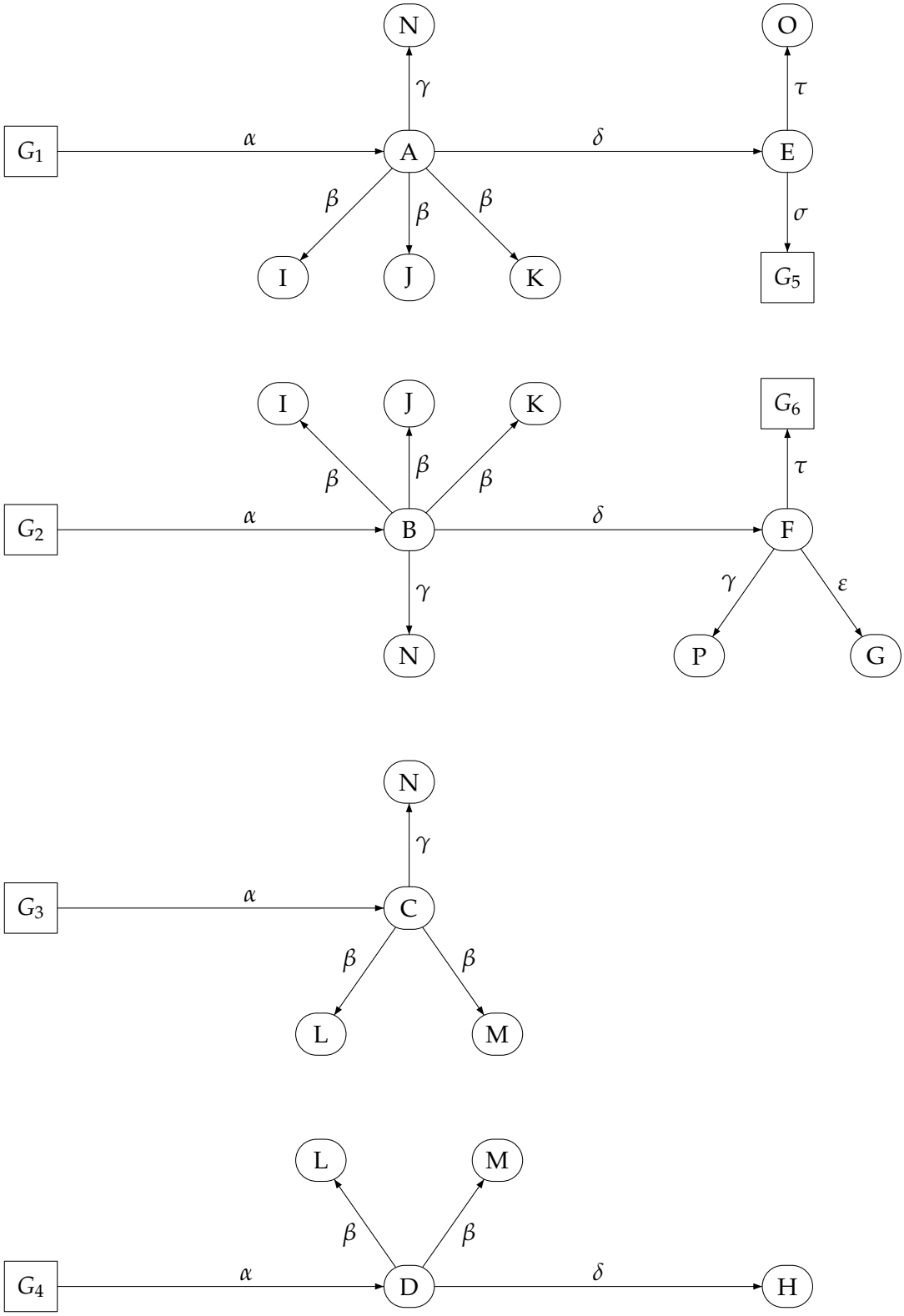


Abbildung 4.2: modifizierte Darstellung der Modellierung, Teil 1

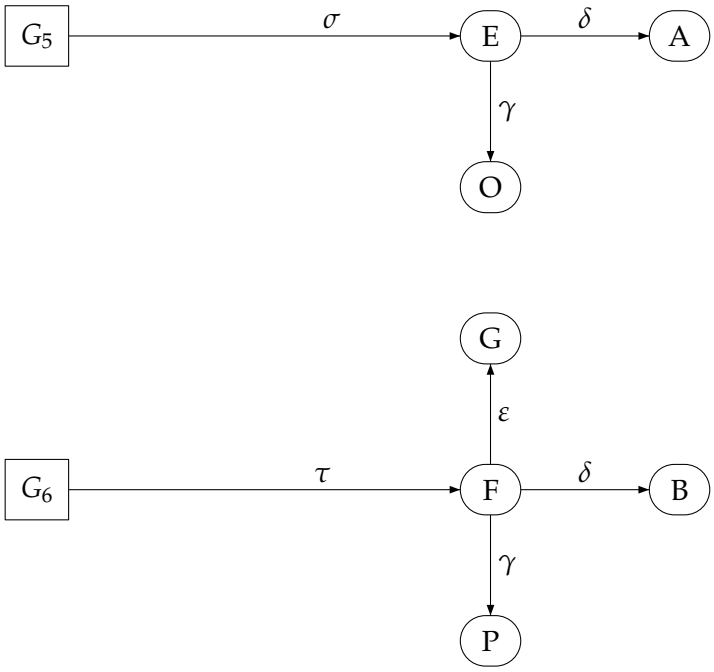


Abbildung 4.3: modifizierte Darstellung der Modellierung, Teil 2

π_j	Aj	$\{n_{k+1} \mid ((n_1, p_1, n_2), \ldots) \in P_j\}$	e_j	$\lvert P_j \rvert$	f_j
α	G_1, G_2, G_3, G_4	A, B, C, D	4	$1+1+1+1$	1
$\alpha\beta$	G_1, G_2, G_3, G_4	I, J, K, L, M	5	$3+3+2+2$	2,5
$\alpha\gamma$	G_1, G_2, G_3	N	1	$1+1+1$	1
$\alpha\delta$	G_1, G_2, G_4	E, F, H	3	$1+1+1$	1
σ	G_5	E	4	1	1
$\sigma\gamma$	G_5	O	1	1	1
$\sigma\delta$	G_5	A	1	1	1
τ	G_6	F	1	1	1
$\tau\delta$	G_6	B	1	1	1
$\tau\varepsilon$	G_6	G	1	1	1
$\tau\gamma$	G_6	P	1	1	1

Tabelle 4.2: Beispiel einer Pfadbündel-Statistik

Kardinalitätsabschätzung mit Hilfe der Pfadbündel-Statistik

Im Folgenden soll nun erläutert werden, wie eine gemäß obigen Regeln generierte Statistik zur Kardinalitätsabschätzung verwendet werden kann. Dabei soll zunächst von einer SPARQL-Anfrage mit ausschließlich *konstanten* Prädikaten ausgegangen und das Verfahren anschließend auf den allgemeinen Fall erweitert werden.

Sei $Q = \{(s,p,o) \mid s,p,o \in V_q \cup V_C\}$ die Menge der Tripelmuster einer Anfrage mit den Variablen $V_Q = \{v_1, v_2, \ldots\}$ sowie den Konstanten $C_Q = \{c_1, c_2, \ldots\}$. Sei darüber hinaus $\tilde{Q} = \{(x,p,y) \mid (x,p,y) \in Q \vee (y,p,x) \in Q\}$. Dann lassen sich die Mengen $P_{Q,V}$ bzw. $P_{Q,C}$ aller bei Geo-Literalen beginnenden Pfade mit variablem bzw. konstantem Ende beschreiben als:

$$\begin{aligned} P_{Q,V} = \{p = ((n1,p1,n2),(n2,p2,n3),\ldots,(n_k,p_k,n_{k+1})) \mid \\ n_1 \in \mathcal{G} \wedge n_{k+1} \in V_Q \wedge (n_i,p_i,n_{i+1}) \in \tilde{Q} \wedge k \leq d \\ \wedge \nexists(((n1,p1,n2),\ldots,(n_k,p_k,n_{k+1}),(n_{k+1},p_{k+1},n_{k+2})) \\ \text{mit } n_{k+2} \in V_Q \wedge (n_{k+1},p_{k+1},n_{k+2}) \in \tilde{Q} \wedge k+1 \leq d)\} \end{aligned}$$

$$\begin{aligned} P_{Q,C} = \{p = ((n1,p1,n2),(n2,p2,n3),\ldots,(n_k,p_k,n_{k+1})) \mid \\ n_1 \in \mathcal{G} \wedge n_{k+1} \in C_Q \wedge (n_i,p_i,n_{i+1}) \in \tilde{Q} \wedge k \leq d \\ \wedge \forall j = 2 \ldots k : n_k \in V_Q\} \end{aligned}$$

Sei desweiteren

$$\pi : ((n_1,p_1,n_2),\ldots,(n_k,p_k,nk+1)) \mapsto (p_1,p_2,\ldots,p_k)$$

eine Abbildung von Pfaden des Anfragegraphen auf die Menge der zu ihm passenden Einträge in der Pfadbündel-Statistik $\mathcal{S}$.

Zur Kardinalitätsabschätzung sind nun die Werte A_Q, μ_Q und σ_Q wie folgt zu bestimmen:

$$\begin{aligned} A_Q &= \bigcap_{p \in P_{Q,C} \cup P_{Q,V}, (\pi(p),A,e,f) \in \overline{\mathcal{S}}} A \\ \mu_Q &= \prod_{p \in P_{Q,C} \cup P_{Q,V}, (\pi(p),A,e,f) \in \overline{\mathcal{S}}} f \\ \sigma_Q &= \min(\{\frac{f}{e} \mid (\pi(p),A,e,f) \in \overline{\mathcal{S}} \wedge p \in P_{Q,C}\}) \end{aligned}$$

Die gesuchte Kardinalitätsabschätzung $card_{est}$ ergibt sich daraus zu:

$$card_{est} = \begin{cases} |A_Q| \cdot \mu_Q \cdot \sigma_Q & \forall p \in P_{Q,C} \cup P_{Q,V} : \exists \pi(p) \in \overline{\mathcal{S}} \\ negl_{max} & \text{sonst} \end{cases}$$

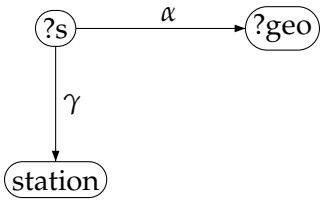
$$\text{mit } negl_{max} = \max(\{t \mid \exists s_j = (\pi_j, A_j, e_j, f_j) \in \mathcal{S} \wedge |A| = t \wedge s \notin \overline{\mathcal{S}}\})$$

Sind alle relevanten Pfade des Anfragegraphen in $\overline{\mathcal{S}}$ enthalten, so ergibt sich die Kardinalität entsprechend dem ersten Fall aus der Anzahl $|A_Q|$ der Teilgraphen, die den Anfragegraphen enthalten *könnten*, der durch variable Pfadenden bedingten Multiplizität μ_Q sowie der durch konstante Pfadenden bedingten Selektivität σ_Q. Sind hingegen *nicht* alle relevanten Pfade des Anfragegraphen in $\overline{\mathcal{S}}$ enthalten, so existiert kein zum Anfragegraphen passender Teilgraph, oder die fehlenden Pfade wurden durch den Übergang von $\mathcal{S}$ nach $\overline{\mathcal{S}}$ vernachlässigt. Um Unterschätzungen ausschließen zu können wird daher von letzterem Fall ausgegangen. Ein fehlender Eintrag (π_j, A_j, e_j, f_j) kann dabei lediglich die Berücksichtigung von Teilgraphen verhindern, die π_j enthalten. Es werden demnach maximal $\left|A_j\right|$ Teilgraphen fälschlicherweise nicht berücksichtigt. Der maximale Abschätzungsfehler ergibt sich daher im zweiten Fall aus dem maximalen Wert $\left|A_j\right|$ vernachlässigter Pfade.

Beispiele für Kardinalitätsabschätzungen

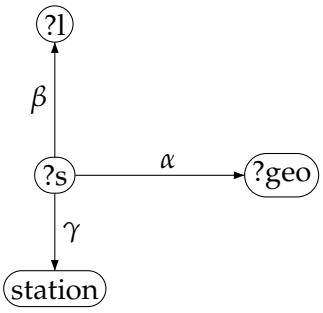
Das Vorgehen zur Kardinalitätsabschätzung soll nachfolgend ebenfalls anhand des obigen Beispiels illustriert werden. Hierzu dienen die im Folgenden dargestellten Beispielanfragen. Ergänzend zur SPARQL-Notation ist dabei auch der korrespondierende Anfragegraph abgebildet. Um die Qualität der Abschätzung einordnen zu können, ist mit *card* die jeweilige tatsächliche Kardinalität angegeben.

Beispielanfrage 1

- ```
  SELECT ?s WHERE {
    ?s a         station .
    ?s locatedAt ?geo .
    FILTER within(?geo, ...)
  }
  ```
- 

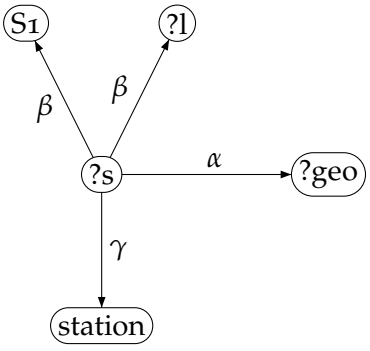
- Kardinalitätsabschätzung: $A_Q = \{G_1, G_2, G_3\}, \quad |A_Q| = 3, \quad \mu_Q = 1, \quad \sigma_Q = 1,$
  $\Rightarrow \quad card_{est} = 3$
  tatsächlich: $card = 3$
  ```

Beispielanfrage 2

- ```
  SELECT ?s WHERE {
    ?s a         station .
    ?s partOf    ?l
    ?s locatedAt ?geo .
    FILTER within(?geo, ...)
  }
  ```
- 

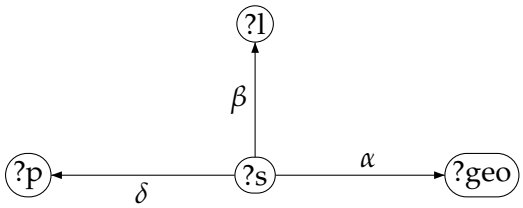
- Kardinalitätsabschätzung: $A_Q = \{G_1, G_2, G_3\}$, $|A_Q| = 3$, $\mu_Q = 2,5$, $\sigma_Q = 1$,
  $\Rightarrow \quad card_{est} = 7,5$
  tatsächlich: $card = 8$

Beispielanfrage 3

- ```
  SELECT ?s ?l WHERE {
    ?s a         station .
    ?s partOf    S1
    ?s partOf    ?l
    ?s locatedAt ?geo .
    FILTER within(?geo, ...)
  }
  ```
- 

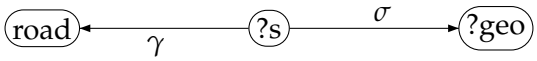
- Kardinalitätsabschätzung: $A_Q = \{G_1, G_2, G_3\}$, $|A_Q| = 3$, $\mu_Q = 2,5$, $\sigma_Q = \frac{1}{2}$,
  $\Rightarrow \quad card_{est} = 3,75$
  tatsächlich: $card = 6$

Beispielanfrage 4

- ```
  SELECT ?s ?l ?p WHERE {
    ?s partOf     ?l
    ?s attachedTo ?p
    ?s locatedAt  ?geo .
    FILTER within(?geo, ...)
  }
  ```
- 

- Kardinalitätsabschätzung: $A_Q = \{G_1, G_2, G_4\}$, $|A_Q| = 3$, $\mu_Q = 2,5$, $\sigma_Q = 1$,
  $\Rightarrow \quad card_{est} = 7,5$
  tatsächlich: $card = 8$

Beispielanfrage 5

