Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung – Seminar Ringvorlesung

Julissa Cusi Juarez

Skyline Query Processing in P2P Netze

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Julissa Cusi Juarez

Skyline Query Processing in P2P Netze

Ausarbeitung eingereicht im Rahmen des Seminars Ringvorlesung
im Studiengang Informatik Master of Science
am Studiendepartment Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuer: Prof. Dr. Olaf Zukunft

Abgegeben am 28.02.2011

Inhalt

1. Einleitung

Ein Typ von verteilten Systemen sind verteilte Datenbanken sowie P2P Systeme. Die Kombination von beiden führen zu einem interessanten und wenig erforschten Bereich. Ein Bestandteil dieses Bereiches ist das Query Processing oder Anfragebearbeitung. Für die Verarbeitung einer Anfrage in einer dynamischen Umgebung wie P2P Netze muss zuerst die Anfrage durch das Overlaynetz verbreitet werden oder an die geeigneten Knoten geschickt und dort bearbeitet werden.

Die Menge und die Dynamik der Daten, die in einem P2P System verfügbar sein können, bilden eine ideale Umgebung für intelligente Anfragemechanismen wie Skyline Query Processing.

Ein Skyline Query wird für multikriterielle Entscheidungsunterstützung verwendet. Zum Beispiel, die Suche von Hotels in Hamburg in der Nähe der Alster, wobei die Suchkriterien die Entfernung des Hotels von der Alster sowie der Preis sind. In diesen Fall kann ein Datenbankensystem nicht entscheiden, welches Hotel die beste für die Suche ist, aber zumindest kann es eine Menge interessanter Hotels zur Auswahl heraussuchen. Die Menge der Hotels bilden eine Skyline und die Suchkriterien sind die Anfragedimensionen.

Die Elemente einer Skyline bzw. die Skyline Punkte sind diejenigen, die nicht schlechter als andere Punkte im Bezug auf die Dimensionen sind. Eine Skyline kann in einem n-dimensionalen Raum repräsentiert werden und jede Dimension entspricht einem Query Attribut.

Dadurch fordert die Verarbeitung einer Skyline Query in P2P Netze ein optimales Verfahren, die eine gute Antwortzeit erreicht sowie niedrige Kommunikationskosten und gleiche Arbeitslast in jedem Knoten. Das Verfahren muss möglichst nur auf Knoten zugreifen, die wichtige Information für die Suche enthalten. Die Effizienz eines Verfahrens kann durch verschiedene messbare Kriterien wie Antwortzeit, Korrektheit der Ergebnisse, usw. gemessen werden.

2. Szenario

Das Szenario des Projektes besteht aus drei Bereichen. Erstens ein simulierte P2P Netzwerk, das die Anwendungsumgebung sein wird. Zweitens das Skyline Query Processing Framework, das die Anfrageverarbeitung in einem P2P System übernehmen wird. Und schließlich die Datenbanken, die auf den Peers verteilt sein werden.

Die Simulationsumgebung wurde mit dem Simulationsframework OverSim aufgebaut. Sie ermöglicht die Simulation eines Chord P2P Netzes sowie die Kommunikation zwischen der simulierten Umgebung und einer externen Applikation bzw. die Skyline Query Processing Framework.

Für das Skyline Query Processing Framework wurden Algorithmen des Frameworks Skyframe implementiert. Skyframe hat zwei wichtige Methoden, um die Netzkommunikation sowie die Antwortzeit zu optimieren.

Die Kommunikation zwischen der Simulationsumgebung und die Skyline Query Processing Framework funktioniert durch Tunneling. Die Simulationsumgebung bildet ein virtuelles Tunnel Device und kann mit dem TUN Interface von Linux Kernel kommunizieren.

Die verwendeten Daten gehören zu einer vordefinierten Datenbank. Sie stammt aus der NBA und besteht aus Saison Statistiken von 1946 bis 2009.

Das aufgebaute Szenario bildet ein experimentelles Szenario, das die Basis für die Masterarbeit sein wird.

