Der Einfluss der Datenverteilung auf die Performanz eines Data Warehouse

Thomas Legler **Wolfgang Lehner**
TU Dresden
Database Technology Group
Technische Universität Dresden
01307 Dresden, Germany
t.legler@sap.com **lehner@inf.tu-dresden.de**

Andrew Ross
PTU NetWeaver AS TREX
SAP AG
Dietmar-Hopp-Allee 16
69190 Walldorf, Germany
a.ross@sap.com

Abstract: Dieses Papier befasst sich mit einer Studie über die Optimierungsmöglichkeiten von Anfragen auf verteilten Data Warehouse Architekturen mittels verschiedenartiger Verteilungsstrategien der beteiligten Tabellen am Beispiel SAP NetWeaver BI.

1 Einleitung

Dank wachsender Vernetzung und durch automatisierende Methoden sammeln Unternehmen mehr Daten als je zuvor. Damit haben sie die Möglichkeit, interne Abläufe und Wirtschaftsdaten genau zu analysieren. Die Daten sind jedoch im Regelfall uneinheitlich und zu granular um persistiert zu werden, wodurch solche Informationen oft in vorverarbeiteter Form in einem Data Warehouse abgelegt werden. Eine weit verbreitete Darstellungsform eines Data Warehouse ist das Sternschema [CCS, KRT+98] , eine zentral angeordnete Faktentabelle mit Belegeinträgen und Verweisen auf Dimensionstabellen, welche weiterführende Informationen enthalten. Derartig komplexe Beziehungsmuster ermöglichen zahlreiche Optimierungen bei der Verarbeitung. Neben speziellen Optimierungsstrukturen und Berechnungsalgorithmen streben beispielsweise verteilte Architekturen eine gleichmässige Auslastung ihrer Einzelkomponenten an, um die entstehende Arbeit bestmöglich zu verteilen.

Dieses Papier präsentiert eine Studie über die Auswirkungen verschiedener Datenverteilungen auf die Performanz eines Data Warehouse am Beispiel der Business Intelligence Lösung der Firma SAP *SAP NetWeaver BI*. Besonders beachtet wurde dabei, wie vorhandene Metadaten neben klassischen Verteilungsstrategien für eine sinnvolle Optimierung nutzbar sind.

1.1 Sternschemas und materialisierte Sichten

Die SAP-Implementierung eines Sternschemas unterscheidet sich im Detail von der klassischen Variante. Wie in Abbildung 1 zu sehen ist, wird die zentrale Faktentabelle nicht nur von ihren Dimensionstabellen, sondern auch von X, Y und S Tabellen umgeben. Dabei speichern die X Tabellen zeitabhängige und die Y Tabellen zeitunabhängige Daten. Die Dimensionstabellen (D) speichern die Dimensionsinformationen, z.B. über Material oder Kunde, und verbinden die X/Y und S Tabellen mit der Faktentabelle. S Tabellen halten Abbildungen zwischen IDs vor.