- ```
  SELECT ?s WHERE {
    ?s a      road
    ?s course ?geo .
    FILTER within(?geo, ...)
  }
  ```
- 

- Kardinalitätsabschätzung: Pfad nicht in $\overline{\mathcal{S}} \quad \Rightarrow \quad card_{est} = 1$
  tatsächlich: $card = 1$

Erweiterung zur Unterstützung variabler Prädikate

Das vorgestellte Verfahren kann wie folgt vergleichsweise einfach für den allgemeinen Fall konstanter sowie variabler Prädikate erweitert werden. Die Grundidee der Verallgemeinerung besteht darin, anstelle einer konkreten Prädikatsequenz sämtliche in $\mathcal{S}$ bzw. $\overline{\mathcal{S}}$ enthaltenen Einträge zu betrachten, die auf das Muster einer Sequenz mit teilweise variablen Prädikaten passen, und diese in geeigneter Weise zu kombinieren. Hierzu sind lediglich die Abbildung π und die Definitionen der Werte A_Q, μ_Q bzw. σ_Q wie folgt zu verallgemeinern:

$$\pi : ((n_1, p_1, n_2), \ldots, (n_k, p_k, nk+1)) \mapsto \{(p'_1, p'_2, \ldots, p'_k) \in \overline{\mathcal{S}} \mid p_i \in P_{Q,C} \Rightarrow p'_i = p_i\}$$

$$A_Q = \bigcap_{p \in P_{Q,C} \cup P_{Q,V}} \left(\bigcup_{(\hat{p},A,e,f) \in \overline{\mathcal{S}}, \hat{p} \in \pi(p)} A \right)$$

$$f_{avg}(p) = \frac{1}{|\pi(p)|} \cdot \sum_{(\hat{(p)},A,e,f) \in \overline{\mathcal{S}}, \hat{p} \in \pi(p)} f$$

$$\mu_Q = \prod_{p \in P_{Q,V}} f_{avg}(p)$$

$$\sigma_Q = \min \left(\{ \left(\frac{\sum_{(\hat{p},A,e,f) \in \overline{\mathcal{S}}, \hat{p} \in \pi(p)} c}{f_{avg}(p)} \right)^{-1} \mid p \in P_{Q,C} \} \right)$$

4.2 Entscheidung über die Nutzung des Selektionsoperators

Gemäß Abschnitt 2.4.4 verfügt das hier betrachtete System GeoRDF-3X neben einem ortsbasierten Index auch über einen Selektionsoperator. Dieser testet beliebige Eingabetupel mit Geo-Literal hinsichtlich ihrer Zugehörigkeit zu einem spezifizierten Anfragegebiet. Er ist dabei ausschließlich für eine Verwendung an der Wurzel des Operatorbaums vorgesehen, da dort die selektive Wirkung möglichst vieler weiterer Attribute bereits berücksichtigt und somit tendenziell eine möglichst geringe Zahl an Eingabetupeln für die verhältnismäßig teure Selektion zu erwarten ist. Neben der in Abschnitt 4.1.2 vorgestellten Kardinalitätsabschätzung zum Zwecke optimierter Anordnung der Operatoren soll nachfolgend ein Verfahren entwickelt werden, um zu entscheiden, ob der Einsatz eines solchen Selektionsoperators zur Realisierung einer Filter-Klausel der Nutzung des ortsbasierten Index vorzuziehen ist.

4.2.1 Ansatz 1

Angesichts der Optimierung durch dynamische Programmierung bietet GeoRDF-3X quasi ohne zusätzliche Kosten eine hervorragende Grundlage für diese Entscheidung: Seien $T = \{T_1, T_2, \ldots, T_k\}$ die durch einzelne Tripelmuster spezifizierten Teilprobleme einer Anfrage und g deren ortsbasierter Filter. Dann werden mittels dynamischer Programmierung Lösungen für sämtliche Kombinationen der Elemente aus $T \cup \{g\}$ bestimmt, die den ortsbasierten Index zur Realisierung des Filters nutzen. Es werden also insbesondere auch Lösungen für die Teilprobleme $P_1 = T \cup \{g\}$ sowie $P_2 = T$ bestimmt. Da im Rahmen der Lösungsfindung auch Kardinalitäts- sowie Kostenabschätzungen vermerkt werden, können die entsprechenden Einträge von P_1 bzw. P_2 verwendet werden, um auf ihrer Grundlage eine Entscheidung hinsichtlich der Realisierung des Filters g zu treffen.
Seien hierzu $card(P_i)$ die Kardinalität und $costs(P_i)$ die Kosten für die Lösung der Probleme P_1 bzw. P_2. Sei darüber hinaus $costs_\sigma(x)$ ein geeignetes Kostenmaß für die Selektion einer Menge mit Kardinalität x. Dann ist für

$$costs(P_1) > costs(P_2) + costs_\sigma(card(P_2))$$

die Nutzung des Selektionsoperators vorzuziehen und dieser an der Wurzel des für P_2 ermittelten Operatorbaums zu platzieren.
Zur Auswertung dieses Kriteriums sind geeignete Kostenmodelle sowohl für den Selektions- als auch für den ortsbasierten Index-Operator erforderlich, da letzterer indirekt zu den Werten $costs(P_i)$ beiträgt. Die jeweils verwendeten Modelle sollen daher nachfolgend kurz erläutert werden.

Kostenmodell des Selektionsoperators

Wie bereits in Abschnitt 2.4.4 dargelegt, sind die Kosten des Selektionsoperators maßgeblich durch die zur Auflösung der IDs in WKB-Strings notwendigen Dictionary-Aufrufe und die daraus resultierenden Seitenzugriffe bestimmt. Mittels Direct Mapping ([EGK95]) erfordert die Auflösung einer ID gemäß Abschnitt 2.3.2 jedoch maximal zwei Seitenzugriffe. Entsprechend können die Kosten des Selektionsoperators durch $costs(card(P_i)) = 2 \cdot card(P_i) \cdot costs_{page}$ modelliert werden. $costs_{page}$ seien hierbei die beim Einlesen einer beliebigen Seite aus dem Sekundärspeicher anfallenden Kosten. Hierfür ist in GeoRDF-3X bereits ein konkreter Wert vorhanden.

Kostenmodell des Spatial-Index-Lookup

Wie in Abschnitt 3.2 bereits dargelegt, werden in [TSS00] geeignete Kostenmodelle für den Zugriff auf R-Bäume diskutiert. Entsprechend ausgefeilte Modelle dürften im hier betrachteten Kontext jedoch kaum Vorteile mit sich bringen, da die Kostenabschätzungen maßgeblich durch oftmals sehr ungenaue Kardinalitätsabschätzungen bestimmt sind. Aus diesem Grund soll ein sehr viel einfacheres Modell Anwendung finden. Sei n die Anzahl aller Einträge des R-Baums und n_Q die Anzahl der durch eine Anfrage ausgewählten Einträge. Dann wird angenommen, dass in jeder Knotenebene des R-Baums Zugriffe auf $\frac{n_Q}{n}$ der jeweiligen Seiten notwendig sind. Dadurch soll die hierarchische Aufteilung des Datenraums in einem R-Baum möglichst einfach reflektiert werden.

4.2.2 Ansatz 2

Leider erwies sich der geschilderte Ansatz im Rahmen der Evaluation als untauglich. Insbesondere dann, wenn zur Kardinalitätsabschätzung auch oder ausschließlich die bereits zuvor in RDF-3X vorhandenen Mechanismen verwendet werden ergeben sich mitunter drastische Unterschätzungen. Diese bewegen sich nicht selten im Bereich 10^{-20} oder sogar darunter. Der Summand $costs_\sigma(card(P_2))$ fällt daher quasi nicht ins Gewicht. Da jedoch $P_2 \subset P_1$ gilt stets auch $costs(P_2) < costs(P_1)$ und damit quasi immer obige Ungleichung.

Entsprechend wird der Selektionsoperator fälschlicherweise auch dann angewendet, wenn dies nicht ratsam ist.

Vor diesem Hintergrund wurde ein zweites Entscheidungskriterium entwickelt. Dieses beruht auf der Beobachtung, dass der Selektionsoperator tendenziell dann effizient eingesetzt werden kann, wenn ein Großteil seiner Eingabetupel dem jeweiligen Filterprädikat genügt. Für kleine Eingabemengen ist dies naheliegend, da insgesamt nur wenige Tupel zu betrachten und somit auch nur wenige Seitenzugriffe notwendig sind, die insgesamt kaum ins Gewicht fallen. Bei großen Eingabemengen muss auch der alternativ in Frage kommende ortsbasierte Index auf viele Seiten des zu Grunde liegenden R-Baums zugreifen, so dass seine Verwendung keine Vorteile mit sich bringt. Als Entscheidungskriterium wurde daher $\frac{card(P_1)}{card(P_2)} > \phi$ gewählt. Zwar sind auch hier potentiell schlechte Kardinalitätsabschätzungen involviert, doch sollten diese in etwa um den gleichen Faktor fehlerhaft sein. Entsprechend dürfte ihr *Verhältnis* in etwas die realen Gegebenheiten widerspiegeln. Da sich angesichts der ungenauen Abschätzungen eine allzu reale Interpretation des Verhältnisses verbietet, wurde der Faktor ϕ empirisch bestimmt. Der Wert $0,5$ bewirkte hierbei größtenteils gewinnbringende Entscheidungen des Optimierers und wurde daher für die Evaluation in Kapitel 6 verwendet.

Erweiterung zur Unterstützung mehrerer Filter

Abschließend soll nun noch eine mögliche Erweiterung des Ansatzes für eine beliebige Zahl ortsbasierter Filter angegeben werden. Sei hierzu wie oben $T = \{T_1, T_2, \ldots, T_k\}$ die Menge der durch Tripelmuster spezifizierten Teilprobleme. Sei darüber hinaus $G = \{g_1, \ldots, g_l\}$ die Menge der ortsbasierten Filter. Dann kann durch den Aufruf decideFilters(T, G) der entsprechenden rekursiven Funktion aus Abbildung 4.4 ein geeigneter Plan erzeugt werden.
Da die Filter hierbei in einer zufälligen, aber festen, Reihenfolge betrachtet werden, liefert ein solcher Ansatz nicht zwingend optimale Pläne. Durch eine veränderte Reihenfolge wirken unter Umständen andere Selektionen auf die *Eingabemengen* der Filter. Entsprechend besitzen diese potentiell veränderte Selektivitäten, die zu abweichenden Entscheidungen führen können, so dass in der Folge unter Umständen bessere Pläne erzeugt werden. Im Mittel sollte der Ansatz jedoch tendenziell gute, wenn auch nicht unbedingt optimale, Pläne liefern, da innerhalb der gewählten Reihenfolge ausschließlich optimale Entscheidungen getroffen werden. Eine derartige Einschränkung fällt jedoch nicht weiter ins Gewicht, da durch die Eingrenzung des Selektionsoperators auf Positionen an der Wurzel von Operatorbäumen ebenfalls möglicherweise optimale Pläne ausgeschlossen werden.
Da die Unterstützung einer beliebigen Zahl ortsbasierter Filter nicht Teil der Aufgabenstellung war, wurde der dargestellte Ansatz zwar implementiert, jedoch nicht evaluiert. Entsprechend ist er lediglich als Idee und erste, grobe Annäherung an die Problematik zu verstehen.

1: **procedure** DECIDEFILTERS(SetOfPartialProblems $P = \{p_1, ..., p_k\}$, SetOfSpatialFilters $G = \{g_1, ..., g_l\}$)
2: select $g \in G$;
3: **if** $\frac{card(P \cup \{g\})}{card(P)} < 0.2$ **then**
4: Plan $p \leftarrow$ DECIDEFILTERS($P \cup \{g\}$, $G \setminus \{g\}$)
5: **else**
6: Plan $p' \leftarrow$ DECIDEFILTERS(P, $G \setminus \{g\}$)
7: Plan $p \leftarrow$ SETONTOP($SpatialSelection(g)$, p')
8: **end if**
9: **return** p
10: **end procedure**

Abbildung 4.4: Prozedur decideFilters

5 Implementierung des Lösungsansatzes

Auf Basis des vorhandenen Systems RDF-3X [BNM10] wurde der in Kapitel 4 vorgestellte Lösungsansatz wie nachfolgend dagestellt implementiert.

5.1 Statistikerhebung

5.1.1 Zerlegung in Kacheln

Zur Statistikerhebung werden zunächst - angelehnt an das Vorgehen von PostGIS [Posb] - in der Prozedur `RawGeoFileReader::detectExtent` sämtliche Geo-Literale durchlaufen und deren minimales, achsenparalleles, einhüllendes Rechteck bestimmt. Dieses wird anschließend in eine vom Benutzer anzugebende Zahl an Kacheln unterteilt. Auf diese Weise kann eine optimale Ausnutzung der zur Verfügung stehenden Kacheln erzielt werden. Die Geo-Literale werden in der Prozedur `RawGeoFileReader::fillCells` anschließend ein weiteres Mal durchlaufen und ihren jeweiligen Kacheln zugeordnet, die durch temporäre Dateien auf Sekundärspeicher verwaltet werden. Zusätzlich werden die Dictionary-IDs aller Geo-Literale in einer sortierten Liste im Primärspeicher vorgehalten. Ausgehend von dieser ersten Liste werden sämtliche zu den Geo-Literalen inzidenten Kanten bestimmt und in der temporären Datei der jeweiligen Kachel abgelegt. Parallel werden die entlang derartiger Kanten erreichbaren Knoten in einer neuen, sortierten Liste vermerkt. Anhand dieser Liste können im nächsten Schritt die zu den bereits bekannten Kanten adjazenten Kanten bestimmt werden. Das beschriebene Vorgehen realisiert also eine Breitensuche auf dem betrachteten RDF-Graphen. Die Tiefe der Breitensuche ist dabei durch die maximale Tiefe der zur Statistikerhebung betrachteten Teilgraphen bestimmt und somit ein Verfahrensparameter.
Aufgrund der Sortierung der Listen können zur Bestimmung inzidenter Kanten die ohnehin bereits in RDF-3X vorhandenen Index-Strukturen in effizienter Weise verwendet werden, indem ein entsprechend sortierter B-Baum auf Blattebene durchlaufen wird. Durch Nutzung der für das Sideways Information Passing (siehe Abschnitt 2.3.5) implementierten Sprungfunktionalität können dabei nicht in der Liste vorhandene IDs gezielt übersprungen werden, so dass weniger Seitenzugriffe notwendig sind.
Am Ende dieses Verarbeitungsschritts stehen temporäre Dateien, die sämtliche für die Statistikerhebung relevanten Kanten der jeweiligen Kachel enthalten.

5.1.2 Aufbau der temporären Tabelle

Während in den bisherigen Verarbeitungsschritten eine gemeinsame, globale Betrachtung sämtlicher Kacheln und ihrer enthaltenen Daten sinnvoll erschien, werden diese nachfolgend *einzeln* analysiert, um die dabei entstehenden Daten im Primärspeicher halten und somit möglichst effizient verarbeiten zu können. Darüber hinaus legt auch bereits die konzeptionelle Aufteilung in Kacheln ein solches Vorgehen nahe.