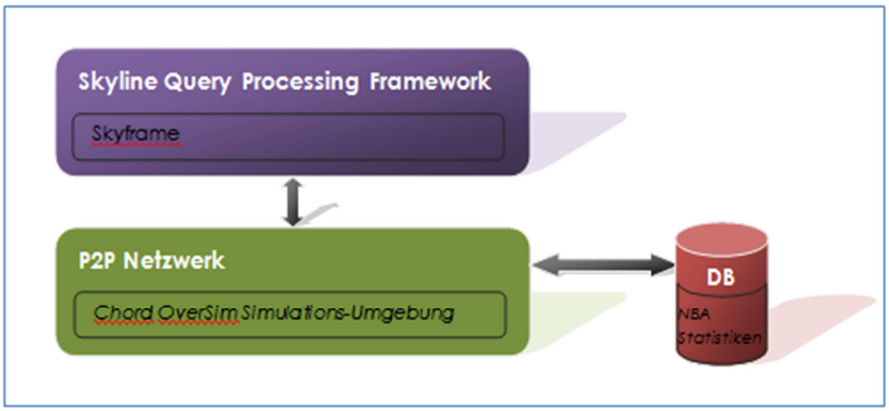

Fig. 1. Architektur des Projektes

3. Vorarbeiten

In Rahmen von Projekt 1 und Projekt 2 wurde das experimentelle Szenario aufgebaut.

3.1. Implementierung simulierter P2P Umgebung

Das Overlay Framework OverSim wurde im Jahre 2007 beim Institut für Telematik der Universität Karlsruhe (TH) im Rahmen des Projektes ScaleNet erstellt. Das ScaleNet Projekt wird vom Bundesministerium für Bildung und Forschung unterstützt. OverSim ist ein auf OMNeT++ basierendes Overlay-Framework für Linux, Windows und Mac OS X. Eine Vielzahl strukturierter und unstrukturierter Overlay-Protokolle wie Chord, Kademlia, Pastry, Bamboo, Koorde, Broose, Gia und Vast sind bereits integriert [3].

OMNeT++ ist eine erweiterungsfähige, modulare, komponentenbasierte C++ Simulationsbibliothek und ein Framework, welches hauptsächlich dazu dient, Netzwerk Simulatoren aufzubauen.

OverSim hat eine modulare Architektur, sie ermöglicht die Modellierung aller Komponenten eines P2P Netzes. Die Module können leicht ausgetauscht oder erweitert werden. Diese Eigenschaft gibt OverSim Flexibilität und ermöglicht, dass komplexe heterogene Underlay Netze simuliert werden. Eine Common API wird für die Kommunikation zwischen den verschieden Komponenten verwendet. [3]

Die Architektur von OverSim hat folgende Schichten:

- **Underlay**. Es gibt drei mögliche Modelle: **Simple Underlay**, das für große Overlay Netze bis zu 100 000 Knoten geeignet ist. **Single Host Underlay** verbindet einen einzelnen simulierten Overlay-Knoten mit einem realen Netzwerk. Schließlich **INET Underlay**, das für komplexe Simulationen heterogener Netzwerke verwendet wird. Es ist ein INET Framework basiertes Modell und ermöglicht die Simulation einer beliebigen Anzahl von Overlay Knoten. Sie können mittels Linux TUN Interface mit einem externen Netz kommunizieren [7]. TUN (TUNel) Interface ist ein virtuelles Point-To-Point Netzwerk-Device, das IP-Tunneling unterstützt. Es liefert zwei Interfaces /dev/tunX für die Benutzeranwendung (in diesem Fall OverSim) und tunX für den Linux-Kernel.

- **Overlay.** Verschiedene Overlay Modelle und Protokolle sind implementiert und gemeinsame Funktionen und Schnittstellen sind integriert.

- **Application.** Diese Schicht hat noch drei Unterteilungen Tier1, Tier2 und Tier3. Verschiedene Funktionen und Anwendungen können in jeder Unterteilung je nach Komplexität implementiert werden. OverSim hat

einige schon implementierte Komponenten. Die Anwendungen können Common API, Application Layer Multicast oder Virtual World Schnittstellen verwenden. Auch OverSim verwendet die XML-RPC Schnittstelle, um Overlay Dienste für externe Anwendungen wie verteilte Data-Speicherung zur Verfügung zu stellen.