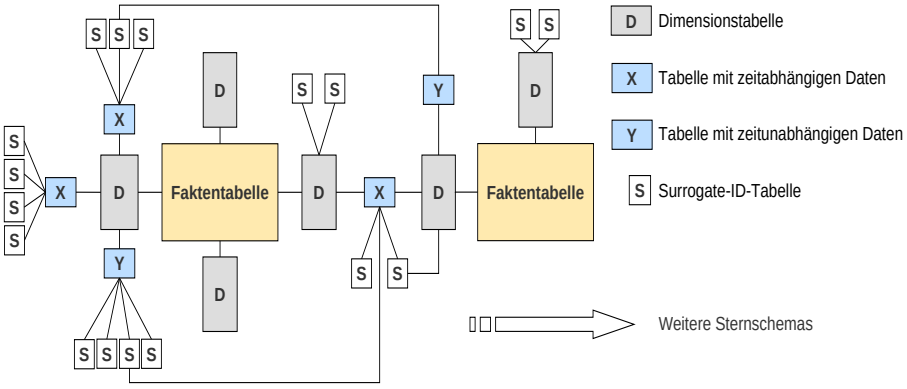

Abbildung 1: SAP Sternschemas

Diese Architektur ist flexibel und performant, benötigt aber für Anfragen auf das Schema mehrere Verbundoperationen, um Abbildungen, Hierarchien usw. aufzulösen und die Daten aus der Faktentabelle zu selektieren. Um Berechnungen bei großen Datenmengen von mehreren hundert Millionen Einträgen in der Faktentabelle schnell zu ermöglichen, werden in klassischen Implementierungen materialisierte Sichten [GHQ95, BPT97] verwendet. Diese halten Daten aus dem Sternschema in verdichteter Form redundant vor und ermöglichen einen direkteren Zugriff auf angeforderte Informationen als es über die Faktentabelle möglich ist. Diese Redundanz bereitet in vielen Firmen großen Arbeitsaufwand, da Änderungen in den Daten des Sternschemas ein Nachführen der Änderungen in die materialisierten Sichten erfordert. Das verursacht Kosten und das Fehlschlagen eines Änderungslaufes kann Inkonsistenzprobleme verursachen. Außerdem benötigen diese Sichten teilweise ein Vielfaches an Speicher im Vergleich zu den Ursprungsdaten.

1.2 SAP NetWeaver BI Accelerator

Als Lösung dieser Problematik hat SAP die Software *SAP NetWeaver BI Accelerator* entwickelt, welcher materialisierte Sichten und die sich daraus ergebenden Probleme ablöst

und die laufzeitkritischen Berechnungen ohne etwaige Vorberechnungen auf den Belegdaten ermittelt [LLR06]. Im Voraus überträgt das SAP NetWeaver BI System die zu optimierenden Sternschemas zum BI Accelerator, welcher seinerseits diese Daten in einer internen Struktur namens BIA Index ablegt.

Bei Bedarf wird dieser BIA Index vollständig in den Hauptspeicher geladen und Anfragen vom SAP BI System zur Laufzeit beantwortet. Die Hardware besteht aus einem Blade-System mit mehreren Einzelblades, welche per Gigabit-Ethernet untereinander verbunden sind und jeweils 2 CPUs und mehrere Gigabyte Hauptspeicher besitzen. Zur dauerhaften Ablage der Daten dient ein per FiberChannel angebundener Dateiserver. Die Architektur erlaubt eine parallele Verarbeitung bei komplexen Berechnungen, was für BI Anfragen ein gute Skalierung der Performanz mit der Anzahl verfügbarer Blades bewirkt. Um den Hauptspeicherverbrauch zu optimieren, werden die Daten hochgradig komprimiert vorgehalten. Beispielsweise werden sämtliche Tabellen spaltenweise abgelegt und deren Inhalt mit Hilfe von Wörterbüchern auf ganzzahlige, bitkomprimierte Werte abgebildet. Das erlaubt ein gezieltes Laden einzelner Attribute, wodurch selten benötigte Attribute auf externem Speicher verbleiben können und nicht unnötig Hauptspeicher belegen. Derartig komprimierte Daten erlauben einen sehr effizienten Zugriff und benötigen nur wenige zusätzliche Ressourcen zum Dekomprimieren von benötigten Daten. Allerdings besteht ein großen Aufwand für Schreiboperationen. Zur Effizienzsteigerung benutzt der BI Accelerator hierfür einen Delta-Mechanismus, welcher neue Informationen separat vom Hauptindex schreiboptimiert vorhält, bei Anfragen parallel zum Hauptindex anfragt und periodisch im Hintergrund mit ihm vereinigt wird. Des weiteren werden Replikate als Backup und zur Effizienz- und Stabilitätssteigerung unterstützt. Im September 2006 veröffentlichte die Winter Corporation einen Bericht über den SAP NetWeaver BI Accelerator:

> The BI Accelerator was effective in reducing response times for a broad range of queries, and is far more practical than the process of building tuned aggregates. The BI Accelerator scales as the problem size grows and as the blade chassis configuration is expanded with additional blades. Data loading and indexing onto the BI Accelerator occurs at impressive rates, outperforming the construction of optimized aggregates that would be required for comparable query response.[BD]