In einem ersten Schritt werden dabei in der Prozedur `CellBuilder::loadCell` sämtliche Kanten einer Kachel aus der temporären Datei eingelesen und in eine Inzidenzlisten-Darstellung der beteiligten Knoten überführt. Auf Basis dieser Darstellung startet `CellBuilder::computeCellStats` anschließend für alle in der Kachel enthaltenen Geo-Literale je eine Tiefensuche auf dem zuvor extrahierten und am jeweiligen Geo-Literal beginnenden Teilgraph. Auf diese Weise können sämtliche, an Geo-Literalen beginnende Pfade einer Kachel erzeugt und in eine temporäre Tabelle mit dem Relationenschema $R_1 = (edge_1, \ldots, edge_n, appearsAt, endSet, frequency)$ eingetragen werden. Die Attribute $edge_i$ beschreiben hierbei die Kanten des jeweiligen Pfades und dienen als Schlüssel. $appearsAt$ ist das bereits in Kapitel 4 eingeführte Bitfeld, *endSet* die Menge aller Endknoten von Pfaden dieses Typs und $frequency$ deren Anzahl. Entsprechend wird beim Einfügen eines Pfades das korrespondierende Bit des Geo-Literals in $appearsAt$ gesetzt, die Dictionary-ID des Endknotens in *endSet* eingefügt und $frequency$ um den Wert 1 erhöht. Anschließend wird mittels `TempSynTable::buildRanking` eine gemäß der in Kapitel 4 definierten Relevanz-Metrik sortierte Liste der Tabelleneinträge generiert.

5.1.3 Ausschreiben der Statistik in eine SQL-Datenbank

Da es im Rahmen des hier vorgestellten Prototyps ausschließlich darum geht, die Auswirkungen des Lösungsansatzes aus Kapitel 4 auf den resultierenden Ausführungsplan und dessen Kosten zu untersuchen, muss die vorangehende Optimierung nicht notwendigerweise besonders effizient sein. Aus diesem Grund werden die generierten Statistikdaten vollständig mittels einer SQLite-Datenbank [SQL] auf persistentem Speicher abgelegt. Hierdurch erübrigen sich Änderungen am ausgefeilten Dateiformat von RDF-3X.
Die Prozedur `SQLDumper::feedData` realisiert hierbei die Details der Übertragung nach SQLite. Die Daten werden dort in Form einer einzigen, großen Tabelle mit dem Relationenschema $R_2 = (cellID, edge_1, \ldots, edge_n, appearsAt, endSetCard, avgFrequency)$ abgelegt. $cellID$ gibt hierbei die ID der zugehörigen Kachel an. $edge_i$ und $appearsAt$ haben die selbe Bedeutung wie bereits im temporären Relationenschema. *endSetCard* bezeichnet die Kardinalität des Attributs *endSet* aus R_1 und *avgFrequency* ergibt sich aus $frequency$ mittels Division durch die Anzahl der Vorkommen, die wiederum durch Zählen gesetzter Bits in $appearsAt$ bestimmt werden kann.
Zusätzlich zu diesen Pfad-Daten werden darüber hinaus auch noch Kachel-spezifische Metadaten in einer Tabelle mit dem Relationenschema $R_3 = (cellID, ACT, featureCount)$ hinterlegt. $cellID$ ist hierbei wiederum die Dictionary-ID der Kachel, (Appearance Count Threshold) die minimale Zahl an Vorkommen eines Pfades, so dass der Eintrag mit Sicherheit

nicht verworfen wurde und *featureCount* die Anzahl der Geo-Literale in der Kachel. In einer weiteren Tabelle werden globale Kennzahlen der Statistik abgelegt. Hierzu gehören die maximale Pfadlänge, die Abmessungen des betrachteten Geo-Bereichs, die globale Anzahl an Geo-Literalen sowie die mittlere Anzahl von Geo-Objekten überdeckter Kacheln. Wie nachfolgend dargestellt, werden diese Informationen zur Anfrageoptimierung benötigt.

5.2 Nutzung der Statistik

5.2.1 Struktur des vorgefundenen Optimierers

Während die Implementierung der Statistikerhebung weitestgehend unabhängig vom bestehenden Code möglich war, erfordert die Integration der Kardinalitätsabschätzung eine weitaus engere Verwebung mit diesem. Daher sollen hier zunächst einmal relevante Implementierungsdetails des zu Grund liegenden Systems genauer dargestellt werden.
RDF-3X verwendet zur Anfrageoptimierung den Ansatz dynamischer Programmierung. Die Probleme erster Ordnung sind hierbei im Wesentlichen einzelne Tripelmuster, deren Lösung einem Scan über der Faktentabelle entspricht. Die vorhandenen Index-Strukturen können dabei genutzt werden, um Kardinalitätsabschätzungen der jeweiligen Resultatsmenge zu erhalten. Durch Kombination der Lösungen einfacher Probleme mit Hilfe von Join-Operatoren, werden anschließend mögliche Lösungen, d.h. optimale Ausführungspläne umfangreicherer Probleme bestimmt. Die Kosten eines solchen Ausführungsplans ergeben sich dabei mittels Operator-spezifischen Modellen aus den Kardinalitätsabschätzungen sowie Kosten der Teilprobleme. Seien c_1 und c_2 die Kardinalitäten der Teilprobleme, so ergibt sich die Kardinalität c des Gesamtproblems unter Einbeziehung eines mit dem jeweiligen Join assoziierten Selektivitätsfaktors σ zu $c = c_1 c_2 \sigma$. Der Wert von σ wird hierbei der in [NW09] beschriebenen Statistik entnommen.
Leider ist der in Kapitel 4 beschriebene Lösungsansatz *nicht unmittelbar* mit diesem Vorgehen kombinierbar, da er keine Selektivitätsabschätzungen für einzelne Join-Operationen, sondern Kardinalitätsabschätzungen für komplette Teilgraph-Muster liefert. Für Muster, die eine Kardinalitätsabschätzung mit Hilfe des Pfadbündel-Ansatzes erlauben, wird diese daher komplett unabhängig vom sonst üblichen Mechanismus durchgeführt.

5.2.2 Darstellung der Anfrage als Graph

Wie in [NM11] ausführlich dargelegt, sind zwei unterschiedliche, graph-strukturierte Darstellungen von Anfragemustern gegebener SPARQL-Anfragen denkbar, die jedoch in einer engen Beziehung zueinander stehen. RDF-3X verwendet zur Anfrageoptimierung die Join-Graph-Darstellung, da sie im Kontext des Dynamic-Programming-Ansatzes eine intuitivere Darstellung von Teilproblemen erlaubt. Eine SPARQL-Anfrage wird hierzu als Menge von Einschränkungen bzw. Relationen aufgefasst, die in einem Teilproblem entweder bereits berücksichtigt sind oder nicht. Ein Teilproblem lässt sich damit als Bitfeld B über der Menge aller Einschränkungen bzw. Relationen darstellen.

Der Pfadbündel-Ansatz hingegen erfordert eine Sicht entsprechend der RDF-Teilgraph-Darstellung. Aus diesem Grund wird zu Beginn der Anfrageoptimierung mit Hilfe der Klasse `RDFQueryGraph` zunächst zusätzlich eine entsprechende Darstellung generiert. Auf Basis dieser Darstellung kann mittels Tiefensuche für ein gegebenes Teilproblem die Menge aller Einschränkungen bzw. Relationen bestimmt werden, die von der in Kapitel 4 vorgeschlagenen Statistik abgedeckt werden. Eine solche Menge wird als Bitfeld B_{geo} dargestellt. Die Tiefensuche beginnt dabei an den Variablen der vorhandenen Spatial Filter und steigt in den Graph ein, bis die bei der Statistikerhebung maximal betrachtete Pfadlänge erreicht ist.

5.2.3 Kardinalitätsabschätzung

Die eigentliche Kardinalitätsabschätzung erfolgt in der Prozedur `PlanGen::createJoin`. Da der Pfadbündel-Ansatz - wie bereits dargelegt - ein abweichendes Verfahren zur Selektivitätsabschätzung erfordert, muss zunächst entschieden werden, ob dieses Anwendung finden kann. Letzteres ist genau dann der Fall, wenn sich der dem betrachteten Teilproblem entsprechende Teilgraph des Anfragegraphen ausschließlich aus Pfaden zusammensetzen lässt, die bei einer Geo-Variablen beginnen und die maximal betrachtete Pfadlänge nicht überschreiten, d.h. $B \subseteq B_{geo}$.

5.2.4 Betrachtung selektierter Kacheln

Ist dies der Fall, so erfolgt die Kardinalitätsabschätzung im Rahmen der Prozedur `GeoEstimator::getCardinality` in weiten Teilen analog zum entsprechenden Vorgehen in PostGIS [Posb]. Zunächst werden hierbei die vom Anfragegebiet partiell oder vollständig überdeckten Kacheln bestimmt. Um das Verfahren an dieser Stelle nicht unnötig kompliziert zu gestalten wurde hier anstelle des eigentlichen Anfragegebiets dessen minimales, einhüllendes Rechteck verwendet. Da im Zuge der Evaluation ausschließlich achsenparallele Rechtecke als Anfragegebiete Verwendung finden sollte diese Einschränkung jedoch keine Auswirkungen auf das Verfahren haben. Eine Modifikation des Verfahrens zur Unterstützung beliebiger Polygone sollte trivial zu implementieren sein, erfordert jedoch einen erhöhten Berechnungsaufwand. Durch die Einschränkung auf das minimale, einhüllende Rechteck genügt es, zur Bestimmung der überdeckten Kacheln die minimale bzw. maximale Kachel-Zeile bzw. -Spalte zu bestimmen. Für die so identifizierten Kacheln wird anschließend mittels der Prozedur `GeoEstimator::accountPaths` jeweils einzeln eine Kardinalitätsabschätzung unter Berücksichtigung der restlichen Anfragebestandteile vorgenommen und die Ergebnisse aufsummiert. Lediglich anteilig überdeckte Kacheln werden hierbei auch lediglich entsprechend ihrem überdeckten Anteil gewichtet. Da geometrische Objekte potentiell mehrere Kacheln überdecken können, fließen sie potentiell auch mehrfach in die Kardinalitätsabschätzung ein. Seien *coveredCells* die Anzahl der vom Anfragegebiet überdeckten Kacheln und *avgCoveredCells* die Anzahl durchschnittlich von einem geometrischen Objekt überdeckter Kacheln. Dann wird auch hier das bewährte Vorge-

hen von PostGIS [Posb] angewandt, und die resultierende Summe mittels Division durch $\min(coveredCells, avgCoveredCells)$ um besagten Effekt bereinigt.

5.2.5 Abschätzung innerhalb einer Kachel

Die Kardinalitätsabschätzung innerhalb einer einzigen Kachel geschieht im Rahmen der Prozedur `GeoEstimator::accountPaths` wie folgt: Per Tiefensuche werden alle bei der Geo-Variablen beginnenden Pfade des spezifizierten Anfrage-Graphen konstruiert. Erreicht ein Pfad einen bereits besuchten Knoten, ein als konstant spezifiziertes Subjekt oder Objekt oder aber die maximal betrachtete Pfadlänge, so wird für ihn die Prozedur `GeoEstimator::exploitCurrentPath` aufgerufen. Dort werden zunächst sämtliche Einträge für die aktuelle Kachel, die den zu untersuchenden Pfad als Präfix enthalten, aus der Statistik-Datenbank abgefragt. Diese Einträge repräsentieren sämtliche Vorkommen des Pfades in der betrachteten Kachel. Entsprechend werden ihre Bitfelder *appearsAt* in einem ersten Schritt disjunktiv verknüpft, um eine Auflistung sämtlicher Vorkommen zu erhalten. Die Ausgangskardinalitäten der verschiedenen Vorkommen werden aufsummiert, um später einen Mittelwert hieraus bilden zu können. Ebenso wird ein Mittelwert der durchschnittlichen Häufigkeiten des Pfades als Präfix verschiedener Einträge gebildet.
Da der untersuchte Pfad in qualifizierten Teilgraphen *zusammen* mit anderen auftreten muss, wird das disjunktiv aggregierte Bitfeld schlussendlich konjunktiv mit den entsprechenden Bitfeldern der anderen Pfade des Bündels verknüpft, um eine Auflistung aller Geo-Literale zu erhalten, die potentiell Bestandteil eines qualifizierten Teilgraphen sind.
Endet der gegenwärtig betrachtete Pfad an einer Variablen, so wird die Kardinalitätsabschätzung um den Faktor seiner durchschnittlichen Häufigkeit nach oben korrigiert, da sich gemäß der Semantik von SPARQL sämtliche Kombinationen der Pfade qualifizieren. Endet er hingegen an einer Konstanten und ist diese selektiver als alle bisherigen Konstanten, so wird die Selektivität entsprechend nach unten korrigiert.

Durch Abarbeiten sämtlicher Pfade des Bündels werden auf diese Weise aggregierte Werte für die Anzahl möglicher Kombinationen von Pfaden sowie eine minimale Selektivität im betrachteten Teilproblem spezifizierter Konstanten bestimmt.
Wurden sämtliche Pfade des Bündels in der Statistik gefunden, so ergibt sich die schlussendliche Kardinalitätsabschätzung für die betrachtete Kachel als Produkt der Anzahl gesetzter Bits im resultierenden Bitfeld mit der Anzahl möglicher Kombinationen sowie der minimalen Selektivität. Wurde jedoch mindestens ein Pfad nicht gefunden, so ergibt sich die Kardinalitätsabschätzung gemäß Kapitel 4 als ACT (vgl. oben) der entsprechenden Zelle und wird dem passenden Metadaten-Eintrag entnommen.
Sind im betrachteten Teilproblem keinerlei Pfade vorhanden, so qualifizieren sich alle Geo-Literale der Kachel und die Kardinalitätsabschätzung wird wiederum dem entsprechenden Metadaten-Eintrag der Statistik entnommen.

5.2.6 Integration in den bestehenden Optimierer

Da das beschriebene Verfahren statistische Abhängigkeiten innerhalb der betrachteten Pfade und damit potentiell über mehrere Joins hinweg berücksichtigt, sollte es für die überdeckten Bereiche nahe der Geo-Variablen zumindest im Mittel bessere Kardinalitätsabschätzungen liefern, als das in [NW09] beschriebene Standard-Verfahren von RDF-3X, das von statistischer Unabhängigkeit ausgeht. Aus diesem Grund wird bei der Anfrageoptimierung versucht, die Kardinalität möglichst weiter Teile des Anfragegraphen wie beschrieben mit Hilfe der Pfadbündel-Statistik abzuschätzen.