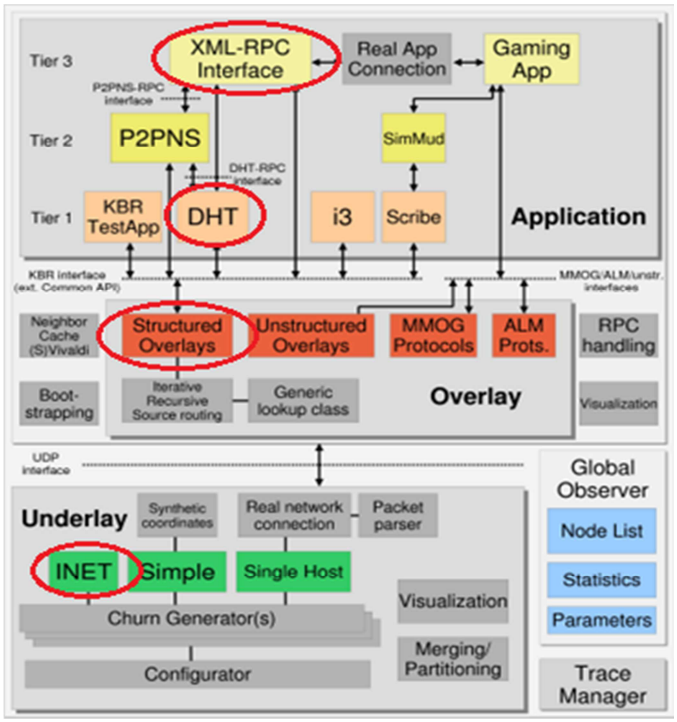

Fig. 2. Modulare Architektur OverSim und ausgewählte Komponenten (angelehnt an [5])

Um die Simulationsumgebung aufzubauen, soll eine Konfigurationsdatei erstellt werden, wo die ausgewählten Komponenten für jede Schicht definiert sind. Die implementierte Konfiguration des Projektes verwendet die INET Underlay für die Underlay-Schicht, Chord für Overlay-Schicht und DHT und RPC Funktionen für die Applikation-Schicht. Diese Konfiguration ermöglicht die Simulation mehrerer Knoten, die durch das Linux TUN Interface mit einer externen Applikation kommunizieren können. [3]

3.2. Datenbank

Die angewandten Daten stammen aus einer vordefinierten Datenbank aus der NBA. Die Datenbanken enthalten Spieler- und Spielestatistiken in Zeitraum von 1946 bis 2009. Die Daten können von der Seite www.databasebasketball.com heruntergeladen werden und sind in .csv Dateien verfügbar.

Für den Test wurde der Bericht „Blocks Leaders“ aus der NBA Seite als Referenz genommen. Die dortige Liste enthält folgende Information:

- Player Name
- Team Name
- GP: Games played
- MPG: Minutes played per game
- BLK: Blocks
- PF: Personal fouls
- BLKPG: Blocks per game
- BLKP48M: Blocks per 4 minutes
- BLK/PF: Blocks / Personal fouls

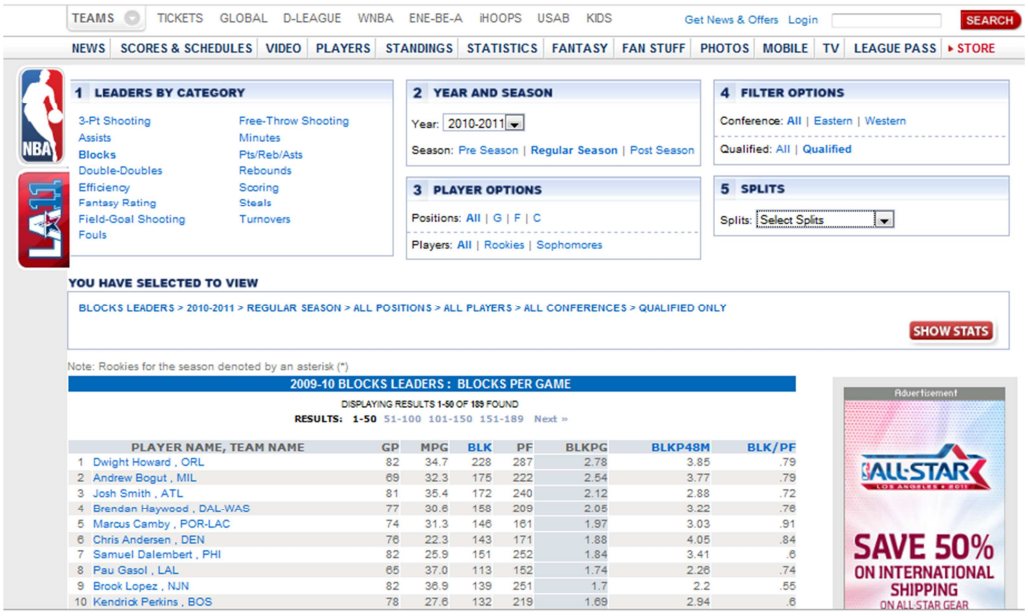

Fig. 3. „Blocks Leaders“ Bericht aus der NBA [6]

Die Daten wurden auf 10 Knoten ohne Replikation verteilt bezüglich Zeit. Zum Beispiel werden die Daten von Zeitraum von 1946 bis 1950 in dem ersten Knoten gespeichert, die Daten des Zeitraums von 1951 bis 1960 werden in dem zweiten Knoten gespeichert, usw.