Da sämtliche Daten des zu optimierenden Sternschemas nur in den BI Accelerator geladen werden, entfällt damit das, im Falle der materialisierten Sichten nötige, mehrfache Einbringen von Änderungen in eine Vielzahl von Replikaten. Ein wichtiges Merkmal des BI Accelerators ist die Partitionierung großer Tabellen. Bisherige Ansätze zur verteilten Berechnung in einem Data Warehouse partitionieren die Faktentabelle auf die zu einem Rechnerverbund gehörenden Knoten und replizieren jeweils alle weiteren Tabellen, um auch verteilt Verbundoperationen effizient berechnen zu können [BM00]. Mit diesem Verfahren können globale Anfragen auf den Einzelteilen der Faktentabelle berechnet und zum Endergebnis vereinigt werden. Für die SAP-Architektur ist diese Lösung in vielen Fällen nicht tragbar, da insbesondere Dimensionstabellen von mehreren Sternschemas benutzt werden können und daher ein solches Replikat auf einer Vielzahl von Sys-

temen erstellt werden müsste. Im BI Accelerator wird Redundanz vermieden und durch geschickte Verteilung der einzelnen Tabellen und eine Nutzung verteilte Verbundoperationen optimiert. Im BI Accelerator kann eine solche Umsortierung der Einzeltabellen automatisiert in regelmässigen Abständen, bei Erreichen eines Schwellwertes für eine theoretisch zu erwartende Verbesserung oder manuell ausgelöst werden. Für den Nutzer oder das übergeordnete SAP-System erfolgt diese Umverteilung transparent.

2 Strategien

Im folgenden Abschnitt werden sechs Ansätze untersucht, nach denen eine Datenverteilung möglich ist. Unterschieden wird hierbei, ob die Distributionsstrategien Wissen über das zu verteilende Schema einbeziehen oder dies ohne Schemainformationen ermitteln.

Zufallsverteilung (ZV)
Als Ausgangspunkt dient eine zufällige Datenverteilung über alle Einzelrechner, bei der die Faktentabellen schon in Abhängigkeit ihrer Größe bis maximal der Anzahl an verfügbaren Rechnern partitioniert wurden. Die Zufallsverteilung nimmt keine Rücksicht auf Hardwareausstattung, Auslastung, Inhalt und semantische Bedeutungen der Daten und verteilt die Tabellen blind. Nach dem *Gesetz der großen Zahlen* [Dur04] bekommen die Einzelrechner mit steigender Tabellenanzahl eine annähernd gleiche Menge an Tabellen zugewiesen. Realistische Szenarien mit einigen hundert Tabellen weisen jedoch noch deutliche Ungleichverteilungen und damit unausgeglichene Belastungen der einzelnen Rechner auf.

Round-Robin (RR)
Als zweiter Algorithmus dient eine Round-Robin-Verteilung. Die Tabellenpartitionierung ist äquivalent zur Zufallsverteilung. Ein Effekt des Verfahrens ist die Gleichverteilung aller Tabellen auf die Einzelrechner (Anzahl Tabellen pro Rechner gleich ± 1). Eine solche Verteilung erfolgt ohne Wissen über das Schema, die zu erwartenden Anfragen, die Datenmenge oder die Art und Bedeutung der Daten. Vor der Zuordnung der Tabellen zu ihren Zielrechnern können alle zu verteilenden Tabellen nach einem Merkmal, z.B. Name oder Größe sortiert werden. Dadurch wird eine komplette Umplanung bei neu hinzugefügten Tabellen benötigt. Je nach Sortiermerkmal erfolgt durch die Umsortierung eine direkte oder indirekte Optimierung. Zum Beispiel erfolgt bei einer Sortierung nach Tabellengrößen eine gleichmässige Verteilung der Datenmengen. Im BI Accelerator bewirkt eine Sortierung nach Namen eine Verteilung einzelner Partitionen auf mehrere Rechner oder die Verteilung von Tabellen des gleichen Typs, da dieser im Namen codiert ist. Somit folgen nach der Sortierung alle Tabellen gleichen Typs aufeinander und werden auf verschiedene Rechner verteilt. Wenn also beispielsweise für eine Anfrage Operationen auf mehreren Dimensionstabellen parallel bearbeitet werden müssen, so ist es wahrscheinlich, dass sich diese auf verschiedenen Knoten befinden und beide Operationen unbeeinflusst voneinander berechnet werden können.