Da im Zuge des Dynamic-Programming-Ansatzes jedoch *sämtliche* möglichen Kombinationen von Teilproblemen zur Lösung eines Problems betrachtet werden, kann es vorkommen, dass die Teilprobleme alleine nicht den kompletten, von der Pfadbündel-Statistik überdeckten Bereich des Problems beinhalten. Aufgrund der Kardinalitätsabschätzung gemäß $c = c_1 c_2 \sigma$ für das zusammengesetzte Problem (vgl. oben) ergibt sich daraus unter Umständen eine suboptimale Abschätzung. Es existiert jedoch in jedem Fall mindestens eine Zerlegung des Problems in zwei Teilprobleme, so dass der von der Pfadbündel-Statistik überdeckte Bereich *vollständig* zur Kardinalitätsabschätzung verwendet wird. Aufgrund dieser Tatsache wurde im Rahmen der Implementierung des Pfadbündel-Ansatzes die Kardinalitätsabschätzung nicht mehr den verschiedenen Ausführungsplänen, d.h. Lösungen eines Problems, sondern dem Problem als solchem zugeordnet. Auf diese Weise ist es möglich, für *alle* Ausführungspläne eines Problems möglichst gute Kardinalitätsabschätzungen zu erhalten, indem letztere lediglich anhand *geeigneter* Kombinationen von Teilproblemen vorgenommen werden. Ein derartiges Vorgehen führt darüber hinaus zu einer intuitiveren Modellierung, da es sich bei der Kardinalität um eine Eigenschaft des Problems, und nicht seiner möglichen Ausführungspläne handelt.
Die Entscheidung darüber, welche Kombinationen verlässliche Abschätzungen erlauben, wird schlussendlich anhand der beiden mitgeführten Attribute `consideredGeoRelations` und `reachableGeoRelations` getroffen, die wiederum Bitfelder über der Menge sämtlicher Einschränkungen bzw. Relationen darstellen. Für eine verlässliche Abschätzung müssen alle *erreichbaren* Relationen auch tatsächlich *berücksichtigt* sein.

5.2.7 Kostenmodell des R*-Baums

Während für die Kardinalitätsabschätzung zahlreiche Änderungen am bestehenden Code notwendig waren, konnte die Kostenabschätzung vollständig übernommen werden. Notwendig war lediglich ein zusätzliches Kostenmodell für den Spatial-Index-Operator. Sei α das Verhältnis der Anzahl durch eine Anfrage qualifizierter Geo-Objekte zu deren Gesamtzahl. Dann wird angenommen, dass zur Abarbeitung des Spatial-Index-Operators auf eben diesen Anteil der Seiten aller Ebenen des zu Grunde liegenden R*-Baums zugegriffen werden muss.

5.3 Entscheidung zwischen Spatial-Index-Scan und Spatial-Selection

Für die Entscheidung, welcher der beiden zur Verfügung stehenden Operatoren zur Realisierung eines ortsbasierten Filters genutzt werden soll, sind gemäß des zweiten in Abschnitt 4.2 beschriebenen Entscheidungskriteriums lediglich die Kardinalitätsabschätzungen zweier Teilprobleme notwendig, die im Rahmen des Dynamic-Programming-Ansatzes ohnehin konstruiert werden. Die Implementierung des Entscheidungskriteriums ist daher mit Hilfe zweier Zugriffe auf die Problemtabelle des Optimierers in trivialer Weise möglich und wurde im Rahmen der Prozedur `PlanGen::reviseSpatialFilters` vorgenommen.

6 Evaluation der Leistungsfähigkeit

Die in Kapitel 5 beschriebene Implementierung soll nun einer ausführlichen Evaluation unterzogen werden, um die Leistungsfähigkeit des Lösungsansatzes aus Kapitel 4 bewerten zu können. Hierfür werden in Abschnitt 6.1 zunächst die verwendete Versuchsanordnung und anschließend in Abschnitt 6.2 die daraus resultierenden Ergebnisse dargestellt.

6.1 Versuchsanordnung

Dieser Abschnitt beschreibt die Details der zur Evaluation verwendeten Versuchsanordnung. Zunächst werden hierfür in Abschnitt 6.1.1 die Verfahrensparameter des Lösungsansatzes zusammengefasst und noch einmal kurz beschrieben. Anschließend skizziert Abschnitt 6.1.2 die zur Durchführung der Testreihen genutzte Rechnerkonfiguration. Die dabei verwendeten Testdaten werden in Abschnitt 6.1.3 beschrieben, während sich Abschnitt 6.1.4 den zugehörigen Beispielanfragen widmet. Als Abschluss werden in Abschnitt 6.1.5 geeignete Parameterwerte sowie Testreihen spezifiziert.

6.1.1 Verfahrensparameter

Der in Kapitel 4 vorgeschlagene und gemäß Kapitel 5 implementierte Lösungsansatz verwendet die nachfolgend beschriebenen Verfahrensparameter.

Anzahl der Kacheln h

Zur Unterteilung in Kacheln gemäß Abschnitt 4.1.1 ist eine Angabe der anzustrebenden Zahl an Kacheln erforderlich. Da es sich bei der Kachelung um eine Art Histogramm handelt, wird diese nachfolgend mit h bezeichnet. Naheliegenderweise können Korrelationen zwischen dem Ort und weiteren Attributen von Entitäten umso besser reflektiert werden, je feiner die Auflösung der Unterteilung, d.h. je größer die Anzahl verwendeter Kacheln ist.

Länge indexierter Pfade d

Gemäß Abschnitt 4.1.2 werden im Rahmen der Pfadbündel-Statistik ausschließlich bei Geo-Literalen beginnende Pfade einer maximalen Länge d (von engl. „distance“) betrachtet. Mit Hilfe dieses Parameters lässt sich festlegen, welche „Entfernung“ zum jeweiligen Geo-Literal Attribute maximal aufweisen dürfen, so dass etwaige Korrelationen in der Statistik berücksichtigt werden. Größere Werte sollten also mehr Korrelationen erfassen und damit genauere Abschätzungen erlauben. Allerdings steigt damit auch der notwendige Berechnungsaufwand rapide an!

Anteil berücksichtigter Pfade c

Gemäß Abschnitt 4.1.2 kann zur Reduzierung des benötigten Speichevolumens die Menge der Pfade einer Pfadbündel-Statistik $\overline{\mathcal{S}}$ auf einen Anteil c (von engl. „contingent“) der ursprünglichen Pfade begrenzt werden. Dieser Wert stellt somit einen weiteren Verfahrensparameter dar und wird nachfolgend in Prozent bezüglich der Kardinalität von $\mathcal{S}$ angegeben. Für sinkende Werte von c steigt offensichtlich der Anteil nicht auffinbarer Pfade, deren Häufigkeit somit gemäß Abschnitt 4.1.2 durch einen Standardwert angenähert werden muss. Entsprechend leidet die Genauigkeit der gesamten Abschätzung.

6.1.2 Rechnerkonfiguration

Zur Durchführung der Testreihen wurde eine Maschine vom Typ Dell Optiplex 755 mit Intel Core2 Quad Q9300 CPU bei einer Taktfrequenz von 2.5 GHz und 4 GB Hauptspeicher verwendet. Als Sekundärspeicher dienten zwei SATA-Magnetplatten mit jeweils 250 GB Kapazität und einer Datenrate von 3 GB/s bei 7200 Umdrehungen pro Minute, auf denen die Daten verteilt abgelegt waren. Als Betriebssystem wurde ein 64-bit Linux Kernel in der Version 2.6.31 verwendet.
Leider bestand keine Möglichkeit, Messungen auf *garantiert* kaltem Dateisystem-Cache durchzuführen. Um konsistente Resultate zu erzielen wurde daher ausschließlich mit warmem Cache gemessen.

6.1.3 Testdaten

Um die in Kapitel 5 beschriebene Implementierung unter realitätsnahen Bedingungen testen zu können, ist ein ausreichend großer Datensatz notwendig, der insbesondere auch reale ortsbasierte Daten beinhaltet. Es wurden daher unterschiedliche verfügbare Datensätze auf ihre Tauglichkeit hin untersucht und schlussendlich der des frei zugänglichen Projekts DBpedia [BLK+09, DBp] ausgewählt. Dieser basiert auf einer Momentaufnahme der semi-strukturierten Anteile von Wikipedia-Artikeln und verknüpft letztere mit Daten aus weiteren verfügbaren Quellen. Die Daten liegen bereits vollständig in Form von RDF-Tripeln vor, so dass sich eine aufwendige Konvertierung nach RDF erübrigt. Der Datensatz

beinhaltet in etwa 311.000.000 Tripel sowie ca. 715.000 Entitäten mit zugehörigen geographischen *Punktdaten*. Diese stellen jedoch nur einen Bruchteil der insgesamt modellierten Entitäten dar. Die Geo-Literale sind durch ein einheitliches Prädikat an ihre jeweiligen Entitäten gebunden und treten über die gesamte Erdoberfläche verteilt auf, so dass diese durch das Systems *vollständig* zu repräsentieren ist. Eine Konvertierung der Ortsdaten in die von GeoRDF-3X benötigte Form war auf Basis simpler Zeichenkettenmanipulationen möglich.
Wird lediglich eine Unterteilung in Kacheln vorgenommen und die Indexierung von Pfaden durch die Wahl des Parameterwerts $d = 0$ de facto deaktiviert, so ist ein Aufbau der Statistik mit dem in Kapitel 5 beschriebenen Prototyp in wenigen Minuten möglich. Für den Wert $d = 2$ werden hingegen zwei bis drei Stunden benötigt. Allerdings sind bereits für den Aufbau der restlichen Datenstrukturen von RDF-3X ca. fünf Stunden notwendig. Die Generierung der Pfadbündel-Statistik ist demnach für $d = 2$ in vertretbarer Zeit möglich. Der Parameter h besitzt keinen nennenswerten Einfluss auf die benötigte Zeit; ebenso der Parameter c, da stets zunächst die vollständige Statistik $\mathcal{S}$ berechnet werden muss.
Bei ausschließlicher Unterteilung in Kacheln ist zur Speicherung der Statistik weniger als 1 MB erforderlich. Werden zusätzlich Pfadbündel indexiert, so steigt der Bedarf auf ca. 350 MB für $h = 1000$ kacheln und geht für $h = 64.000$ Kacheln auf knappe 200 MB zurück. Diese Entwicklung ist nachvollziehbar, da kleinere Kacheln weniger Geo-Objekte beinhalten und somit kürzere Bitfelder zur Repräsentation des Attributs A_j (vgl. Abschnitt 4.1.2 sowie Kapitel 5) verwenden können. Es sollte daher möglich sein, die Pfadbündel-Statistik *vollständig* im Hauptspeicher vorzuhalten. Die Planoptimierung könnte hiervon massiv profitieren, da für die häufigen Kardinalitätsabschätzungen keine teuren Sekundärspeicherzugriffe notwendig würden.

6.1.4 Testanfragen

Auf Basis der verwendeten und im vorherigen Abschnitt 6.1.3 beschriebenen Testdaten wurde außerdem ein Paket von zwölf Testanfragen konstruiert, die im Anhang zu finden sind. Dabei war das vorrangige Ziel, möglichst viele der potentiell auftretenden Konstellationen in geeigneter Weise zu berücksichtigen. Aus diesem Grund wurde zunächst eine Klassifizierung möglicher Anfragen gemäß zweier orthogonaler Kriterien vorgenommen.

Das erstere hiervon unterscheidet selektive von weniger selektiven ortsbasierten Filterprädikaten. Deren Einordnung ist dabei im Wesentlichen durch die Größe ihres jeweiligen Anfragefensters bestimmt. Wird zur Realisierung des Filters ein ortsbasierter Index-Scan herangezogen, so sollte dieser für selektive Prädikate *tendenziell* bereits weit unten im Operatorbaum platziert werden, um möglichst frühzeitig nicht qualifizierte Tupel zu eliminieren. Für weniger selektive Prädikate hingegen bietet sich eine Position oberhalb selektiverer Operatoren an.
Das zweite Kriterium unterscheidet selektive von weniger selektiven Tripelmustern. Aus Sicht eines etwaigen Selektionsoperators bewirken selektive Tripelmuster eine vergleichsweise kleine Menge von Eingabetupeln, weniger selektive hingegen eine vergleichsweise große. In ersterem Fall *könnte* der Selektionsoperator also unter Umständen gewinnbringend eingesetzt werden, in letzterem eher nicht.

	wenig selektive Tripelmuster	selektive Tripelmuster
wenig selektives Filterprädikat	Q1, Q2	Q3, Q4, Q5, Q6
selektives Filterprädikat	Q7, Q8	Q9, Q10, Q11

Tabelle 6.1: Klassifikation der Testanfragen Q1 bis Q11

An dieser Stelle sei bemerkt, dass die Unterteilung der Anfragen gemäß der beschriebenen Klassifikation ein Stück weit Spekulation darstellt, da sie auf semantischen Interpretationen der Anfragen und entsprechenden Annahmen über die Daten beruht, die so nicht zwangsläufig zutreffen müssen! Des Weiteren stellt sie eine überaus grobgranulare Klassifikation dar und berücksichtigt nicht möglicherweise komplexe Zusammenhänge innerhalb einer Anfrage. Die resultierenden Ausführungspläne können daher durchaus von der erwarteten Form abweichen. Aus diesem Grund werden für jede der vier resultierenden Klassen verschiedene Anfragen untersucht. Ungeachtet der genannten Einschränkungen ermöglicht eine derartige Klassifikation die *strukturierte* Untersuchung der in Frage kommenden Konstellationen.

Tabelle 6.1 zeigt die Zuordnung der im Anhang einsehbaren Anfragen Q1 bis Q11 gemäß obiger Klassifikation.
Die Anfrage Q12 stellt einen Sonderfall dar. Sie enthält einen Zyklus und soll zeigen, dass der Ansatz gemäß Abschnitt 4.1.2 *grundsätzlich* auch zur Behandlung derartiger Strukturen geeignet ist. Selbstverständlich verbietet sich in diesem Fall eine Verallgemeinerung der Messergebnisse. Vielmehr bedürfte die Behandlung derartiger Anfragen einer darauf fokussierten und detaillierten Evaluation, was jedoch über den Rahmen dieser Arbeit hinausgeht. Da der Ansatz gemäß Abschnitt 4.1.2 auch zur Kardinalitätsabschätzung für Anfragen mit variablen Prädikaten geeignet ist, war darüber hinaus ursprünglich eine entsprechende Anfrage vorgesehen. Leider ist jedoch bereits das unmodifizierte System RDF-3X nicht in der Lage, diese auszuwerten.

Die vergleichsweise großen Anfragefenster sind bewusst gewählt und reflektieren die relativ geringe Dichte an Geo-Objekten in den verwendeten Testdaten sowie die relativ großen Kacheln. Dies dient dem Nachweis der Skalierbarkeit des Verfahrens für Anwendungen mit detailreicherer Modellierung. Denkbar wären hier unter anderem (Geo-)Informationssysteme im urbanen Umfang. Bleibt die Anzahl der Geo-Objekte pro Kachel in etwa konstant und werden die Kachelgrößen den entsprechend kleineren Anfragefenstern angepasst, so sollten damit die hier erzielten Resultate auf den geschilderten Fall übertragbar sein.