3.3. Skyline Query Processing Framework

Skyline Queries werden für Multi-Kriterien Entscheidungsunterstützung benutzt. Eine Skyline besteht aus Skyline Punkten. Sie sind von anderen Datenobjekten nicht dominiert bzw. sind sie die Punkte, die am besten eine Skyline Query erfüllen. [1]

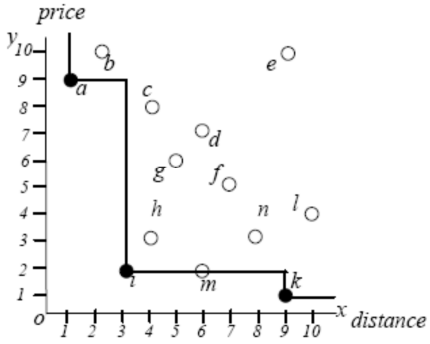

Fig. 4. Beispiel Datenmenge und Skyline [7]

Die Implementierung der Skyline Query Processing Framework basiert auf einem experimentell Framework Skyframe. Skyframe ist ein Framework für effiziente Skyline Query Processing in P2P Systemen. Sein Ziel ist es den Zeitverlauf der Anfrageverarbeitung zu optimieren, die Netzwerkkommunikationskosten zu reduzieren und die Query-load durch die Peers zu balancieren [8].

Die folgende Abbildung stellt die Skyframe Architektur dar. Die wichtigste Komponente ist der Query Processing Manager, der die Skyline Query Processing übernimmt, sowie der Load Balancing Manager, der für die Query Balancing unter den Knoten verantwortlich ist.

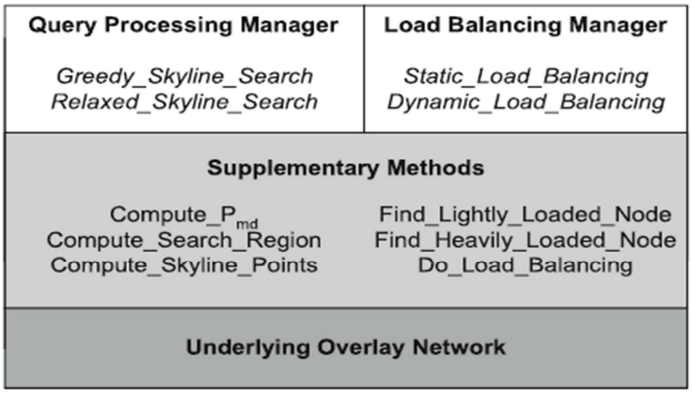

Fig. 5. Skyframe [8]

- **Query Processing Manager.** Sein Ziel ist es den Suchraum zu beschneiden, unnötigen Knotenzugriff zu vermeiden und die Skyline Punkte zu finden. Der Query Processing Manager hat zwei Suchmethoden, die sich gegenseitig ergänzen und gleichzeitig funktionieren: die Greedy und Relaxed Skyline Search. Um den Greedy Suchbereich zu bestimmen, muss man einen Anfangsknoten bzw. SQ-*Starter* bestimmen und dort einen dominantesten Punkt finden. Dieser Punkt definiert den Greedy Suchbereich. Die Knoten, die Daten in dem Greedy Suchbereich haben, sind SQ-Border Knoten und bilden den Relaxed Suchbereich.

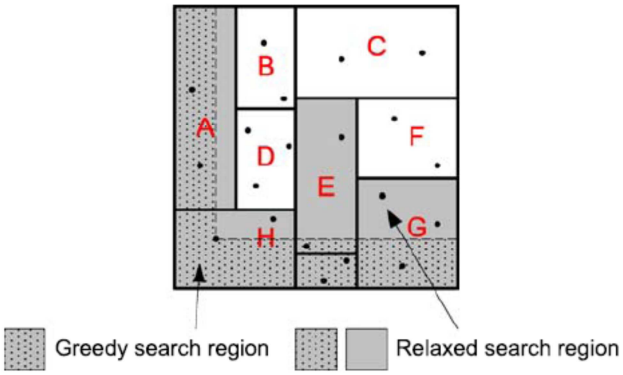

Fig. 6. Greedy und Relaxed Search Space [8]

- **Load-balancing Manager** ist verantwortlich für die Query Load Balancing zwischen den Knoten. Er partitioniert den Daten Raum und dann wird das Query-Load auf die Partitionen gleichmäßig verteilt. Außerdem wird es regelmäßig überprüft, ob es ein Gleichgewicht gibt.