Speicherverbrauch (SV)
Unter der Annahme, dass der belegte Speicher einer Relation mit der Last korreliert, die eine Auswertung auf dieser Relation erzeugt, kann eine optimale Lastverteilung nach dem Speicherverbrauch erfolgen. Dabei werden alle Tabellen in einer Weise verteilt, dass die Summe aller Abweichung vom mittleren Speicherverbrauch aller Knoten minimal ist. Dieser Algorithmus kann im Falle des BI Accelerators vor dem Füllen mit Daten ausgeführt werden, da hier die zu erwartende Tabellengröße schon beim Definieren der Tabellen angegeben wird. Andere Systeme können eine entsprechende Umverteilung machen, wenn alle Daten geladen und die benötigten Statistiken ermittelt wurden. Da sich diese zugrunde liegenden Statistiken ändern können, muss eine regelmässige Kontrolle und eventuelle Umverteilung erfolgen. Die Annahme der Korrelation zwischen Datenmenge und Verarbeitungszeit kann in realen Szenarien verletzt werden, da nicht alle Schemas und damit auch nicht alle Daten gleichmässig angefragt und bearbeitet werden. Beispielsweise werden X und Y Tabellen wahrscheinlich sehr oft benötigt, da sie in mehreren Schemas eingebunden sind. Die Faktentabelle von selten angefragten Sternschemas belegen ihrerseits viel Speicher, werden allerdings deutlich seltener angefragt.

CPU-Auslastung (CA)
Um eine genauere Aussage über die Auslastung der einzelnen Knoten zu bekommen, kann die Zeit gemessen werden, die eine CPU in einem bestimmten Zeitraum mit Berechnungen auf einer Tabelle verbracht hat. Die Zuweisung der Tabellen zu ihren Rechnern erfolgt durch das Minimieren der Abweichung vom Mittelwert der Gesamtarbeitszeiten aller Einzelknoten. Diese Berechnungen sind nur durch Statistikdaten möglich, die in einem definierten Zeitraum durch eine Analyse von Nutzerverhalten gesammelt werden. Diese Statistiken sind teilweise abhängig von periodischen Schwankungen, verschiedene Anfrageprioritäten, außergewöhnlich häufig benötigte Tabellen usw., wodurch nicht optimale Verteilungsmuster entstehen können. Um diese Effekte zu minimieren, muss eine Statistik über einen längeren Zeitraum ermittelt werden. Außerdem sind regelmässige Umplanungen sinnvoll. Das Verfahren ist sehr adaptiv, kann aber nicht ohne vorhandenen Statistiken benutzt werden. Eine gute Verteilung wird oft erst nach mehreren Umplanungsläufen erreicht.

Verbundpfadoptimiert (VO)
In manchen Anwendungsszenarien ist neben den rein technischen Informationen wie Speichergröße und CPU-Last auch Wissen über die Struktur und die Bedeutung der zu verteilenden Daten vorhanden. Eine klassische OLAP-Anfrage auf einem oder mehreren Sternschemas lässt sich laut [CM04] auf folgende Arbeitsschritte reduzieren:

1. Selektieren von Datensätzen auf der Faktentabelle
2. Gruppieren nach mindestens einer Dimension bzw. deren Attributen
3. Aggregieren der Kennzahlen innerhalb der Einzelgruppen

Selektiert werden dabei im Regelfall nur Ausschnitte der Faktentabelle, meist nur wenige Prozente der Gesamttabelle. Um diese Selektion zu erreichen, wird im allgemeinen Sternschema zuerst eine Selektion auf den gegenüber der Faktentabelle deutlich kleineren Dimensionstabellen durchgeführt und deren Ergebnis zur meist drastischen Einschränkung der betrachteten Einträge der Faktentabelle mittels Verbundoperationen genutzt. Das SAP-Sternschema hat die Dimensionstabellen noch weiter normalisiert und semantisch partitio-

niert und damit um die Dimensionstabellen noch weitere Satellitentabellen platziert. Für eine OLAP-Anfrage muss dadurch erst eine Reihe von Verbundoperationen und Selektionen ausgeführt werden. Diese betreffen jeweils nur kleine Datenmengen und können effizient bearbeitet werden.