6.1.5 Auswahl und Gruppierung durchzuführender Versuche

Auf Grundlage der in den vorangegangenen Abschnitten beschriebenen Verfahrensparameter, Testdaten und Testanfragen wurden nachfolgende Überlegungen angestellt und Entscheidungen hinsichtlich der durchzuführenden Versuche getroffen.

Das Ziel dieser Arbeit besteht in einer Reduzierung der zur Anfragebearbeitung notwendigen Zeit. Hierzu ist die Genauigkeit der Kardinalitätsabschätzung zu verbessern, um in

der Folge *sinnvolle* Optimiererentscheidungen bei der Wahl des jeweiligen Ausführungsplanes zu ermöglichen. Entsprechend sind sowohl Kardinalitätsabschätzungen, als auch Ausführungszeiten von Interesse und somit zu erfassen. An dieser Stelle sei jedoch darauf hingewiesen, dass die Kardinalitätsabschätzung und Ausführungszeit ein und der selben Anfrage in keinem unmittelbaren Verhältnis zueinander stehen! Vielmehr ist der gewählte Plan und damit die Ausführungszeit einer Anfrage durch die Kardinalitätsabschätzungen für deren Teilprobleme bestimmt! Eine Betrachtung sämtlicher Teilprobleme der Anfragen würde jedoch den Rahmen dieser Arbeit sprengen. Aus diesem Grund sollen die Kardinalitätsabschätzungen ohnehin untersuchter Anfragen bestimmt, und in ihrer Gesamtheit als *Indikator* für die „globale" Qualität der Kardinalitätsabschätzung interpretiert werden.

Nach dieser Vorüberlegung hinsichtlich der zu erfassenden Messgrößen sollen nun geeignete Werte der Parameter d, c und h identifiziert werden.

Wird für den Parameter d der Wert 0 gewählt, so entspricht dies einer Deaktivierung der Pfadbündel-Statistik, indem Pfade der maximalen Länge 0 und somit überhaupt keine Pfade betrachtet werden. Gleichwohl findet eine Unterteilung in Kacheln statt und die Statistik-Komponente wird vom Optimierer genutzt, um Kardinalitätsabschätzungen für die Menge der von einem Spatial Index-Scan selektierten Geo-Objekte zu bestimmen. Ein solcher Wert kann also genutzt werden, um die Auswirkungen der Unterteilung in Kacheln *unabhängig* von möglichen Effekten der darauf aufbauenden Pfadbündel-Statistik zu analysieren.

Versuchsreihe 1

Entsprechend sollen in einer ersten Versuchsreihe mit $d = 0$ für sämtliche Anfragen Q1 bis Q12 und geeignete Werte der Parameter h beziehungsweise c die resultierenden Kardinalitätsabschätzungen bestimmt und mit denen des ursprünglichen Systems verglichen werden. Selbiges gilt gemäß obiger Überlegung für die Ausführungszeiten der Anfragen. Da für $d = 0$ keinerlei Pfade betrachtet und damit auch nicht in die Pfadbündel-Statistik $\mathcal{S}$ aufgenommen werden, ist der Parameter c in dieser Versuchsreihe irrelevant und kann auf einen beliebigen Wert gesetzt werden. Als einziger zu variierender Parameter verbleibt somit die Anzahl h der Histogramm-Zellen.
Offensichtlich wird der indexierte Bereich für kleine Werte von h in vergleichsweise große Kacheln unterteilt. Kleine Anfragefenster überdecken daher nur einen geringen Anteil einzelner Kacheln. Zur Abschätzung der Anzahl, durch ein Fenster selektierter Geo-Objekte, muss somit innerhalb einer Kachel von Gleichverteilung ausgangen und der Kachelinhalt anteilig gewertet werden. Da die Gleichverteilungsannahme im Regelfall jedoch eine Idealisierung darstellt, wird hierdurch ein Abschätzungsfehler induziert. Für $h = 64.000$ ist die verwendete Kachelgröße identisch mit der des kleinsten gemäß Abschnitt 6.1.4 auftretenden Anfragefensters. Auch dieses sollte daher *nicht* in der oben beschriebenen Weise zu Abschätzungsfehlern führen. Entsprechend dürften größere Werte von h nur noch geringfügige Verbesserungen der Kardinalitätsabschätzung bewirken. Der maximal zu betrachtende Wert des Parameters ist somit durch $h = 64.000$ gegeben. Gleichzeitig kommt der durch die Kachelung angestrebte Effekte für sehr kleine Werte von h offensichtlich zum Erliegen. Im

Rahmen der Versuchsreihe sollen daher Werte zwischen $h = 1.000$ und $h = 64.000$ betrachtet werden.

Gemäß Abschnitt 6.1.3 tritt in den verwendeten Testdaten nur ein einziges Prädikat zur Anbindung der Geo-Literale an ihre jeweiligen Entitäten auf. Des Weiteren besteht bei geographischen Daten üblicherweise nahezu eine 1:1-Beziehung zwischen Geo-Literal und zugehöriger Entität. Aus diesem Grund beinhaltet die unmittelbar an das Geo-Literal angrenzende Kante im RDF-Graph weder in Form ihres Prädikats, noch durch ihr jeweiliges Subjekt relevante Information für die Kardinalitätsabschätzung. Vielmehr ist für jedes Geo-Literal in der Regel exakt eine angrenzende Kante vorhanden. Aus diesem Grund ist für eine Pfadbündel-Statistik mit $d = 0$ kein über den Fall $d = 0$ hinausgehender Nutzen zu erwarten.

Versuchsreihe 2

Betrachtet man Pfade der Länge 2, so beinhalten diese durchaus relevante Information. In einer zweiten Versuchsreihe soll daher das Verhalten des Prototyps für $d = 2$ untersucht werden.
In diesem Fall ist es auch tatsächlich möglich, zum Zwecke der Komprimierung eine Teilmenge der zunächst in $\mathcal{S}$ indexierten Pfade zu verwerfen. Deren durch den Parameter c bestimmter Umfang ist somit erstmals von Bedeutung und soll daher variiert werden. Mangels Erfahrungswerten hinsichtlich der Wirkung dieses Parameters wird sein gesamter Wertebereich mit einer Schrittweite von 10% durchlaufen (der Fall $c = 0\%$ entspricht hierbei dem Fall $d = 0$ und ist somit nicht zu untersuchen). Sollte sich dabei ein interessantes Verhalten in einem beschränkten Wertebereich abzeichnen, so kann letzterer im Rahmen einer dritten Versuchsreihe unter Umständen genauer analysiert werden.
Der Parameter h schließlich soll auch hier, vor dem Hintergrund obiger Überlegungen, im Bereich zwischen $h = 1.000$ und $h = 64.000$ variiert werden. Mit Blick auf die Darstellbarkeit der Messdaten ist dabei jedoch eine geringere Anzahl unterschiedlicher Parameterwerte sinnvoll. Aus diesem Grund sollen lediglich für $h = 1.000$, $h = 8.000$ und $h = 64.000$ Messungen durchgeführt werden.

Für $d = 3$ ist zur Generierung der Statistik bereits ein immenser Rechenaufwand notwendig. Im Rahmen eines versuchweisen Datenimports waren zum Aufbau der Pfadbündel-Statistik von weniger als zehn Kacheln bereits mehrere Stunden notwendig! Dies resultiert aus der mit d exponentiell wachsenden Anzahl zu betrachtender Pfade. Für $d > 2$ ist der Ansatz daher, zumindest im Kontext der hier betrachteten Testdaten, nicht praktikabel.

Um belastbare Daten zu erhalten, soll für sämtliche Messgrößen, Anfragen und Parameterkombinationen der Median aus zehn Ausführungen bestimmt und im Rahmen der nachfolgenden Auswertung verwendet werden. Dabei ist die tatsächliche Ausführungszeit der Anfragen *ohne* den zur Optimierung benötigten Zeitaufwand zu messen!
Nachdem gemäß Abschnitt 6.1.2 auf der für die Versuche benutzten Maschine kein Mechanismus zur Anfragebearbeitung mit *garantiert* kaltem Cache zur Verfügung steht, sind die Messungen im Sinne konsistenter Ergebnisse allesamt auf warmem Cache durchzuführen.

Um diesen aufzuwärmen sollen die Anfragen für sämtliche betrachteten Wertekombinationen vor Beginn der eigentlichen Messung ein weiteres Mal ausgeführt werden.

6.2 Versuchsergebnisse

Nachdem im vorangegangenen Abschnitt die Verfahrensparameter, Testdaten sowie Testanfragen der durchgeführten Versuche beschrieben wurden, sollen hier nun die sich daraus ergebenden Messergebnisse dargestellt und diskutiert werden. Hierzu wird in Abschnitt 6.2.1 zunächst die Effektivität der Kachelung *ohne* die darauf aufbauende Pfadbündel-Statistik betrachtet. Anschließend widmet sich Abschnitt 6.2.2 der Analyse des vollständigen Ansatzes, bestehend aus Kachelung *und* Pfadbündel-Statistik.

6.2.1 Effektivität der Kachelung

Im Rahmen dieses Abschnitts sollen die Auswirkungen der Unterteilung in Kacheln anhand der Resultate aus Versuchsreihe 1 genauer untersucht werden. Abschnitt 6.2.1 beschreibt daher zunächst die zu beobachtende Verteilung der Geo-Objekte auf die unterschiedlichen Kacheln. Abschnitt 6.2.1 analysiert danach die Auswirkungen der Kachelung auf die Genauigkeit der Kardinalitätsabschätzung. Die Folgen dieser modifizierten Abschätzung auf die Ausführungszeiten der Testanfragen werden anschließend in Abschnitt 6.2.1 behandelt.

Verteilung der Geo-Objekte

Der gemäß Abschnitt 4.1.1 durch die Unterteilung in Kacheln beabsichtigte Effekt ist selbstverständlich umso ausgeprägter, je mehr die Geo-Objekte auf unterschiedliche Kacheln verteilt sind, da feingranulare Korrelationen hierdurch explizit repräsentiert und nicht durch die Annahme von Gleichverteilung „verwischt“ werden. Ebenso können leere, und damit im Endeffekt überflüssige, Kacheln je nach Implementierung möglicherweise die Effizienz des Ansatzes beeinträchtigen. Eine gleichmäßige Verteilung der Geo-Objekte auf die verschiedenen Kacheln erhöht daher in der Regel sowohl die Effektivität als auch die Effizienz des Ansatzes.
Leider ist eine solche Verteilung im betrachteten Fall nicht zu beobachten, wie Abbildung 6.1 für den Wert $h = 64.000$ exemplarisch dokumentiert (für die übrigen betrachteten Werte von h ergeben sich vergleichbare Verteilungen).

Offensichtlich bleibt ein Großteil der insgesamt 64.000 Kacheln leer. Wie anhand der logarithmischen Darstellung in Abbildung 6.2 zu erkennen ist, beinhalten gleichzeitig nicht wenige Kacheln eine jeweils beträchtliche Zahl an Geo-Objekten.
Beide Tatsachen resultieren aus einer ungeschickten Kombination der betrachteten Testdaten mit dem verwendeten, überaus simplen Unterteilungs-Schema. Die Geo-Objekte des Datensatzes erstrecken sich gemäß Abschnitt 6.1.3 über die komplette Erdoberfläche. Entsprechend ist diese vollständig im System zu modellieren. Gleichzeitig treten die Objekte

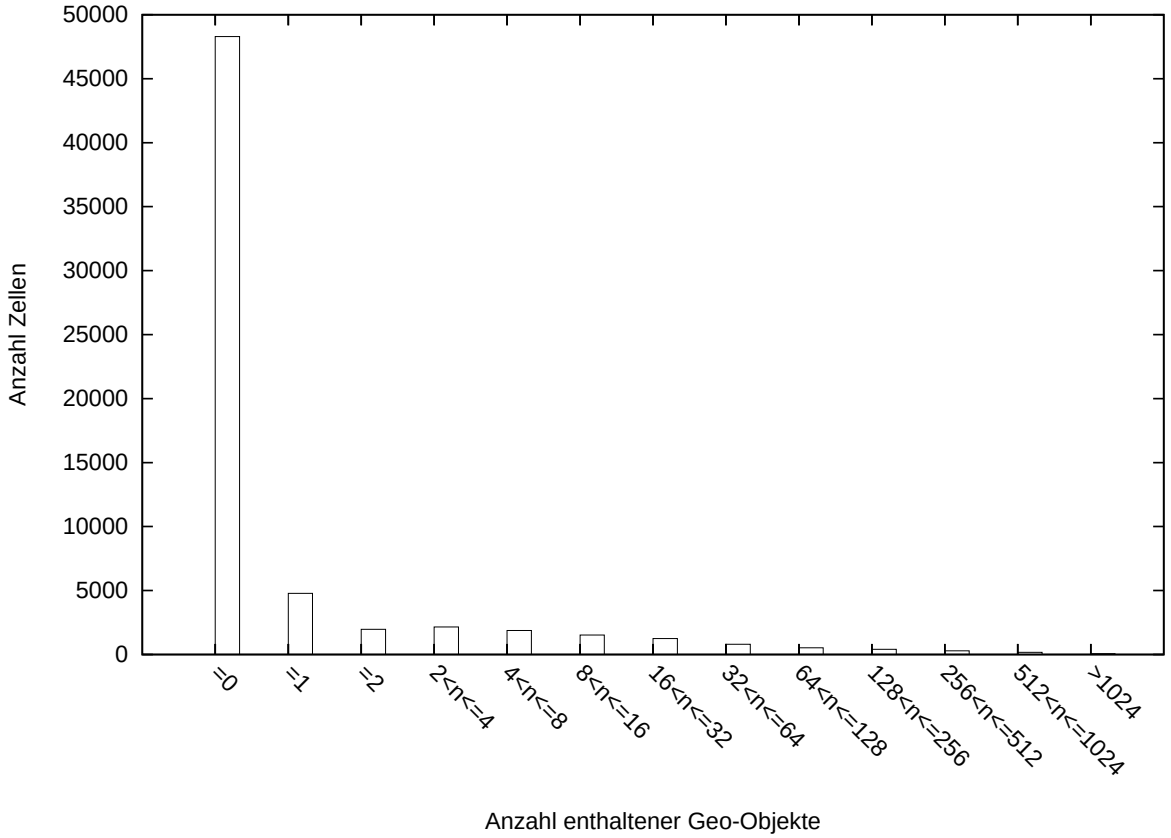


Abbildung 6.1: Verteilung der Geo-Objekte für $h = 64.000$, lineare Skala

im Bereich der Landmassen und insbesondere in dicht besiedelten Gebieten gehäuft auf. Diese Charakteristik steht jedoch in Diskrepanz zur starren Unterteilung der vollständigen Erdoberfläche gemäß Abschnitt 4.1.1.
Eine feinere Unterteilung der stark frequentierten Kacheln könnte entsprechend obiger Überlegung jedoch die Effektivität des Ansatzes verbessern. Gleichzeitig könnte eine weniger feine Auflösung der gering frequentierten Bereiche die Effizienz erhöhen. Zumindest für Datensätze mit der beschriebenen Charakteristik ist daher zukünftig eine hierarchische Unterteilung des modellierten Bereichs anzustreben. Entsprechende Ansätze existieren ([FFN+08, PHIS96, GKTD05]), sind jedoch mit einem erhöhten Implementierungsaufwand verbunden.

Genauigkeit der Kardinalitätsabschätzung

Es soll nun untersucht werden, wie die Aufteilung des modellierten Gebietes in Kacheln die Genauigkeit der Kardinalitätsabschätzung beeinflusst. Hierzu wurden für sämtliche Anfragen Q1 bis Q12 die jeweiligen Abschätzungen und tatsächlichen Kardinalitäten als Funk-

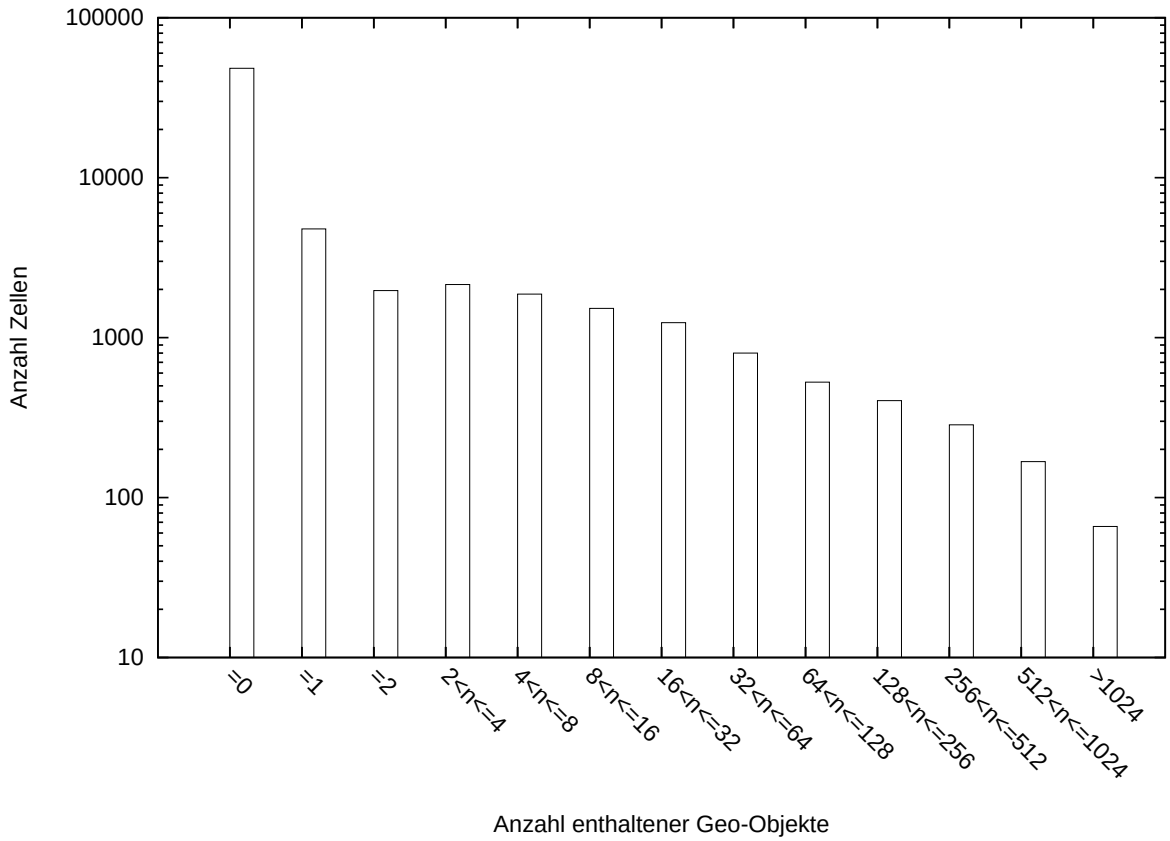


Abbildung 6.2: Verteilung der Geo-Objekte für $h = 64.000$, logarithmische Skala

tion über dem Parameter h bestimmt. Abbildung 6.3 zeigt exemplarisch das resultierende Schaubild für Q2. Als Vergleichsgröße ist hier außerdem die Kardinalitätsabschätzung des unmodifizierten Systems GeoRDF-3X abgebildet. Abbildung 6.4 zeigt mit einer etwas detaillierteren Auflösung noch einmal das Verhalten der Kardinalitätsabschätzung auf Basis der Kachelung. Offensichtlich variiert diese ohne erkennbares Muster innerhalb eines kleinen Bereichs, der einige Größenordnungen von der tatsächlichen Kardinalität entfernt ist. Die minimalen Variationen dürften dabei im Allgemeinen keinen Einfluss auf die Entscheidungen des Optimierers haben. Vielmehr sind letztere im Wesentlichen durch die *Größenordnung* der Abschätzung bestimmt. Die entsprechenden Schaubilder der übrigen Testanfragen zeigen ein äquivalentes Verhalten und wurden daher nicht explizit abgedruckt.

Die Kardinalitätsabschätzung auf Basis der Kachelung kann somit im untersuchten Wertebereich als weitgehend unabhängig von der Anzahl verwendeter Kacheln angesehen werden. Offenkundig liefert sie bereits bessere Werte als der entsprechende Ansatz des unmodifizierten Systems. Da sie letzteren zur Abschätzung der durch Tripelmuster bedingten Selektivitäten jedoch nach wie vor nutzt, liegen allerdings auch ihre Abschätzungen weit