  Das Query Load eines Knotens ist die Summe von (1) der Menge lokaler Datenpunkte, die das Ergebnis einer Anfrage sind, und (2) der Menge der erzeugten Nachrichten eines Knotens, die zu keiner lokalen Anfrageverarbeitung führt [8].

Für die Implementation des Algorithmus wurde ein Skyline Query bezüglich des Berichtes „Blocks Leaders“ aus der NBA ausgewählt. Die Skyline Query hat drei Dimensionen: GP (games played), BLK (blocks) und PF (personal fouls). Man kann entweder nach Minimum oder Maximum Werte in jeder Dimension suchen. Folglich gibt es acht mögliche Anfragen (min, min, min), (min, min, max), usw. Es ist angenommen, dass die Anfragedimensionen in einem dreidimensionalen Raum repräsentiert sind. Wobei jeder Anfragedimension eine

Dimension des Raumes entspricht. Die Datensätze werden als Datenpunkte in dem dreidimensionalen Raum repräsentiert. [4]

Da die Daten auf den verschiedenen Knoten verteilt wurden, ist es das Ziel des Algorithmus unnötige Knotenzugriffe zu vermeiden. Aus diese Weise werden Metadata für jeden Knoten verwendet, um zu wissen, ob der Knoten für die Suche relevant ist. Die Metadata enthält acht Grenzpunkte, d.h. der Punkt, der am besten eine Skyline Query erfüllt.

Anschließend wurde aus den Grenzpunkten aller Knoten der beste Punkt ausgewählt. Dieser Punkt wird der dominanteste Punkt sein und bestimmt den Greedy Suchbereich. Alle Knoten, die Daten im Greedy Suchbereich haben, bilden den Relaxed Suchbereich.

Die folgende Abbildung stellt beide Suchbereiche im Bezug zu der Dimension BLK dar. Der blaue Bereich repräsentiert den Greedy Suchbereich für Minimum Werte in der Dimension BLK. Der innere Würfel repräsentiert den Datenbereich eines Knotens. Da dieser Knoten Daten im Greedy Suchbereich hat, werden sie zum Relaxed Suchbereich dazugehören. Der ganze Greedy und Relaxed Suchbereich umfasst die Bereiche bezüglich aller Dimensionen. Aufgrund der einfachen grafischen Darstellung zeigt die Abbildung nur eine Dimension.

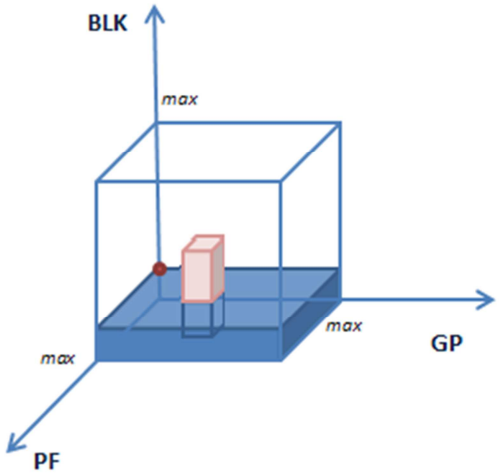

Fig. 7. Greedy und Relaxed Suchbereich bezüglich Dimension BLK

Schließlich wird auf die Knoten der Relaxed Suchbereich zugegriffen und nach Skyline Punkte im Greedy Suchbereich gesucht. Zuerst werden lokale Skyline Punkte gesucht und dann alle als globale Skyline Punkte zusammengefasst.

4. Methodischer Ansatz

Im Rahmen der Vorarbeiten wurde das Framework Skyframe als Basis für die Implementation des Skyline Query Processing Verfahrens genommen. Derzeit gibt es noch weitere Verfahren, die für die Entwicklung der Masterarbeit berücksichtigt werden. Einige dieser Ansätze sind Compressed Skycube und Skyline Operator.

4.1. Compressed Skycube

Skyline Query Processing kann zu viele Ergebnisse ergeben, insbesondere wenn das Dataset eine hohe Raumdimensionalität hat, um den Bedarf einer großen Anzahl von Benutzern zu erfüllen. Dies kann ein Overhead Processing bewirken [2].