Der BI Accelerator speichert beim Anlegen eines Sternschemas zusätzliche zu den Tabellennamen und Daten noch weitere Metadaten. So genannte Verbundpfade repräsentieren alle möglichen zu erwartenden Verbundoperationen innerhalb des Sternschemas. Diese Einschränkung auf die Menge aller möglichen Operationen ist für dieses Szenario möglich, da grundsätzlich nur das SAP-Sternschema benutzt wird und innerhalb dieses Schemas nur entlang fester Kanten und immer von außen nach innen zur Faktentabelle verbunden wird. Dies ermöglicht eine Optimierung auf die fest definierten Operationen und vermeidet die Nachteile allgemein gültigerer Implementierungen klassischer Datenbanken. Abbildung 1 zeigt ein solches Beispiel. Gezeigt wird eine Faktentabelle umgeben von vier Dimensionstabellen, die ihrerseits wieder X, Y und S Tabellen referenzieren. Ein wichtiges Detail stellen die Mehrfachnutzungen der X, Y und S Tabellen dar, welche zur Redundanzvermeidung sowohl innerhalb eines Sternschemas als auch schemaübergreifend mehrfach benutzt werden.

Mit diesen Metadaten wird dem BI Accelerator ein Möglichkeit zur Verfügung gestellt, eine Gruppierung der zu verteilenden Tabellen nach den Verbundpfaden, also von der Faktentabelle über die Dimensionstabellen zu den X, Y und S Tabellen, durchzuführen. Eine Optimierung erfolgt auf eine möglichst große Lokalität für Verbundoperationen. Im Regelfall ist die Laufzeit für diese Operationen ohne Verwendung der Faktentabelle klein im Vergleich zur Gesamtlaufzeit einer Anfrage. Eine verteilte Berechnung einer solchen Operation ist meist teurer als eine lokale unverteilte Verarbeitung und sollte vermieden werden. Deutlich zeitaufwändiger ist die Selektion, Gruppierung und die danach folgende Aggregation auf den Daten der Faktentabelle. Der damit verbundene Rechenaufwand sollte auf mehrere Rechner verteilt werden, damit eine Parallelverarbeitung der Daten stattfinden kann. Die Vereinigung der Teilergebnisse im Falle einer verteilten, parallelen Aggregation ist wegen den relativ kleinen zu erwartenden Ergebnismengen meist sehr schnell und effizient, wodurch eine deutliche Steigerung gegenüber einer unpartitionierten Berechnung zu erwarten ist.

Eine weitere Optimierung ist möglich, wenn mehrere Verbundpfaden ein und dieselbe Tabelle benötigen. In diesem Fall werden möglichst alle beteiligten Gruppen auf dem selben Rechner platziert. Ist dies nicht möglich, weil beispielsweise mehrere Tabellen eines Verbundpfades mehrfach referenziert werden, so muss eine Gruppe aufgeteilt werden und die Verbundoperationen innerhalb der geteilten Gruppe über Rechnergrenzen hinweg erfolgen. Der theoretisch zu erwartende Speicher- und CPU-Verbrauch weicht bei diesem Verfahren in der Regel deutlich von einer Gleichverteilung ab. Reale Szenarien weisen hier meist mehrere unabhängige Gruppen auf, um den kompletten Rechnerverbund gleichmässig nutzen zu können. Eine nachträgliche Optimierung und Angleichung der einzelnen Speicherverbräuche kann die Performanz aber weiter verbessern.