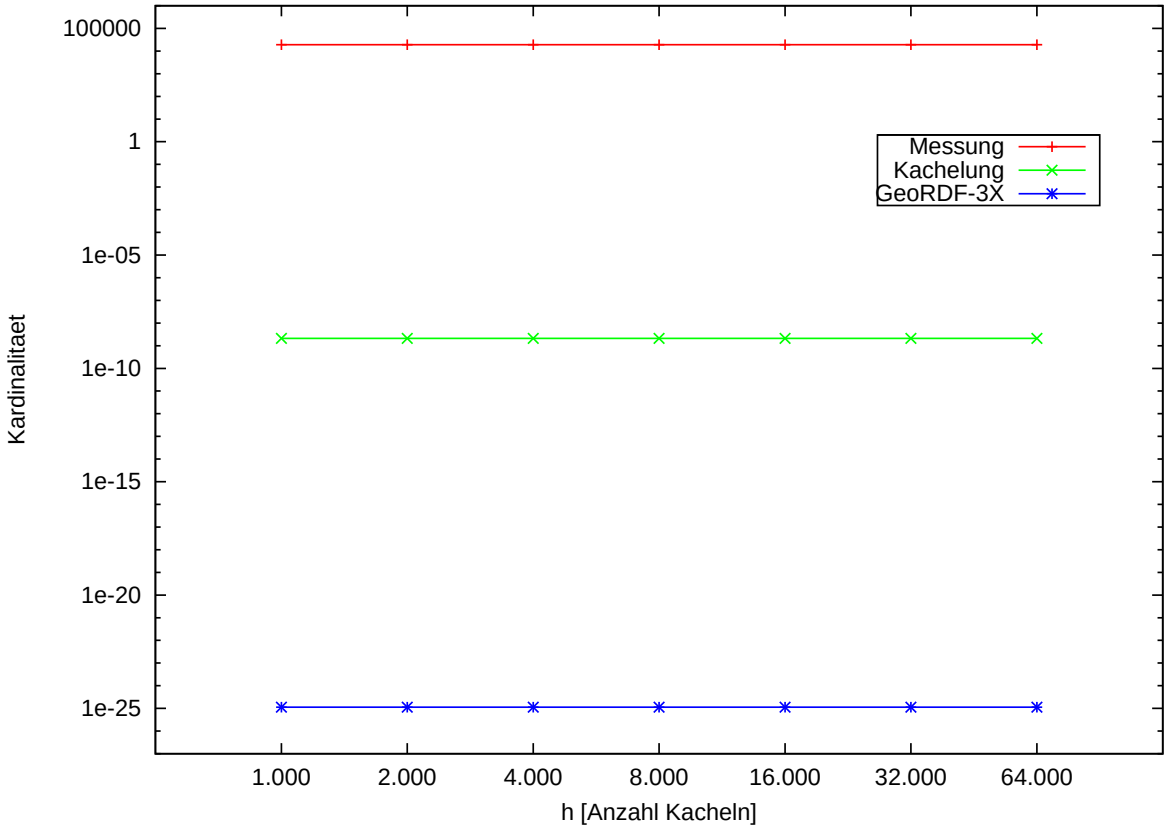


Abbildung 6.3: Kardinalitätsabschätzungen für Testanfrage Q2, $d = 0$

unterhalb der tatsächlichen Kardinalitäten. Eine weitere Verbesserung durch Nutzung der Pfadbündel-Statistik wäre daher gewinnbringend.

Auswirkung auf die Ausführungszeiten

In diesem Abschnitt soll nun die Auswirkung der Kachelung auf die Ausführungszeiten der Testanfragen untersucht werden. Hierfür wird zunächst einmal deren Abhängigkeit vom Wert des Parameters h untersucht, um anschließend einen Vergleich mit den Ausführungszeiten im unmodifizierten System GeoRDF-3X vorzunehmen.
Im Rahmen der durchgeführten Messungen wurden zu diesem Zweck die beobachteten Ausführungszeiten der Anfragen Q1 bis Q12 als Funktion über dem Parameter h bestimmt. Eine Analyse der hierbei gemessenen Werte ergab für sämtliche Anfragen lediglich Variationen im Bereich der Messungenauigkeit. Aus diesem Grund wurden die generierten und der Ausführung zu Grunde liegenden Pläne genauer untersucht. Dabei ergab sich, dass für sämtliche Anfragen außer Q10 und Q11 unabhängig von h stets identische Pläne generiert werden. Ein solches Verhalten ist nicht weiter verwunderlich, da die Kardinalitätsabschät-

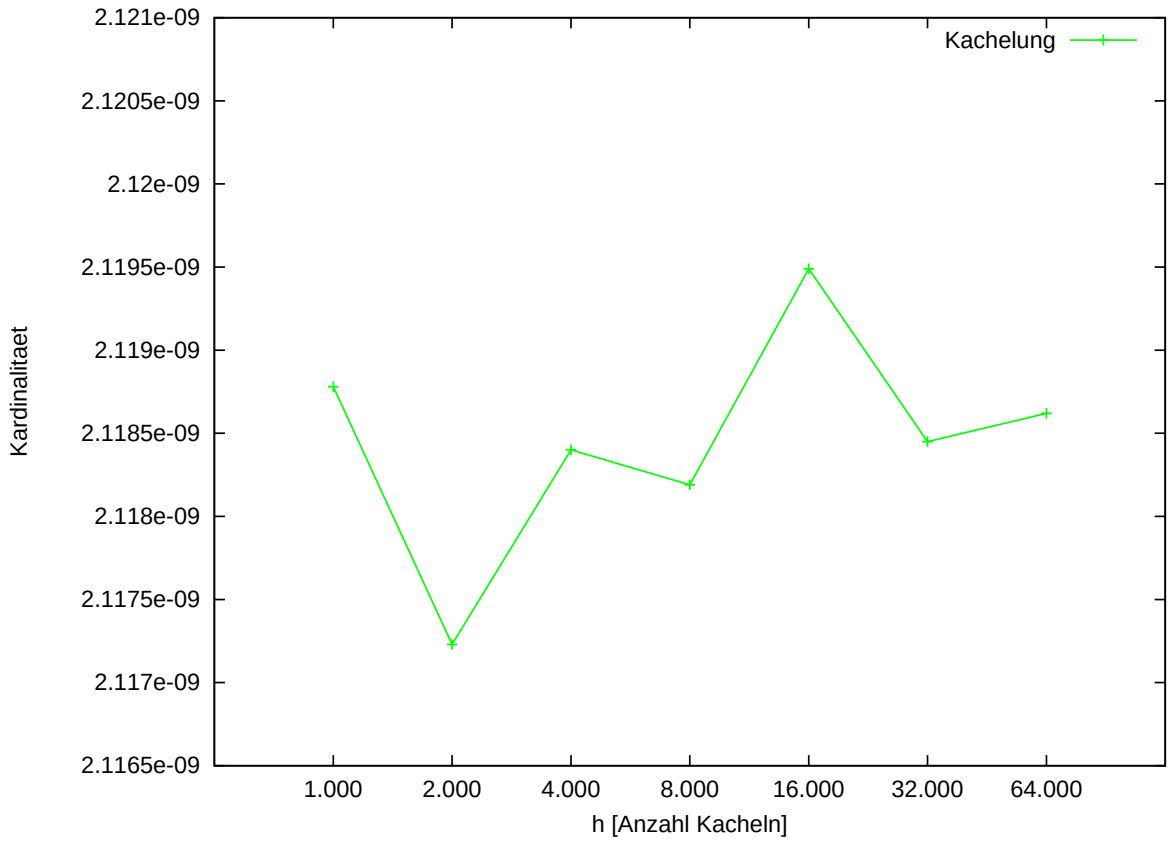


Abbildung 6.4: Kardinalitätsabschätzungen für Testanfrage Q2, $d = 0$, Detaildarstellung

zungen auf Basis der Kachelung gemäß Abschnitt 6.2.1 lediglich in einem sehr kleinen Bereich variieren. Entsprechend ist es vergleichsweise unwahrscheinlich, dass der Optimierer auf Basis dieser Abschätzungen unterschiedliche Entscheidungen trifft. Grundsätzlich ist dies, wie im Fall der strukturell ähnlichen Anfragen Q10 und Q11, aber trotzdem möglich. Vor dem Hintergrund des geschilderten Verhaltens können die Ausführungszeiten dieser Variante zum Zweck einer einfacheren Visualisierung zukünftig somit als konstant hinsichtlich des Parameters h betrachtet werden. Ohne Beschränkung der Allgemeinheit werden daher in Abbildung 6.9 die Ausführungszeiten für $h = 1.000$ exemplarisch denen des unmodifizierten Systems GeoRDF-3X gegenübergestellt. Es zeigt sich dabei eine moderate Verbesserung für die Anfragen Q3, Q4 und Q6. Einzig im Fall der Anfrage Q5 ermöglicht das unmodifizierte System eine effizientere Ausführung. Für die restlichen Anfragen ergibt sich durch die Kachelung noch keine Verbesserung ihrer Ausführungszeit. Die zu beobachtenden Differenzen im Schaubild stellen hierbei Messungenauigkeiten dar, wie eine Analyse der generierten Ausführungspläne ergab.
Insgesamt ermöglicht die Unterteilung des modellierten Gebiets in Kacheln also noch keine nennenswerte Verbesserung. Um eine solche zu erreichen, sollen im nachfolgenden Abschnitt daher zusätzlich die Effekte der Pfadbündel-Statistik berücksichtigt werden.

6.2.2 Effektivität der Pfadbündel-Statistik

Nachdem im vorherigen Abschnitt bereits die Effekte der Kachelung untersucht wurden, sollen hier nun die Auswirkungen der darauf aufbauenden Pfadbündel-Statistik betrachtet werden. Abschnitt 6.2.2 beschreibt dabei das Verhalten der Kardinalitätsabschätzung, während Abschnitt 6.2.2 die daraus resultierenden Veränderungen der Ausführungszeiten analysiert.

Genauigkeit der Kardinalitätsabschätzung

Zur Untersuchung der Genauigkeit der Kardinalitätsabschätzung sollen im Folgenden lediglich die Testanfragen Q1, Q2, Q3, Q4, Q7 und Q8 betrachtet werden. Die in ihnen enthaltenen und bei der Geo-Variablen beginnenden Pfade besitzen die maximale Länge 2. Entsprechend kann die Kardinalität der *vollständigen* Anfrage von einer Pfadbündel-Statistik mit $d = 2$ abgeschätzt werden. Auf diese Weise ist eine Analyse der Abschätzungsgenauigkeit *ohne* potentiell verfälschende Einflüsse der gewöhnlichen Statistiken von RDF-3X möglich.

Im Folgenden soll nun zunächst der Einfluss der Verfahrensparameter auf die Genauigkeit der Abschätzung untersucht werden. Anschließend wird diese mit der Genauigkeit des bislang in RDF-3X genutzten Mechanismus verglichen.

Da gemäß Abschnitt 6.1.5 nur für $d = 2$ eine sinnvolle Nutzung der Pfadbündel-Statistik möglich ist, sollen daher in einem ersten Schritt die Auswirkungen verschiedener Werte der Parameter h und c analysiert werden.
Abbildung 6.5 zeigt hierzu exemplarisch das Verhalten der Testanfrage Q1 im Rahmen der Versuchsreihe 2. Die Anfragen Q2, Q3 und Q4 weisen ein äquivalentes Verhalten auf und werden daher nicht explizit dargestellt. Die Werte des Parameters c werden in besagtem Diagramm entlang der horizontalen Achse variiert, die des Parameters h durch unterschiedliche Kurven im Schaubild dargestellt. Der Wert $c = 100\%$ entspricht hierbei dem Fall $\mathcal{S} = \overline{\mathcal{S}}$, der im Rahmen der Statistikgenerierung keine Pfade vernachlässigt. Für $c = 10\%$ hingegen werden nur 10% der ursprünglich betrachteten Pfade in die Statistik $\mathcal{S}$ übernommen. Offensichtlich sinkt die Genauigkeit der Abschätzung für kleinere Werte von c. Dies ist nicht weiter verwunderlich, da für kleine Werte von c zunehmend benötigte Pfade nicht in der Statistik gefunden werden. Gemäß Abschnitt 4.1.2 verwendet der Lösungsansatz in einem solchen Fall als Kardinalitätsabschätzung die *maximale* Häufigkeit vernachlässigter Pfade. Dies stellt jedoch eine Überschätzung dar, die sich für eine zunehmende Zahl solcher Fälle akkumuliert. Ein solches Verhalten ist für $h = 64.000$ offensichtlich, bei detaillierterer Darstellung aber auch für $h = 8.000$ sowie $h = 1.000$ zu erkennen.
Das vom beschriebenen Muster abweichende Verhalten der Testanfrage Q7 ist in Abbildung 6.6 dargestellt. Wie zu erwarten tritt der einzige in Q7 enthaltene Pfad vergleichsweise häufig auf, und wird daher beim Übergang von $\mathcal{S}$ nach $\overline{\mathcal{S}}$ auch für kleine Werte von c nicht vernachlässigt. Entsprechend kann auf dessen Grundlage stets die gleiche Abschätzung vorgenommen werden. Das ebenfalls abweichende Verhalten der Anfrage Q8 ist des Weiteren in Abbildung 6.7 dargestellt. Es unterscheidet sich vom Fall der Anfragen Q1 bis Q4 durch

die zunehmende *Unterschätzung* anstelle einer Überschätzung. Offenbar wird der einzige in Q8 enthaltene Pfad für sinkende Werte von c zunehmend in den unterschiedlichen Kacheln vernachlässigt. Entsprechend muss zur Kardinalitätsabschätzung gemäß Abschnitt 4.1.2 die maximale Häufigkeit vernachlässigter Pfade herangezogen werden. In einem solchen Fall werden jedoch keine durch variable Pfadenden induzierten Multiplizitäten berücksichtigt. Dies bewirkt im betrachteten Fall eine zunehmende Unterschätzung und stellt einen Entwurfsfehler dar. Dieser sollte in eventuellen zukünftigen Implementierungen geeignet behoben werden.
In allen drei betrachteten Fällen kann die Abschätzung für $h = 64.000$ ab ca. $c = 50\%$ als stationär angenommen werden. Bei $h = 8.000$ und $h = 1.000$ gilt dies bereits für kleinere Werte von c. Im Zuge des nachfolgenden Vergleichs mit alternativen Verfahren soll daher der Wert $c = 50\%$ angenommen werden. Dies ermöglicht einerseits eine übersichtlichere Darstellung, gleichzeitig aber auch einen objektiven Vergleich ohne idealisierte Verfahrensparameter.

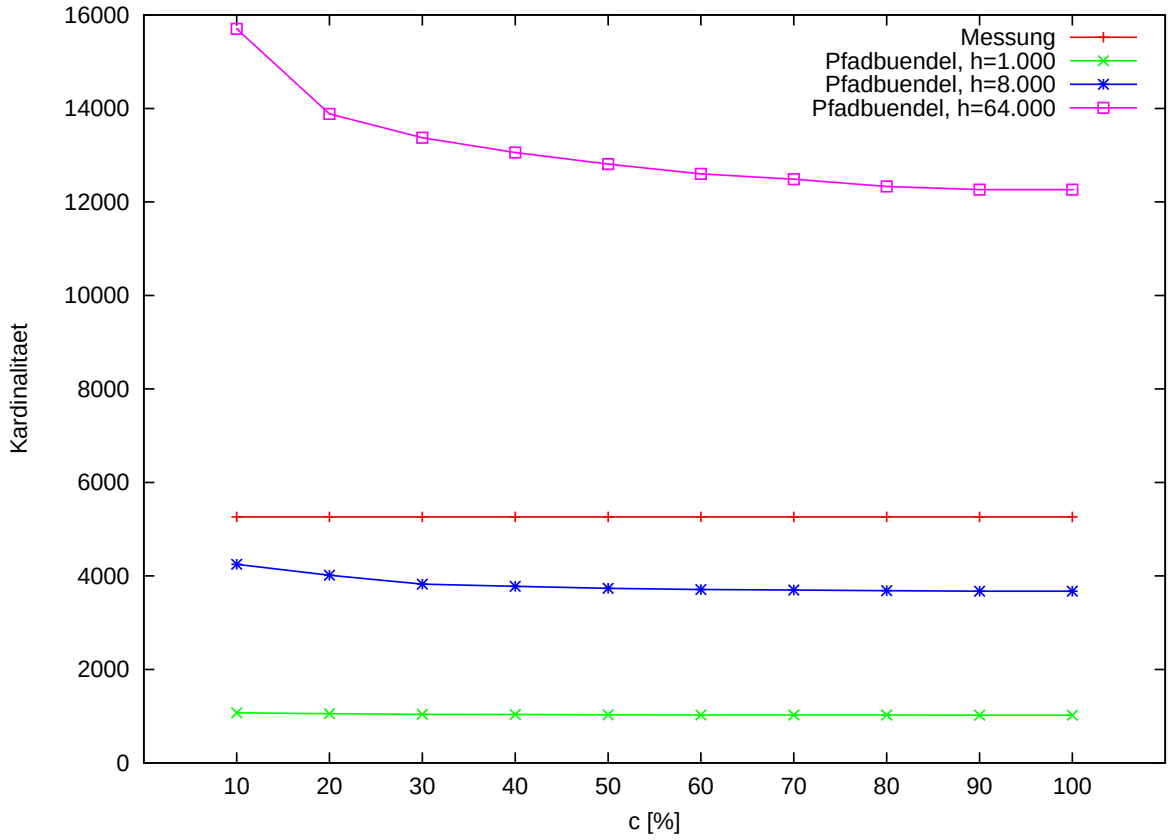


Abbildung 6.5: Kardinalitätsabschätzungen, Testanfrage Q1, $d = 2$

Die Genauigkeit der Kardinalitätsabschätzung auf Basis einer Pfadbündel-Statistik soll nun mit dem gewöhnlichen Abschätzungsmechanismus von RDF-3X verglichen werden. Abbildung 6.8 stellt hierzu für $c = 50\%$ die von den beiden Verfahren für die Anfragen Q1, Q2, Q3, Q4, Q7 und Q8 berechneten Werte dar. Des Weiteren wurden zum Vergleich die tatsäch-

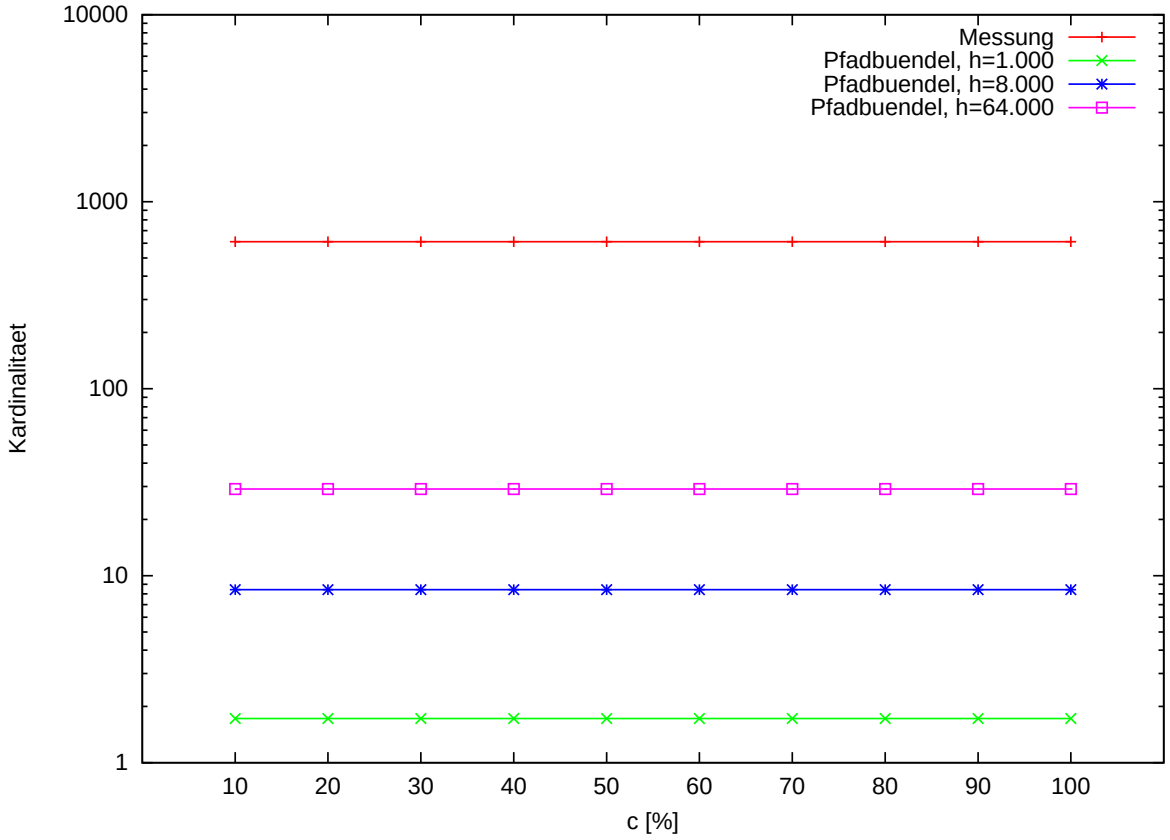


Abbildung 6.6: Kardinalitätsabschätzungen, Testanfrage Q7, $d = 2$

liche Kardinalität sowie die bereits in Abschnitt 6.2.1 genauer untersuchte Abschätzung mittels Kachelung für $h = 1.000$ (gemäß Abschnitt 6.2.1 kann dieser Wert repräsentativ für alle betrachteten Werte von h verwendet werden) in das Schaubild aufgenommen.
In Anbetracht der logarithmischen Skala führt offenbar bereits die Kachelung durchweg zu einer drastischen Verbesserung der Abschätzung. Kommt zusätzlich die Pfadbündel-Statistik zum Einsatz, so bewirkt dies in allen Fällen eine weitere Verbesserung um mehrere Größenordnungen. Über- und Unterschätzungen halten sich dabei in etwa die Waage. Zwar wäre zu erwarten, dass die Pfadbündel-Statistik für zunehmende Werte von h Korrelationen zwischen Attributen detaillierter rerpräsentieren und damit bessere Abschätzungen liefern kann, doch wird diese Vermutung durch die Testdaten nicht bestätigt. Vielmehr liefert das Verfahren für $h = 1.000$ im Mittel ähnlich gute Werte. Da im Zuge der Messungen für $h = 64.000$ mitunter enorme Optimierungszeiten von mehreren Minuten zu beobachten waren, ist daher der Parameterwert $h = 1.000$ zumindest im betrachteten Fall vorzuziehen.

Zur Relativierung der Darstellung sei an dieser Stelle bemerkt, dass die Entscheidungen des Optimierers *im Wesentlichen* durch die *Größenordnung* der Kardinalitätsabschätzungen bestimmt sind. Vor diesem Hintergrund sind die durch Nutzung einer Pfadbündel-Statistik

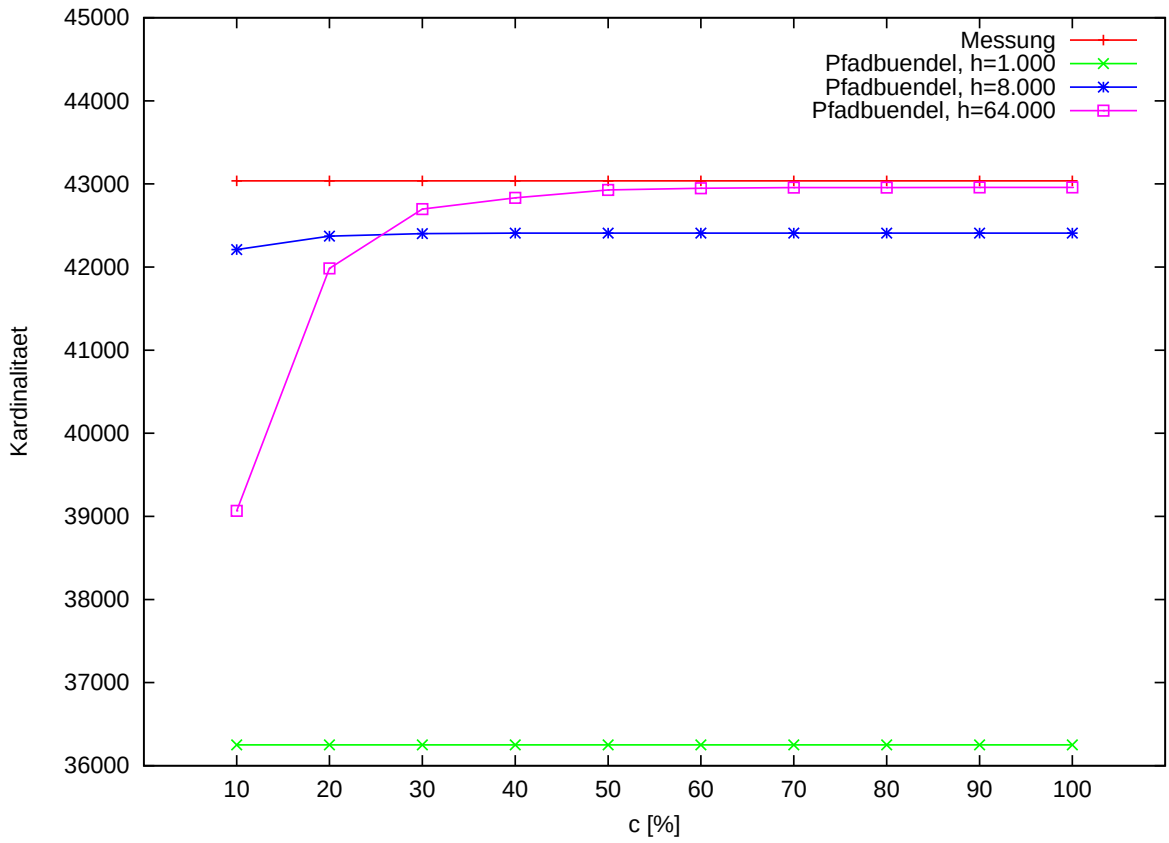


Abbildung 6.7: Kardinalitätsabschätzungen, Testanfrage Q8, $d = 2$

erzielbaren Verbesserungen der Abschätzungsgenauigkeit nicht in dem hohen Maße gewinnbringend, wie dies durch Abbildung 6.8 auf den ersten Blick vermeintlich impliziert wird. Gleichwohl liefert das Verfahren überaus erfreuliche Resultate und könnte insbesondere auch in anderen Anwendungskontexten gewinnbringende Verbesserungen der Abschätzungsgenauigkeit ermöglichen.

Auswirkung auf die Ausführungszeiten

Nachdem zuvor bereits die Auswirkungen einer Nutzung der Pfadbündel-Statistik auf die Genauigkeit der Kardinalitätsabschätzung betrachtet wurden, soll hier nun eine Untersuchung der entsprechenden Ausführungszeiten erfolgen.

Da die Pfadbündel-Statistik gemäß Abschnitt 6.1.5 nur für $d = 2$ eine sinnvolle Nutzung ermöglicht, ist auch hier lediglich dieser Wert zu betrachten. Eine detaillierte Analyse der Messergebnisse aus Versuchsreihe 2 zeigt darüber hinaus, dass die Ausführungszeiten der Testanfragen Q1 bis Q12 im Rahmen der Messgenauigkeit sowie der hier betrachteten Wertebereiche weitgehend unabhängig von den Werten der beiden verbleibenden Parameter h und

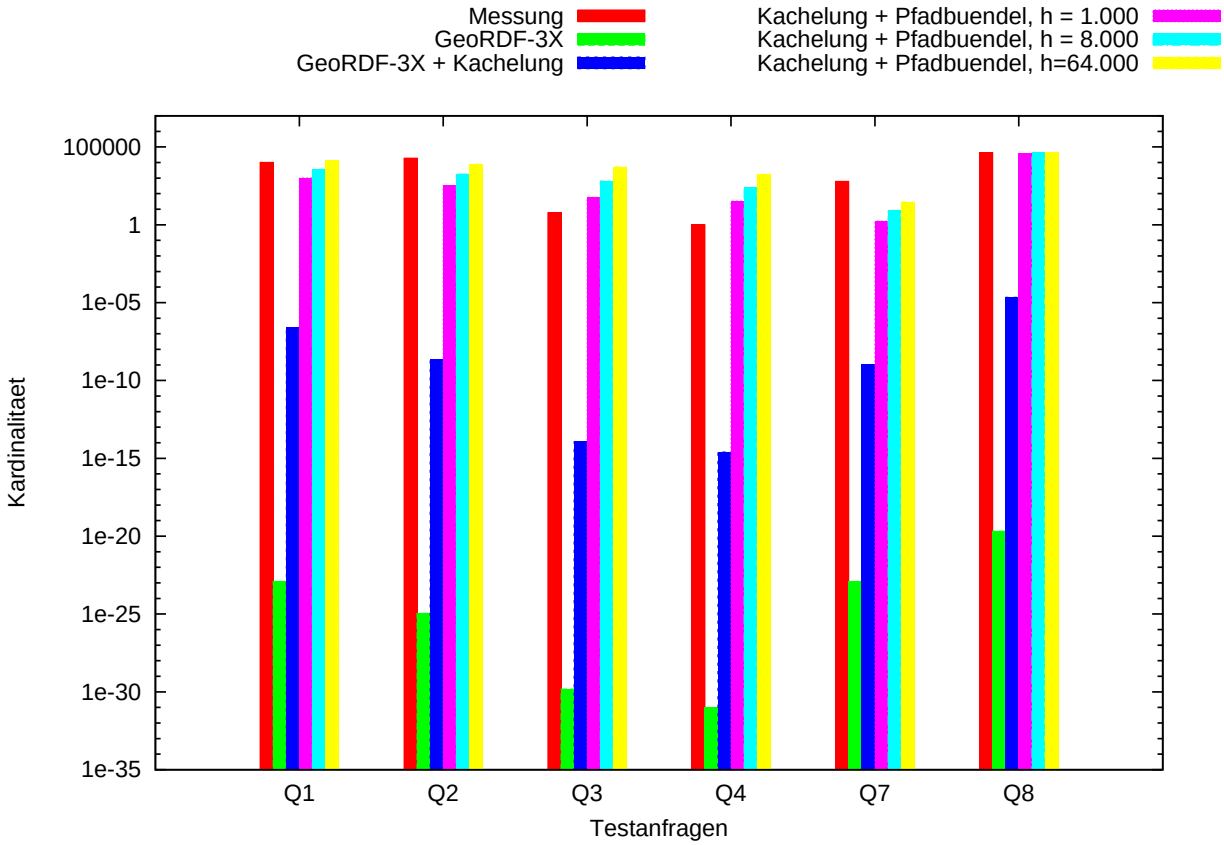


Abbildung 6.8: Kardinalitätsabschätzungen, $d = 2$, $c = 50\%$

c sind. Aufgrund der Erkenntnisse hinsichtlich optimaler Werte von h aus Abschnitt 6.2.2, wird daher für die weitere Analyse $h = 1.000$ verwendet.

Basierend auf diesen Überlegungen sollen im Folgenden die auf Grundlage einer Pfadbündel-Statistik erzielbaren Ausführungszeiten mit den entsprechenden Werten des unmodifizierten Systems GeoRDF-3X ([BNM10]) verglichen werden. Da erstere gemäß obiger Überlegung weitgehend unabhängig vom Wert des Parameters c sind, wird dieser zu $c = 10\%$ gewählt. Sollten sich dadurch gegenüber dem Vergleichspartner verbesserte Ausführungszeiten ergeben, so kann davon ausgegangen werden, dass dies für größere Werte von c (und dadurch tendenziell genauere Abschätzungen) ebenso der Fall ist. Damit sind nun alle Verfahrensparameter fest gewählt. Entsprechend können die resultierenden Ausführungszeiten für sämtliche Anfragen in einem gemeinsamen Diagramm zusammengefasst werden.
Dieses ist in Abbildung 6.9 dargestellt und beinhaltet analog zu Abschnitt 6.2.2 als zusätzlichen Vergleichskandidaten die ausschließlich auf Kachelung basierende Variante aus Abschnitt 6.2.1. Um sinnvolle Vergleiche zu ermöglichen wurde für diese ebenfalls der Wert $h = 1.000$ gewählt.

Offensichtlich ermöglicht die Kachelung alleine noch keine nennenswerten Verbesserungen der Ausführungszeiten. Im Falle der Anfrage Q5 steigt diese sogar leicht an. Die vermeintlichen Zuwächse bei den Anfragen Q7 und Q8 stellen jedoch Messungenauigkeiten dar. Eine Analyse der generierten Ausführungspläne zeigt, dass die Kachelung hier keine Planänderungen bewirkt.
Erst die zusätzliche Nutzung der Pfadbündel-Statistik ermöglicht im Mittel eine deutliche Reduzierung der Ausführungszeit. Die für Q3 und Q4 ermittelten Werte sind dabei so klein, dass sie in der gewählten Skalierung nicht mehr dargestellt werden können. Maßgeblich für besagte Reduzierung der durchschnittlichen Ausführungszeit sind die Anfragen Q1 bis Q5. Die verbesserten Abschätzungen bewirken hier in erster Linie ein „Umschalten“ auf den Selektionsoperator, der in diesen Fällen offensichtlich dem Spatial-Index-Lookup überlegen ist und somit sinnvoll genutzt werden kann.