Das Ziel des Compressed Skycube ist eine vordefinierte Cube mit Skylines von verschiedener Anfrage zu speichert. So wird die Antwortzeit und das Overhead Processing reduziert.

Das Dataset besteht aus einer beliebigen Anzahl von Attributen bzw. Dimensionen, die Skyline Queries können bezüglich aller Dimensionen oder nur einer Untermenge davon erstellt werden. Eine Untermenge der Dimensionen wird Subspace genannt, die entsprechende Query ist eine Subspace Skyline Query und die Skyline ist die Subspace Skyline.

Ein Skycube besteht aus allen möglichen Subspace Skylines. Die Punkte, die zu der Skyline gehören, können in mehrere Subspaces gespeichert sein. Folglich versucht ein Compressed Skycube die wiederholten Elemente nur einmal zu speichern. Dafür muss man die Subspaces analysieren. Wenn ein Skyline Punkt sich in zwei Subspaces befindet, und ein Subspace eine Untermenge des zweiten Subspace ist, dann wird der wiederholte Skyline Punkt nur in dem kleinsten Subspace gespeichert. Auf dieser Weise muss man für eine Subspace Skyline Query außerdem die Skyline Punkte der dazugehörenden Subspaces nehmen. Eine Subspace wird „Cuboid" genannt.

Als Beispiel hat man ein Skyline Query mit drei Dimensionen: Preis (A), Abstand zum Meer (B), Anzahl der Sterne (C). Man sucht nach Minimum Werten für „Preis", Minimum Werten für „Abstand zum Meer" und Maximum Werten für „Anzahl der Sterne" [2]. Die Tabelle 1 stellt das Dataset dar.

	A	B	C
t1	40	30	4
t2	50	10	5
t3	10	40	2
t4	30	50	1
t5	20	20	3

Tabelle 1. Beispiel Dataset [2]

Die Tabelle 2 stellt die Subspaces oder Cuboids und seine Skyline Punkte für das Beispiel dar und die Tabelle 3 stellt die Cuboids des Compressed Skycube, d.h. ohne duplizierende Skyline Punkte dar.

Cuboid	Skyline
ABC	{t1, t2, t3, t5}
AB	{t2, t3, t5}
AC	{t1, t2, t3, t5}
BC	{t2}
A	{t3}
B	{t2}
C	{t2}

Tabelle 2. Cuboids von Skycube [2]

Cuboid	Skyline
AB	{t5}
AC	{t1, t3, t5}
A	{t3}
B	{t2}
C	{t2}

Tabelle 3. Cuboids von Compressed Skycube [2]

4.2. Skyline Operator

Die Skyline Operator ist eine Erweiterung der SQL SELECT Anweisung durch die Implementierung der Klausel „SKYLINE OF“.

SELECT ... FROM ... WHERE ...
GROUP BY ... HAVING ...
SKYLINE OF [DISTINCT] d_1 [MIN | MAX | DIFF], ..., d_m [MIN | MAX | DIFF]
ORDER BY ...

Wobei d_1 bis d_m die Dimensionen der Anfrage sind. MIN, MAX und DIFF entsprechen die mögliche Werte einer Dimension Minimum, Maximum oder unterschiedlich. Die Klausel DISTINCT entfernt duplizierende Daten. Die Implementation des Skyline Operators basiert auf den „block nested loops“ und „divide and conquery“ Algorithmen. [9]

Die verteilten Daten in jedem Knoten können in mehreren Tabellen gespeichert sein. Deswegen versucht der Skyline Operator die join-Berechnungskosten, zu minimieren.

Derzeit existiert eine öffentliche Implementation des Skyline Operators: das Projekt „Random Dataset Generator for SKYLINE Operator Evaluation Project“. Dieses wurde beim Forschungsprojekt der PostgreSQL Community entwickelt [10].