Für diese Szenarien schlagen wir folgenden Algorithmus vor:

1. Faktentabelle partitionieren und auf mehrere Rechner verteilen
2. Verbundpfade (mit Ausnahme der Faktentabelle) extrahieren
3. Mehrfach benutzte Tabellen zwischen den Verbundpfaden ermitteln
4. Alle so verbundenen Tabellen auf einem Rechner platzieren
5. Ungleiche Verteilungen der Datenmengen ausgleichen

Dezentrale Ablage (DA)
Um eine möglichst hohe parallele Verarbeitung zu erreichen, ist auch eine maximale Verteilung aller wahrscheinlich gleichzeitig an einer Anfrage beteiligter Tabellen denkbar. Unter den gleichen Gegebenheiten wie bei einer verbundpfadoptimierten Verteilung können die an einem Pfad beteiligten Tabellen auf möglichst viele Rechner verteilt werden, um die Selektionen auf den Einzeltabellen möglichst parallel zu verarbeiten. Nachteilig entwickelt sich dann hierbei der Aufwand zur Berechnung einer Verbundoperation, da dieser dann grundsätzlich über Rechnergrenzen hinweg erfolgen muss. Im Szenario eines SAP-Sternschemas ist das jedoch tragbar, da die an dem Verbundpfad beteiligten Tabellen oft sehr klein sind und damit wenig Daten einer Tabelle über ein Netzwerk an den jeweiligen Verbundpartner geschickt werden.

Verfahren	Schemawissen	Umplanungen	Aufwand
ZV	nein	keine	minimal
RR	nein	bei neuen Tabellen	gering
SV	nein	bei neuen Daten	mittel
CA	nein	regelmässig	hoch
VO	ja	bei Schemaänderung	sehr hoch
DA	ja	bei Schemaänderung	hoch

Tabelle 1: Übersicht über Verteilungsstrategien

3 Evaluation

Im folgenden Abschnitt werden die vorgestellten Verteilungsverfahren ausgemessen und ihr Einfluss auf die Ausführungszeiten sowohl für sequentielle als auch für Massenanfragen dargestellt.

3.1 Daten und Anfragen

Da die zugrunde liegende Architektur, der BI Accelerator, derzeit nur in der Lage ist, mit SAP-Szenarios umzugehen, ist noch keine Messung an standardisierten Benchmarks wie

z.B. TPC-H [TPC] möglich. Daher wurden die folgenden Ergebnisse auf realen Kundendaten ermittelt. Verwendet wurden acht Einzelrechner (Blades). Jedem Knoten standen hierbei ein Intel Dual Xeon 3.2 GHz, sowie 2 GB Hauptspeicher zur Verfügung.

Das Szenario umfasst insgesamt neun Sternschemas und entsprechend neun Faktentabellen. Die Sternschemas besitzen jeweils 14-16 Dimensionstabellen unterschiedlicher Größen. Die X, Y und S Tabellen sind jeweils von mehreren Schemas und mehrfach innerhalb eines Schemas benutzt und umfassen je zwischen 500.000 und 5.000.000 Einträge für die X/Y Tabellen. Die S Tabellen sind zu ca. 85% mit unter 10.000 Einträgen belegt, 10% erreichen bis zu 1.000.000 Einträge und 5% bis zu 6.000.000 Einträge. Tabelle 2 zeigt eine Übersicht über die Größe der Faktentabellen der verwendeten Sternschemas. Der Aufbau

Sternschema	Einträge in Mill.	Partitionen	Dimensionen	X/Y	S
1	463	26	16	5	84
2	183	17	16	5	89
3	110	6	14	5	75
4	37	7	16	5	84
5	36	4	16	5	89
6-9	$<$20	1	14-16	5	75-89

Tabelle 2: Einträge pro Faktentabelle

aller Schemas ist identisch. Jede Faktentabelle besteht aus 14-16 Fremdschlüsseln auf die jeweiligen Dimensionstabellen, sowie 22 weiteren Attributen. Insgesamt sind in dem kompletten Szenario 327 verschiedene Tabellen involviert.