Die Anfrage Q8 produziert aufgrund des variablen Typs der spezifizierten Tupel eine überdurchschnittlich große Ausgabemenge (> 40.000). Vermutlich resultiert ihre vergleichsweise lange Ausführungszeit bereits aus dieser inhärenten Charakteristik und kann somit auch durch verbesserte Kardinalitätsabschätzungen nicht deutlich reduziert werden. Der vermeintliche Zuwachs bei Q7 ist erneut der Messungenauigkeit geschuldet, da der generierte Ausführungsplan mit dem der beiden Vergleichspartner übereinstimmt. Im Falle der Anfrage Q9 hingegen werden tatsächlich unterschiedliche Pläne erzeugt. Nachdem sich die Differenz ihrer Ausführungszeiten jedoch im Bereich der bei Q7 ersichtlichen Messungenauigkeit bewegt, kann sie vernachlässigt werden. Die Pläne sind daher als gleichwertig zu betrachten.
Für die Anfragen Q10 bis Q12 schließlich werden bei Nutzung der Pfadbündel-Statistik moderat verbesserte Ausführungszeiten erzielt. Dies resultiert aus einer (auch intuitiv sinnvollen) Optimierung der Join-Reihenfolgen, die aufgrund der verbesserten Kardinalitätsabschätzungen möglich wurde. Die Realisierung des ortsbasierten Filters wird dabei jedoch nicht modifiziert. Zur Illustration dieses Sachverhalts sind in den Abbildungen 6.10 und 6.11 der ursprüngliche sowie der optimierte Ausführungsplan der Anfrage Q11 dokumentiert. Die Darstellung ist in beiden Fällen an [NW10] angelehnt.
Insgesamt treten also keine nennenswerten Verschlechterungen auf. Vielmehr bewirkt die Pfadbündel-Statistik in mehreren Fällen sogar deutliche Verbesserungen der Ausführungszeit.

6.3 Diskussion

Die Variation von Join-Reihenfolgen in den Anfragen Q10 bis Q12 erbrachte nur geringfügige Verbesserungen der Ausführungszeit. Die *maßgeblichen* Verbesserungen entstanden durch geschickte Wahl des Selektionsoperators. Entsprechend ist die Entscheidung zwischen Spatial-Index-Lookup und Selektionsoperator offenbar von zentraler Bedeutung für die Leistungsfähigkeit des Systems. Diese Tatsache motiviert die nachfolgenden Überlegungen.

Die gewinnbringenden Entscheidungen hinsichtlich der Verwendung eines Selektionsoperators in den Anfragen Q1 bis Q5 (und *nur* in diesen!) bestätigen das in Abschnitt 4.2 gewählte

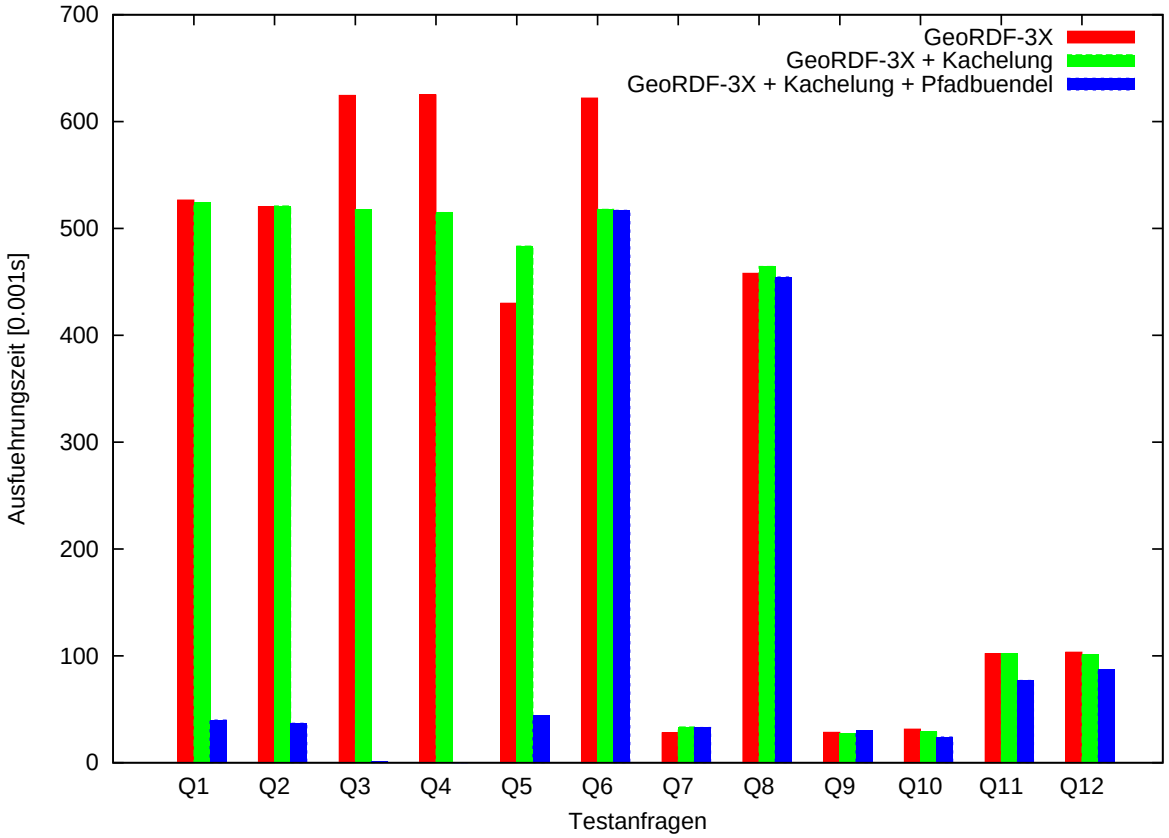


Abbildung 6.9: Ausführungszeiten, $d = 2$, $h = 1.000$, $c = 10\%$

Entscheidungskriterium. Gleichwohl sollte dessen Stabilität, im Hinblick auf seine zentrale Bedeutung, einer ausführlicheren Analyse unterzogen werden. Unter Umständen wäre auch die Entwicklung eines ausgefeilteren Kriteriums lohnenswert.

Obwohl die Selektionsoperatoren der optimierten Ausführungspläne für Q1 und Q2 jeweils weit mehr als 10.000 Tupel überprüfen müssen, können die Anfragen vergleichsweise schnell ausgeführt werden. Dies ist nur möglich, wenn massive Clustering-Effekte vorliegen oder die benötigten Seiten bereits im Cache vorhanden sind. Besagte Effekte haben also einen deutlichen Einfluss auf die Effizienz des Selektionsoperators und sollten daher unter Umständen zukünftig bei der Entscheidung zwischen Spatial-Index-Lookup und Selektionsoperator berücksichtigt werden.

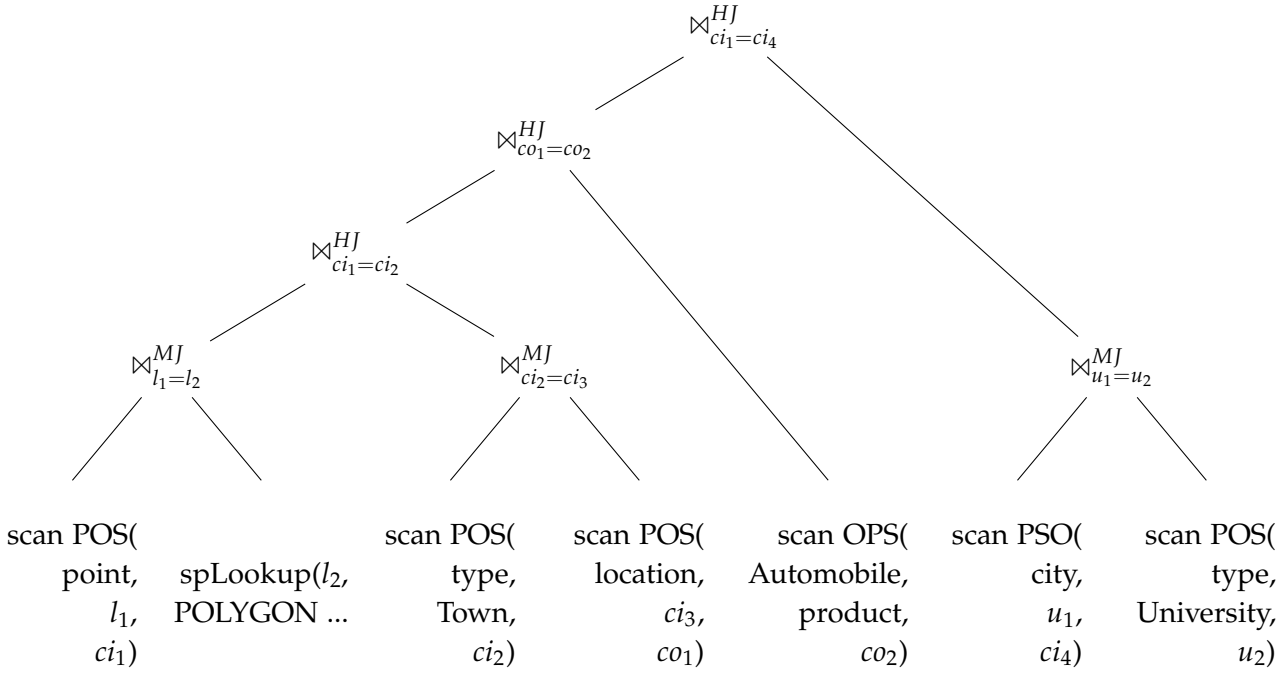


Abbildung 6.10: ursprünglicher Ausführungsplan zu Q11

Die *maßgeblichen* Verbesserungen hinsichtlich der Ausführungszeiten von Anfragen ergeben sich offenbar aus der Kombination folgender drei Mechanismen:

1. Kachelung
2. Pfadbündel-Statistik
3. Kriterium für die Entscheidung zwischen Spatial-Index-Lookup und Selektionsoperator

Die Kachelung liefert hierbei die Grundlage zur *sinnvollen*, lokal individuellen Repräsentation möglicher Korrelationen im Rahmen der Pfadbündel-Statistik. Letztere wiederum liefert in Form ihrer verbesserten Kardinalitätsabschätzungen wichtige Meta-Daten, auf deren Grundlage das Entscheidungskriterium die jeweils optimale Form der Realisierung des ortsbasierten Filters festlegen kann (gemäß Abbildung 6.8 sind die Kardinalitätsabschätzungen auf Basis der Kachelung hierfür offensichtlich noch nicht ausreichend).

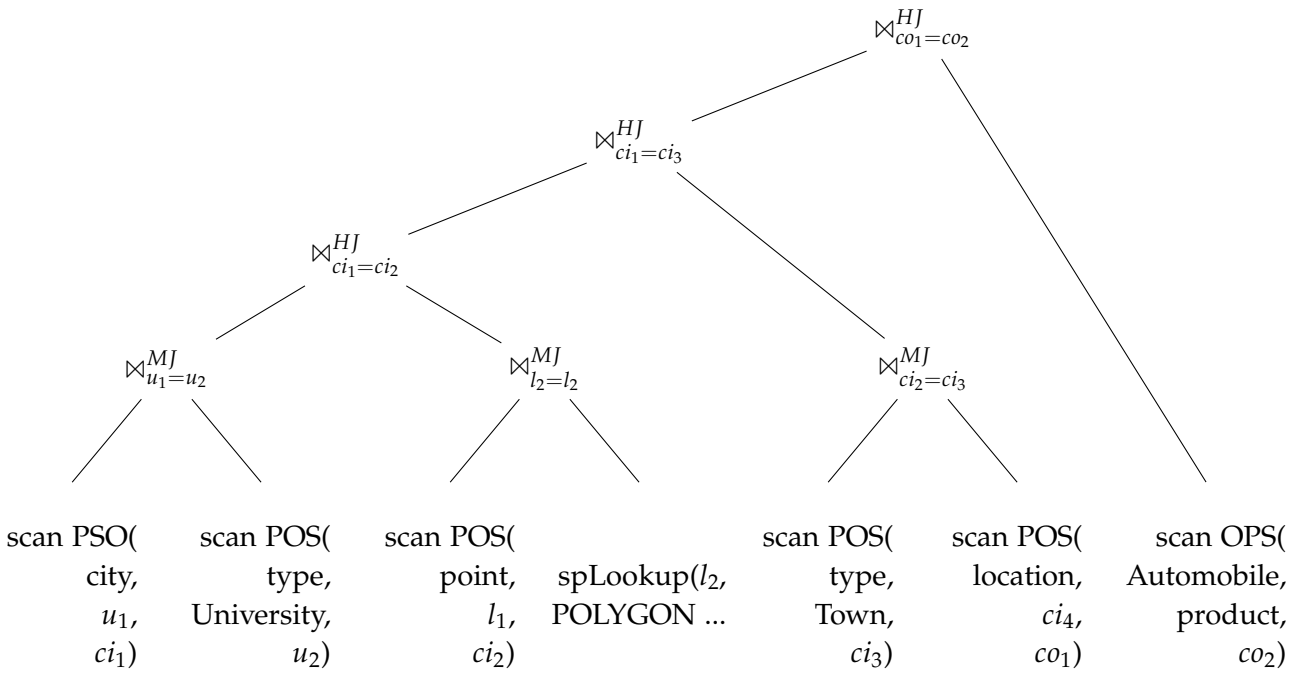


Abbildung 6.11: optimierter Ausführungsplan zu Q11

7 Zusammenfassung und Ausblick

Dieses Kapitel soll zunächst eine kurze Zusammenfassung der vorliegenden Arbeit liefern und im Anschluss einige Schlussfolgerungen ziehen. Abschließend wird ein Ausblick auf mögliche weitere Forschungsanstrengungen im Bereich der behandelten Thematik gegeben.

7.1 Zusammenfassung

Technologien des Semantic Web, wie insbesondere RDF, eignen sich hervorragend zur Darstellung und Verarbeitung semistrukturierter Daten. Dies gilt ebenso, falls die Daten Ortsinformationen beinhalten. Systeme zur Verarbeitung derartiger, ortsbasierter Daten auf Basis von RDF stellen somit erstrebenswerte Werkzeuge dar. Im Rahmen der Entwicklung eines solchen Systems, sieht man sich jedoch mit der Aufgabe konfrontiert, geeignete Ausführungspläne zu optimieren. Diese Arbeit befasste sich daher mit der Bereitstellung adäquater Entscheidungshilfen für den Optimierer eines nativen, ortsbasierten RDF-Datenbankverwaltungssystems. Aufgrund möglicher Korrelationen zwischen ortsbasierten und gewöhnlichen Attributen von Entitäten ist hierbei insbesondere ein Mechanismus zur Kardinalitätsabschätzung entsprechender Anfragen notwendig, denn die Werte einer solchen Abschätzung bestimmen maßgeblich die Anordnung der Joins in einem geeigneten Ausführungsplan, der wiederum ausschlaggebend für die Performanz der Anfragebearbeitung ist.
Da im Rahmen einer ausführlichen Literaturrecherche keine geeigneten Ansätze zur Lösung dieser Aufgabe zu finden waren, wurden hierfür eigene Konzepte entwickelt. Das erste dieser Konzepte unterteilt den betrachteten, geographischen Bereich in disjunkte Kacheln, um damit eine Kardinalitätsabschätzung für die in einem Anfragefenster enthaltenen Objekte zu ermöglichen. Auf dieser Grundlage wurde anschließend ein Ansatz entwickelt, der die Kardinalität einer Menge von Teilgraphen - und damit potentiellen Anfrageergebnissen - anhand einer Kombination ihrer enthaltenen Pfade abschätzt. Auf Basis dieser Abschätzungen war es in der Folge möglich, ein Kriterium zu konzipieren, mit dessen Hilfe geeignete Operatoren zur Realisierung eines ortsbasierten Filters ausgewählt werden können.
Die Evaluation einer Implementierung dieser Konzepte zeigte, dass anhand der Unterteilung in Kacheln lediglich moderate Verbesserungen der Abschätzungsgenauigkeit sowie der daraus resultierenden Ausführungszeiten möglich sind. Durch Hinzunahme der Kardinalitätsabschätzung mittels Pfaden konnten jedoch deutlich bessere Resultate erzielt werden. Auf deren Basis war es dem erwähnten Entscheidungskriterium außerdem möglich, in

gewinnbringender Art und Weise die jeweils optimalen Operatoren zur Realisierung eines ortsbasierten Filters auszuwählen.

7.2 Schlußfolgerungen

Die verwendete, simple Methode zur Kachelung ist gemäß Abschnitt 6.2.1 offensichtlich nicht ausreichend. Um die Leistungsfähigkeit des Systems zu verbessern, sollten daher ausgefeiltere Verfahren zum Einsatz kommen. In diesem Zusammenhang bieten sich insbesondere hierarchische, adaptive Ansätze an ([FFN+08, PHIS96, GKTD05]).

Da die betrachteten Wertebereiche der Verfahrensparameter weitgehend frei gewählt wurden und oftmals keine signifikanten Abhängigkeiten zu identifizieren waren, steht zu vermuten, dass die Parameter erst *außerhalb* des betrachteten Bereichs ihre volle Wirkung entfalten. Aus diesem Grund sollten weitere Messungen auch die bislang ignorierten Wertebereiche untersuchen.
Für den betrachteten Bereich konnte im Laufe der Evaluation eine Kombination von Verfahrensparametern gefunden werden, die einen guten Ausgangspunkt für die Nutzung des Ansatzes darstellen. Kommen hierarchische Kachelungsmethoden zum Einsatz, so wäre jedoch zu untersuchen, ob eine Unterteilung in kleinere Kacheln in einem solchen Fall Leistungssteigerungen ermöglicht.

Die Evaluation ergab moderate Verbesserungen bei Nutzung der reinen Kachelung. Deutliche Leistungssteigerungen waren erst durch geschickte Wahl des Selektionsoperators möglich. Hierfür waren jedoch die durch *zusätzliche* Nutzung der Pfadbündel-Statistik verbesserten Kardinalitätsabschätzungen notwendig. Die zu beobachtende Leistungssteigerung scheint demnach ein Resultat der geschickten Kombination dieser drei Komponenten zu sein.
Angesichts ihrer zentralen Bedeutung für die Leistungsfähigkeit des Systems, sollte der Entscheidung zur Wahl des Selektionsoperators zukünftig eine erhöhte Aufmerksamkeit geschenkt werden. Des Weiteren ist ihre Stabilität unter veränderten Rahmenbedingungen zu untersuchen und gegebenenfalls zu verbessern.

Als Quintessenz der Evaluation bleibt festzuhalten: Die Kardinalitätsabschätzungen unter Nutzung des hier entwickelten Ansatzes bieten zwar noch Potential für Verbesserungen, sind aber durchaus geeignet, um *sinnvolle* Entscheidungen des Optimierers und in der Folge effiziente Ausführungspläne zu bewirken.

7.3 Ausblick

Gemäß [NM11] nutzt die aktuelle Version von RDF-3X eine sehr viel genauere Methode zur Kardinalitätsabschätzung auf Basis der in [NM11] beschriebenen Characteristic Sets. Ein Vergleich des hier entwickelten Ansatzes mit diesem Verfahren drängt sich daher geradezu

auf. Leider stand die entsprechende Implementierung jedoch nicht zur Verfügung und deren Nachbildung hätte den Rahmen dieser Arbeit gesprengt. Ein derartiger Vergleich sollte jedoch in jedem Fall nachgeholt werden!
Darüber hinaus könnten Characteristic Sets wie folgt auch für den hier betrachteten Ansatz gewinnbringend eingesetzt werden: Die Kardinalitätsabschätzungen gemäß [NW09] stellen in aller Regel drastische Unterschätzungen dar. Ihre *relative* Ordnung spiegelt jedoch vergleichsweise gut die tatsächlichen Größenverhältnisse wider. Für sich alleine liefern sie daher mehr oder weniger brauchbare Resultate. Durch Kombination (wie in Abschnitt 4.1.2 beschrieben) mit den sehr viel präziseren, und damit größeren, Kardinalitätsabschätzungen auf Basis der Pfadbündel-Statistik entsteht jedoch eine gewisse Diskrepanz, die negative Auswirkungen auf das Gesamtergebnis haben kann. Die Nutzung einer Statistik auf Basis der Characteristic Sets könnte diese Problematik umgehen, da sich die Abschätzungen beider Verfahren in einer vergleichbaren Größenordnung bewegen.

Des Weiteren sollte zukünftig untersucht werden, inwiefern sich der hier entwickelte Ansatz zur Unterstützung von Update-Operationen modifizieren lässt.
In Ermangelung entsprechender Testdaten wurde das Verfahren bislang nur mit ortsbezogenen Punktdaten evaluiert. Für den Nachweis seiner allgemeinen Tauglichkeit steht daher noch eine zusätzliche Auswertung mit Linienzügen und Polygonen aus.
Ebenso wurde die zur Optimierung einer Anfrage notwendige Zeit bislang nur bedingt berücksichtigt. In diesem Zusammenhang sollten noch umfangreiche Verbesserungen der Implementierung möglich sein. Denkbar wären in diesem Zusammenhang sowohl eine Organisation der Pfade mittels Präfixbäumen, als auch die Anordnung der Kachel-spezifischen Statistiken auf persistentem Speicher mittels Space-Filling-Curves. Hierdurch sollte eine Reduktion der zur Optimierung notwendigen Externspeicherzugriffe möglich sein.
Da die Bitfelder zur Repräsentation zugehöriger Graphen den Speicherbedarf der Statistiken maßgeblich beeinflussen, sollte darüber hinaus auch in diesem Zusammenhang nach möglichen Optimierungen Ausschau gehalten werden. Denkbar wären hier beispielsweise eine Darstellung mittels verketteter Listen oder die Anwendung von Komprimierungsverfahren auf die generierten Bitfelder.

Im Rahmen einer allgemeineren Sichtweise wäre außerdem zu untersuchen, ob die hier entwickelten Konzepte auch in anderen Forschungsgebieten gewinnbringend eingesetzt werden können. Naheliegend wäre beispielsweise eine Nutzung zur Kardinalitätsabschätzung von Pfadausdrücken in XML-Datenbankverwaltungssystemen.

Um das Verhalten des Ansatzes im allgemeinen Fall besser einschätzen zu können, sollte dieser nicht zuletzt auch auf weiteren Datensätzen evaluiert werden. Eine Anwendung auf den Daten des Billion Triple Challenge [BTC] könnte hierbei interessante Einsichten bezüglich der Skalierbarkeit des Verfahrens ermöglichen.

A Anhang

A.1 Testanfragen

A.1.1 Testanfrage 1

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?airport WHERE {
  ?airport rdf:type     dbpediaowl:Airport .
  ?airport georss:point ?location .
  FILTER geofilter:within (?location,
    "POLYGON((11 -167, 11 -49, 74 -49, 74 -167, 11 -167))"^^geordf:geography)
    # North America
}
```