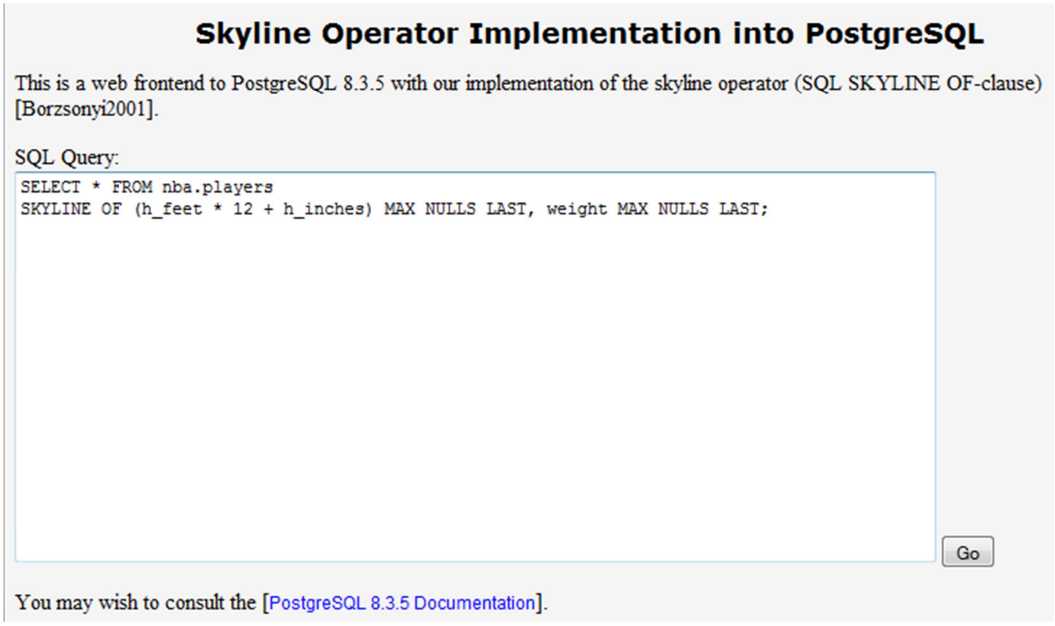

Fig. 8. Implementation der Skyline Operator[11]

5. Herausforderung

Das Ziel der Masterarbeit ist ein effizientes Verfahren zu erzeugen. Dafür hat man folgende Herausforderungen:

Das Verfahren soll nicht nur in einer simulierten Umgebung angewendet werden können, sondern auch in einer reellen Umgebung. Nur so kann das Verfahren gänzlich getestet werden. Ebenso ist der Wunsch, dass das Verfahren in mehrere Overlay-Netze funktionieren soll.

Ein wichtiger Punkt bei der Umsetzung des Verfahrens ist die Anzahl der Knoten. Für das experimentelle Szenario wurden zehn Knoten berücksichtigt. In einer reellen Umgebung muss die Anzahl der Knoten beliebig sein und das Verfahren jedoch trotzdem gut funktionieren. Bisher wurden die Daten auf den zehn Knoten ohne Redundanz verteilt. Das Verfahren muss jedoch ebenso den Redundanzfall steuern.

Andererseits ist es wichtig dass, das Verfahren für eine Skyline Query sowohl mit zwei Dimensionen als auch mit drei oder mehr effizient ist.

6. Kriterien zur Evaluierung

Um die Effizienz des Verfahrens zu bestimmen, müssen verschiedene Kriterien zur Evaluierung verwendet werden.

Ein wichtiges Kriterium ist die Korrektheit der Ergebnisse, dies kann durch die Komparation der Ergebnisse mit anderen Verfahren überprüft werden.

Weitere Kriterien sind Antwortzeit und Kommunikationskosten, es kann sowohl mit anderen Verfahren vergleichen als auch mit der simulierten Umgebung verglichen werden.

Schließlich muss es auch überprüfen, ob jeder Knoten die gleiche Last hat. Das ist ein wichtiger Punkt, um den Suchbereich zu balancieren und die Kommunikations- und Processingkosten zu reduzieren.

7. Risiken

Die Entwicklung der Masterarbeit kann einige Risiken in sich bergen.

- Das Simulations Framework könnte nicht ausreichende Bibliotheken haben, um eine Evaluierung zu realisieren. OverSim hat leider nicht viel Dokumentation, so dass die Entwicklung neue Komponenten viel Zeit in Anspruch nehmen könnte.

- Ein großes Risiko ist, dass die Simulationsergebnisse zu weit von der Realität entfernt sein könnten. In diesen Fall könnte das Verfahren falsch aufgebaut sein.

- Anderseits kann das implementierte Verfahren nicht effizient in einem anderen Overlay Netz funktionieren. So muss das Verfahren möglicherweise verändert werden.

- Ein anderes Risiko könnte in den verwendeten Daten entstehen. Die Daten könnten nicht optimal strukturiert oder verteilt sein. Diese beiden Aspekte sind fundamental für die Effizienz des Verfahrens.

8. Zusammenfassung

In dieser Arbeit wurden ein erster Plan und Erwartungen für die Masterarbeit präsentiert, sowie eine Übersicht der realisierten Arbeiten der den ersten drei Semestern des Masterstudiums.