Die 20 Testanfragen wurden aus dem Anfragebestand nach dem Nutzungsverhalten des Kunden ausgewählt. Sie benötigen in den meisten Fällen Informationen aus mehreren Sternschemas. Das System muss daher mehrerere Anfragen vom ähnlichem Typ parallel bearbeiten. Die Anfragen analysieren Lagerbestände, Umsätze einzelner Kunden, Provisionen, Reklamationen, Steuerausgaben und das Kundenmanagement. Berechnet werden dabei jeweils mehrere Aggregationen pro Anfrage und Sternschema.

3.2 Messungen

Die Messung zur Bestimmung der Auswirkungen unterschiedlicher Verteilungsstrategien erfolgt jeweils für sequentiell sowie parallel ausgeführte Anfragen. Die Laufzeiten stellen die Summe der einzelnen durchschnittlichen Anfragelaufzeiten von 20 Messungen dar. Betrachtet wurden in dem Versuch zwei Versionen einer zufälligen Verteilung (ZV1, ZV2), eine Optimierung nach Speicherverbrauch (SV), eine Round-Robin Implementierung mit nach Namen sortierten Tabellen (RR), eine maximal dezentralisierte Verteilung von Verbundpartnern (DA), eine Gruppierung nach dem Verbundpfad mit und ohne Optimierung bei der Nutzung einer mehrfach benutzten Tabelle (VO1, VO2), sowie sechs Durchläufe mit wachsenden Zeitaufwand zur Bestimmung von Laufzeitstatistiken einer Tabelle (CA1-CA6). Sämtliche Daten befinden sich zur Zeit der Messung bereits im Speicher, was den

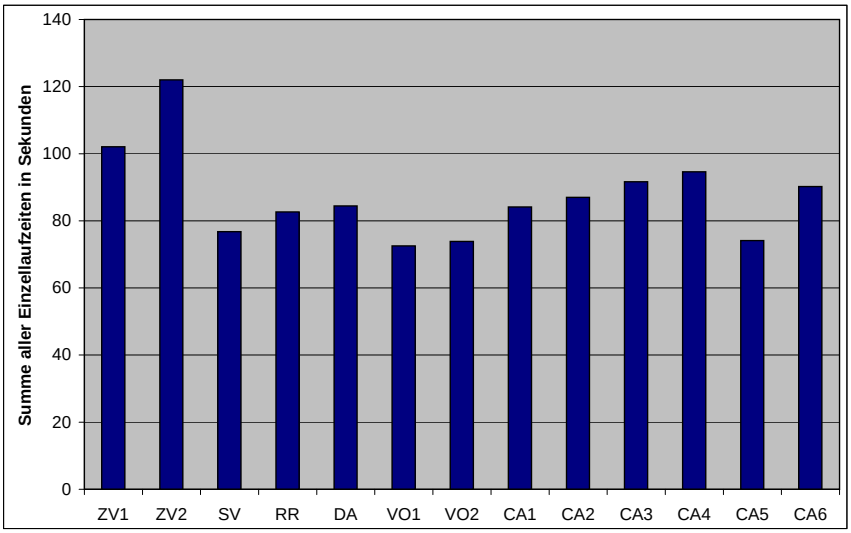

Abbildung 2: Summe aller Laufzeiten, sequentiell

regulären Funktionsprinzipien des BI Accelerators entspricht.