A.1.2 Testanfrage 2

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX dbpprop: <http://dbpedia.org/property/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?timezone WHERE {
  ?city rdf:type         dbpediaowl:City .
  ?city georss:point     ?location .
  ?city dbpprop:timezone ?timezone .
  FILTER geofilter:within (?location,
    "POLYGON((11 -167, 11 -49, 74 -49, 74 -167, 11 -167))"^^geordf:geography)
    # North America
}
```

A.1.3 Testanfrage 3

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?country WHERE {
  ?country rdf:type                  dbpediaowl:Country .
  ?country dbpediaowl:governmentType
           <http://dbpedia.org/resource/Federal_republic> .
  ?country georss:point              ?location .
  FILTER geofilter:within (?location,
    "POLYGON((35 -14, 35 41, 72 41, 72 -14, 35 -14))"^^geordf:geography)
    # Europe
}
```

A.1.4 Testanfrage 4

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?location WHERE {
  ?mountain rdf:type     dbpediaowl:Mountain .
  ?mountain rdfs:label   "Zugspitze" .
  ?mountain georss:point ?location .
  FILTER geofilter:within (?location,
    "POLYGON((35 -14, 35 41, 72 41, 72 -14, 35 -14))"^^geordf:geography)
    # Europe
}
```

A.1.5 Testanfrage 5

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX dbpprop: <http://dbpedia.org/property/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?city ?leader ?birthPlace WHERE {
  ?city   rdf:type              dbpediaowl:Settlement .
  ?city   georss:point          ?location .
  ?city   dbpediaowl:leaderName ?leader .
  ?leader dbpprop:birthPlace    ?birthPlace .
  FILTER geofilter:within (?location,
    "POLYGON((21 -126, 21 -58, 50 -58, 50 -126, 21 -126))"^^geordf:geography)
    # USA, except Hawaii and Alaska
}
```

A.1.6 Testanfrage 6

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX dbpprop: <http://dbpedia.org/property/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?city ?language WHERE {
  ?city    rdf:type            dbpediaowl:City .
  ?city    georss:point        ?location .
  ?country rdf:type            dbpediaowl:Country .
  ?country dbpediaowl:capital  ?city .
  ?country dbpprop:currency    "Euro" .
  ?country dbpediaowl:language ?language .
  FILTER geofilter:within (?location,
    "POLYGON((35 -14, 35 41, 72 41, 72 -14, 35 -14))"^^geordf:geography)
    # Europe
}
```

A.1.7 Testanfrage 7

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?settlement WHERE {
  ?settlement rdf:type     dbpediaowl:Settlement .
  ?settlement georss:point ?location .
  FILTER geofilter:within (?location,
    "POLYGON((48 8, 48 10, 49 10, 49 8, 48 8))"^^geordf:geography)
    # Baden-Württemberg
}
```

A.1.8 Testanfrage 8

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?object ?type WHERE {
  ?object rdf:type     ?type .
  ?object georss:point ?location .
  FILTER geofilter:within (?location,
    "POLYGON((30 129, 30 147, 46 147, 46 129, 30 129))"^^geordf:geography)
    # Japan
}
```

A.1.9 Testanfrage 9

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX yago: <http://dbpedia.org/class/yago/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?settlement WHERE {
  ?settlement rdf:type              dbpediaowl:Settlement .
  ?settlement georss:point          ?location .
  ?person     rdf:type              yago:GermanPoets .
  ?person     dbpediaowl:birthPlace ?settlement .
  FILTER geofilter:within (?location,
    "POLYGON((48 8, 48 10, 49 10, 49 8, 48 8))"^^geordf:geography)
    # Baden-Württemberg
}
```

A.1.10 Testanfrage 10

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX yago: <http://dbpedia.org/class/yago/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?city WHERE {
  ?city       rdf:type              dbpediaowl:Settlement .
  ?city       georss:point          ?location .
  ?person     rdf:type              yago:GermanPoets .
  ?person     dbpediaowl:birthPlace ?city .
  ?university rdf:type              dbpediaowl:University .
  ?university dbpediaowl:city       ?city .
  FILTER geofilter:within (?location,
    "POLYGON((48 8, 48 10, 49 10, 49 8, 48 8))"^^geordf:geography)
    # Baden-Württemberg
}
```

A.1.11 Testanfrage 11

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX dbpprop: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?city WHERE {
  ?city       rdf:type            dbpediaowl:Town .
  ?city       georss:point        ?location .
  ?university rdf:type            dbpediaowl:University .
  ?university dbpprop:city        ?city .
  ?company    dbpediaowl:product  dbpedia:Automobile .
  ?company    dbpediaowl:location ?city .
  FILTER geofilter:within (?location,
    "POLYGON((48 6, 48 16, 55 16, 55 6, 48 6))"^^geordf:geography)
    # Germany
}
```

A.1.12 Testanfrage 12

```
PREFIX dbpediaowl: <http://dbpedia.org/ontology/>
PREFIX georss: <http://www.georss.org/georss/>
PREFIX geofilter: <http://example.org/sparql/geography/filter#>
PREFIX geordf: <http://example.org/sparql/geography/>

SELECT ?country ?leader ?birthDate WHERE {
  ?country dbpediaowl:largestCity ?city .
  ?country dbpediaowl:leaderName  ?leader .
  ?leader  dbpediaowl:birthPlace  ?city .
  ?leader  dbpediaowl:birthDate   ?birthDate .
  ?city    georss:point           ?location .
  FILTER geofilter:within (?location,
    "POLYGON((-37 -21, -37 61, 38 61, 38 -21, -37 -21))"^^geordf:geography)
    # Africa + Arabian Peninsula
}
```

Abkürzungsverzeichnis

ACT	Appearance Count Threshold
DBVS	Datenbankverwaltungssystem
GeoRDF-3X	um einen Geoindex erweiterte Variante von RDF-3X, s. [BNM10]
RDF	Resource Description Framework, s. [KC04]
RDF-3X	eine native RDF-Datenbank, s. [NW10]
SIP	Sideways Information Passing
SPARQL	SPARQL Protocol And RDF Query Language
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium

Literaturverzeichnis

[ACZH10] M. Atre, V. Chaoji, M. J. Zaki, J. A. Hendler. Matrix "Bit"loaded: a scalable lightweight join query processor for RDF data. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pp. 41–50. ACM, 2010. (Zitiert auf Seite 19)

[AMFH08] J. Aguiar Moraes Filho, T. Härder. Tailor-made XML Synopses. In *DB&IS*, pp. 25–36. 2008. (Zitiert auf den Seiten 53 und 58)

[AMFHS09] J. Aguiar Moraes Filho, T. Härder, C. Sauer. Enhancing the Estimation Quality of Element-centered XML Summarization Methods. *Int. Journal of Database Theory and Application (IJDTA)*, 2(4):39–57, 2009. (Zitiert auf Seite 53)

[AMMH07] D. J. Abadi, A. Marcus, S. R. . Madden, K. Hollenbach. Scalable semantic web data management using vertical partitioning. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pp. 411–422. VLDB Endowment, 2007. (Zitiert auf den Seiten 18 und 19)

[APR99] S. Acharya, V. Poosala, S. Ramaswamy. Selectivity estimation in spatial databases. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, SIGMOD '99, pp. 13–24. ACM, New York, NY, USA, 1999. (Zitiert auf Seite 43)

[BBP+08] U. Bojārs, J. G. Breslin, . V. Peristeras, G. Tummarello, S. Decker. Interlinking the Social Web with Semantics. *IEEE Intelligent Systems*, 23:29–40, 2008. (Zitiert auf Seite 12)

[BDGL08] I. Bordino, D. Donato, A. Gionis, S. Leonardi. Mining Large Networks with Subgraph Counting. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pp. 737–742. IEEE Computer Society, Washington, DC, USA, 2008. (Zitiert auf den Seiten 46 und 58)

[BFH+75] Y. M. M. Bishop, S. E. Fienberg, P. W. Holl, R. J. Light, F. Mosteller, P. B. Imrey, Y. M. M. Bishop, S. E. Fienberg, P. W. Holl. *Discrete Multivariate Analysis*. The MIT Press, 1975. (Zitiert auf Seite 45)

[BKH01] J. Broekstra, A. Kampman, F. V. Harmelen. Sesame: An Architecture for Storing and Querying RDF Data and Schema Information. In *Semantics for the WWW*. MIT Press, 2001. (Zitiert auf Seite 18)

[BLF99] T. Berners-Lee, M. Fischetti. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. Harper San Francisco, 1st edition, 1999. (Zitiert auf Seite 9)

[BLK+09] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann. DBpedia - A crystallization point for the Web of Data. *Web Semantics*, 7:154–165, 2009. (Zitiert auf Seite 84)

[BNM10] A. Brodt, D. Nicklas, B. Mitschang. Deep integration of spatial query processing into native RDF triple stores. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pp. 33–42. ACM, New York, NY, USA, 2010. (Zitiert auf den Seiten 11, 12, 13, 15, 19, 24, 25, 27, 28, 75, 98 und 113)

[BTC] URL http://km.aifb.kit.edu/projects/btc-2010/. (Zitiert auf Seite 105)

[CDES05] E. I. Chong, S. Das, G. . Eadon, J. Srinivasan. An efficient SQL-based RDF querying scheme. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, pp. 1216–1227. VLDB Endowment, 2005. (Zitiert auf Seite 18)

[DBp] URL http://dbpedia.org/. (Zitiert auf Seite 84)

[Dem80] R. Demolombe. Estimation of the number of tuples satisfying a query expressed in predicate calculus language. In *Proceedings of the sixth international conference on Very Large Data Bases - Volume 6*, pp. 55–63. VLDB Endowment, 1980. (Zitiert auf Seite 31)

[DGR01] A. Deshpande, M. Garofalakis, R. Rastogi. Independence is good: dependency-based histogram synopses for high-dimensional data. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, SIGMOD '01, pp. 199–210. ACM, New York, NY, USA, 2001. (Zitiert auf den Seiten 44 und 45)

[DLS80] J. N. Darroch, S. L. Lauritzen, T. P. Speed. Markov Fields and Log-Linear Interaction Models for Contingency Tables. *The Annals of Statistics*, 8(3):522–539, 1980. (Zitiert auf Seite 45)

[EGK95] A. Eickler, C. A. Gerlhof, D. Kossmann. A Performance Evaluation of OID Mapping Techniques. In *Proceedings of the 21th International Conference on Very Large Data Bases*, VLDB '95, pp. 18–29. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995. (Zitiert auf den Seiten 21 und 72)

[FFN+08] Y. Fang, M. Friedman, G. Nair, M. Rys, A.-E. Schmid. Spatial Indexing in Microsoft SQL Server 2008. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pp. 1207–1216. ACM, New York, NY, USA, 2008. (Zitiert auf den Seiten 39, 40, 90 und 104)

[FSR87] C. Faloutsos, T. Sellis, N. Roussopoulos. Analysis of object oriented spatial access methods. In *Proceedings of the 1987 ACM SIGMOD international conference on Management of data*, SIGMOD '87, pp. 426–439. ACM, New York, NY, USA, 1987. (Zitiert auf Seite 40)

[GKTD05] D. Gunopulos, G. Kollios, J. Tsotras, C. Domeniconi. Selectivity estimators for multidimensional range queries over real attributes. *The VLDB Journal*, 14:137–154, 2005. (Zitiert auf den Seiten 39, 43, 44, 45, 56, 90 und 104)

[Gut84] A. Guttman. R-trees: a dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, SIGMOD '84, pp. 47–57. ACM, New York, NY, USA, 1984. (Zitiert auf Seite 27)

[Her06] J. R. Herring. OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture. Candidate, Open Geospatial Consortium, Inc., 2006. (Zitiert auf Seite 25)

[Ing] URL http://www.ingres.com/. (Zitiert auf Seite 38)

[Jag90] H. V. Jagadish. Linear clustering of objects with multiple attributes. In *Proceedings of the 1990 ACM SIGMOD international conference on Management of data*, SIGMOD '90, pp. 332–342. ACM, New York, NY, USA, 1990. (Zitiert auf Seite 42)

[KC04] G. Klyne, J. J. Carroll, editors. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. W3C Recommendation. World Wide Web Consortium, 2004. URL http://www.w3.org/TR/rdf-concepts/. (Zitiert auf den Seiten 9, 15 und 113)

[KF93] I. Kamel, C. Faloutsos. On packing R-trees. In *Proceedings of the second international conference on Information and knowledge management*, CIKM '93, pp. 490–499. ACM, New York, NY, USA, 1993. (Zitiert auf Seite 40)

[LL00] S. T. Leutenegger, M. A. López. The Effect of Buffering on the Performance of R-Trees. *IEEE Transactions on Knowledge and Data Engineering*, 12:33–44, 2000. (Zitiert auf Seite 40)

[lsi] URL http://sourceforge.net/projects/spatialindexlib. (Zitiert auf Seite 28)

[MASS08] A. Maduko, K. Anyanwu, A. Sheth, P. Schliekelman. Graph summaries for subgraph frequency estimation. In *Proceedings of the 5th European semantic web conference on The semantic web: research and applications*, ESWC'08, pp. 508–523. Springer-Verlag, Berlin, Heidelberg, 2008. (Zitiert auf den Seiten 47, 49, 50, 52 und 58)

[MFH08] J. d. A. Moraes Filho, T. Härder. Accurate histogram-based XML summarization. In *Proceedings of the 2008 ACM symposium on Applied computing*, SAC '08, pp. 998–1002. ACM, New York, NY, USA, 2008. (Zitiert auf den Seiten 53 und 58)

[NM11] T. Neumann, G. Moerkotte. Characteristic sets: Accurate cardinality estimation for RDF queries with multiple joins. In *ICDE*, pp. 984–994. 2011. (Zitiert auf den Seiten 50, 51, 52, 57, 58, 60, 77 und 104)

[NW08] T. Neumann, G. Weikum. RDF-3X: a RISC-style engine for RDF. *Proc. VLDB Endow.*, 1:647–659, 2008. (Zitiert auf den Seiten 29 und 40)

[NW09] T. Neumann, G. Weikum. Scalable join processing on very large RDF graphs. In *Proceedings of the 35th SIGMOD international conference on Management of Data*, SIGMOD '09, pp. 627–640. ACM, New York, NY, USA, 2009. (Zitiert auf den Seiten 8, 23, 32, 33, 34, 41, 77, 80 und 105)

[NW10] T. Neumann, G. Weikum. The RDF-3X engine for scalable management of RDF data. *The VLDB Journal*, 19:91–113, 2010. (Zitiert auf den Seiten 11, 19, 99 und 113)

[PG02] N. Polyzotis, M. Garofalakis. Statistical synopses for graph-structured XML databases. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, SIGMOD '02, pp. 358–369. ACM, New York, NY, USA, 2002. (Zitiert auf den Seiten 53 und 58)

[PHIS96] V. Poosala, P. J. Haas, Y. E. Ioannidis, E. J. Shekita. Improved histograms for selectivity estimation of range predicates. In *Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, SIGMOD '96, pp. 294–305. ACM, New York, NY, USA, 1996. (Zitiert auf den Seiten 42, 56, 90 und 104)

[PI97] V. Poosala, Y. E. Ioannidis. Selectivity Estimation Without the Attribute Value Independence Assumption. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, VLDB '97, pp. 486–495. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997. (Zitiert auf den Seiten 42, 43, 44 und 45)

[Posa] URL http://www.postgresql.org/. (Zitiert auf Seite 18)

[Posb] URL http://postgis.refractions.net/. (Zitiert auf den Seiten 38, 75, 78 und 79)

[Pru07] SPARQL Query Language for RDF. http://www.w3.org/TR/rdf-sparql-query/, 2007. URL http://www.w3.org/TR/rdf-sparql-query/. (Zitiert auf den Seiten 10 und 16)

[PSTW93] B.-U. Pagel, H.-W. Six, H. Toben, P. Widmayer. Towards an analysis of range query performance in spatial data structures. In *Proceedings of the 12th ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, PODS '93, pp. 214–221. ACM, New York, NY, USA, 1993. (Zitiert auf Seite 40)

[RS10] P. Ribeiro, F. Silva. g-tries: an efficient data structure for discovering network motifs. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pp. 1559–1566. ACM, New York, NY, USA, 2010. (Zitiert auf den Seiten 49, 50 und 58)

[SAB+05] M. Stonebraker, D. J. Abadi, A. . Batkin, X. Chen, M. Cherniack, . M. Ferreira, E. Lau, A. Lin, S. . Madden, E. O'Neil, P. O'Neil, A. . Rasin, N. Tran, S. Zdonik. C-store: a column-oriented DBMS. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, pp. 553–564. VLDB Endowment, 2005. (Zitiert auf Seite 18)

[SAC+79] P. G. Selinger, M. M. Astrahan, D. D. Čhamberlin, R. A. Lorie, T. G. Price. Access path selection in a relational database management system. In *Proceedings of the 1979 ACM SIGMOD international conference on Management of data*, SIGMOD '79, pp. 23–34. ACM, New York, NY, USA, 1979. (Zitiert auf den Seiten 30 und 31)

[SQL] URL http://www.sqlite.org/. (Zitiert auf Seite 76)

[SSB+08] M. Stocker, A. Seaborne, . A. Bernstein, C. Kiefer, D. Reynolds. SPARQL basic graph pattern optimization using selectivity estimation. In *Proceeding of the 17th international conference on World Wide Web*, WWW '08, pp. 595–604. ACM, New York, NY, USA, 2008. (Zitiert auf den Seiten 20, 40, 41, 42 und 52)

[TSS00] Y. Theodoridis, E. Stefanakis, T. Sellis. Efficient Cost Models for Spatial Queries Using R-Trees. *IEEE Trans. on Knowl. and Data Eng.*, 12:19–32, 2000. (Zitiert auf den Seiten 40 und 72)

[Vir] URL http://virtuoso.openlinksw.com/. (Zitiert auf Seite 19)

[VWI98] J. S. Vitter, M. Wang, B. Iyer. Data cube approximation and histograms via wavelets. In *Proceedings of the seventh international conference on Information and knowledge management*, CIKM '98, pp. 96–104. ACM, New York, NY, USA, 1998. (Zitiert auf Seite 43)

[WHFaG92] R. Want, A. Hopper, . V. Falcão, J. Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10:91–102, 1992. (Zitiert auf Seite 25)

[WKB08] C. Weiss, P. Karras, A. Bernstein. Hexastore: sextuple indexing for semantic web data management. *Proc. VLDB Endow.*, 1:1008–1019, 2008. (Zitiert auf Seite 19)

[WSKR03] K. Wilkinson, C. Sayers, H. Kuno, . D. Reynolds. Efficient RDF Storage and Retrieval in Jena2. Technical Report HPL-2003-266, Enterprise Systems and Data Management Laboratory, HP Laboratories Palo Alto, 2003. (Zitiert auf den Seiten 18 und 19)

[YCCP10] X. Yan, R. Chen, C. Cheng, X. Peng. Spatial query processing engine in spatially enabled database. In *Geoinformatics*, pp. 1–6. 2010. (Zitiert auf den Seiten 38 und 56)

[YH02] X. Yan, J. Han. gSpan: Graph-Based Substructure Pattern Mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, ICDM '02, pp. 721–. IEEE Computer Society, Washington, DC, USA, 2002. (Zitiert auf Seite 47)

Die Gültigkeit sämtlicher URLs wurde am 17. Juli 2011 verifiziert.

Erklärung

Hiermit versichere ich, diese Arbeit
selbständig verfasst und nur die angegebenen
Quellen benutzt zu haben.

(Björn Dick)