In den Vorarbeiten wurde ein experimentelles Szenario erstellt. Das Szenario besteht aus drei Bestandteilen: P2P Netzwerk, Skyline Query Processing Framework und der Datenbank.

Für die P2P Netzwerk Komponente wurde eine simulierte Umgebung aufgebaut mit der Simulation Framework OverSim und Chord als Overlay-Protokoll. Die Skyline Query Processing wurde auf Basis der Framework Skyframe implementiert. Die Datenbank ist eine vordefinierte Datenbank, stammt aus der NBA und enthält Statistiken von 1946 bis 2009.

Anschließend wurden neue Ansätze präsentiert, die für die Entwicklung der Masterarbeit berücksichtigt werden. Die präsentierten Ansätze sind Compressed Skycube, der eine vordefinierte Skyline Punkte Struktur speichert und Skyline Operator, der die Klausel *SKYLINE OF* durch eine Erweiterung der SQL-Anweisung implementiert.

Schließlich wurden Herausforderungen für die Masterarbeit sowie möglichen Risiken, die entstehen könnten, definiert.

Schließlich wurde Skyframe, ein beispielhaftes Framework, beschrieben. Skyframe versucht den Zeitverlauf zu optimieren und niedrige Netzwerkkommunikationskosten zu erreichen.

9. Ausblick

Das Ziel der Masterarbeit ist die Weiterentwicklung eines Skyline Query Processing Verfahrens, mit Berücksichtigung neuer Ansätze, die es ermöglichen ein optimales Verfahren ausarbeiten zu können.

Das Verfahren soll die Skyline Suche optimieren angesichts der Eigenschaften einer P2P Umgebung. Ebenso soll es unnötige Knotenzugriffe vermeiden und die Last im Knoten balancieren. So können das Processing- und das Kommunikationsoverhead reduziert werden.

Die Effizienz des Verfahrens muss gemessen werden und mit weiteren Verfahren verglichen werden.

A. Literatur

[1] J. Cusi Juarez. *P2P Datenbanken.* Ausarbeitung im Rahmen der Vorlesung Anwendungen 1 im Studiengang Informatik Master of Science am Studiendepartment Informatik der Fakultät. Technik und Informatik der Hochschule für Angewandte Wissenschaften Hamburg. Februar 2010.

[2] J. Cusi Juarez. *Skyline Query Processing.* Ausarbeitung im Rahmen der Vorlesung Anwendungen 2 im Studiengang Informatik Master of Science am Studiendepartment Informatik der Fakultät. Technik und Informatik der Hochschule für Angewandte Wissenschaften Hamburg. August 2010.

[3] J. Cusi Juarez. *Skyline Query Processing für P2P Datenbanken.* Ausarbeitung im Rahmen der Vorlesung Projekt 1 im Studiengang Informatik Master of Science am Studiendepartment Informatik der Fakultät. Technik und Informatik der Hochschule für Angewandte Wissenschaften Hamburg. September 2010.

[4] J. Cusi Juarez. *Skyline Query Processing in P2P Netze.* Ausarbeitung im Rahmen der Vorlesung Projekt 2 im Studiengang Informatik Master of Science am Studiendepartment Informatik der Fakultät. Technik und Informatik der Hochschule für Angewandte Wissenschaften Hamburg. Februar 2011.

[5] I. Baumgart, B. Heep, and S. Krause. *OverSim: A Flexible Overlay Network Simulation Framework*. In Proceedings of 10th IEEE Global Internet Symposium (GI '07) in conjunction with IEEE INFOCOM 2007, Anchorage, AK, USA, May 2007.

[6] http://www.nba.com/statistics/player/Blocks.jsp

[7] D. Papadias, Y. Tao, G. Fu, B. Seeger. *An Optimal and Progressive Algorithm for Skyline Queries*. In ACM SIGMOD 2003, June 9-12.

[8] S. Wang, Q. H. Vu, B. C. Ooi, A. K. H. Tung, L. Xu. *Skyframe: a framework for skyline query processing in peer-to-peer systems*. In VLDB Journal, Vol.18, No.1, 2009.

[9] S. Börzsönyi, D. Kossmann, K. Stocker. *The Skyline Operator*. In Proceedings of the 17th International Conference on Data Engineering IEEE Computer Society Washington, DC, USA, 2001.

[10] http://randdataset.projects.postgresql.org/

[11] http://skyline.dbai.tuwien.ac.at/

B. Abbildungsverzeichnis