Abbildung 2 zeigt die ermittelten Laufzeiten aufsummiert über allen Anfragen bei sequentieller Ausführung. Man kann hierbei erkennen, dass die Ausführungsgeschwindigkeit der Anfragen deutlich von der Verteilung der benötigten Daten abhängt. Das gilt sowohl für eine gleichmässige Belastung aller beteiligten Rechner nach ihrem Speicherverbrauch als auch bei den schemaorientierten Ansätzen. Der Round Robin- und der dezentralisierte Ansatz (RR, DA) profitieren von einer Verteilung der zeitaufwändigen Operationen auf den Faktentabellen und zeigen eine deutliche Optimierung gegenüber einer zufälligen Verteilung. Auffällig ist die Entwicklung der Laufzeiten bei einer Optimierung nach gemessenen CPU-Auslastungen (CA). Hierbei ist gut zu erkennen, dass eine genauere Messung oder Statistik nicht zwangsläufig zu einer guten Optimierung führen muss. Eine deutliche Verschlechterung ist durchaus möglich, da entstehende Lasten nach jeder Umplanung anders gelagert sein können. Außerdem sind je nach Verteilung der einzelnen Tabellen negative Konstellationen möglich, wenn z.B. zufällig mehrere Teile einer Anfrage, welche voneinander unabhängig und parallel berechenbar sind, durch eine gemeinsame Zuordnung zu einem Rechner gleichzeitig berechnet werden und sie sich somit gegenseitig behindern. Beispielsweise weist die Verteilung mit der CA5-Optimierung sehr gute Laufzeiten auf, welche sich allerdings im sechsten Schritt wieder deutlich verschlechtern. Abbildung 3 wiederholt den Versuch, wobei aber alle Einzelanfragen gleichzeitig mehrfach nacheinander abgesetzt werden. Ein solches Szenario simuliert eine praktische Nutzung des Systems, da in realen Szenarien meist mehrere verschiedenartige Anfragen parallel beantwortet werden müssen. Die Laufzeiten sind hierbei weniger von Optimierungen im Einzelfall als von einer globalen Optimierung abhängig. Die Unterschiede zwischen den einzelnen Verteilungsverfahren werden bei paralleler Ausführung deutlicher, entsprechen aber im wesentlichen den gleichen Mustern wie bei sequentieller Ausführung. Zu beachten ist hier, dass

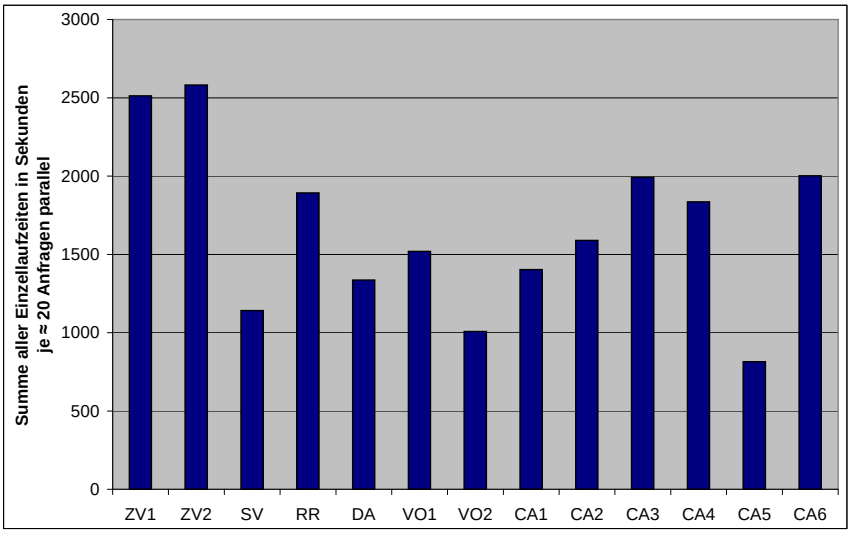

Abbildung 3: Summe aller Laufzeiten, parallel

eine einfache Optimierung nach Verbundpfaden (VO1) deutlich an Effizienz verliert, da Pfade mit mehrfach genutzten Tabellen geteilt wurden. Wenn diese Gruppenlokalität beachtet wird (VO2), so ergibt sich ein günstigeres Laufzeitverhalten. Die CA5-Optimierung zeigt aber, dass deutlich bessere Laufzeiten möglich sind.

Abbildung 4 zeigt die einzelnen Anfragelaufzeiten für die Verteilungsstrategien im sequentiellem Betrieb. Hierbei ist gut ersichtlich, dass die für eine Anfrage benötigte Zeit je nach Verfahren sehr starken Schwankungen unterliegt.

3.3 Auswertung und Ausblick

Die Evaluation zeigt deutliche Unterschiede für alle getesteten Verteilungsverfahren. Am effizientesten war hierbei eine Optimierung nach CPU-Verbrauch (CA), welche aber nicht deterministisch bestimmt werden konnte und somit nur bedingt zu empfehlen ist. Alle weiteren Messungen zur Verteilung nach CPU-Verbrauch liegen sogar nur im mittlerem bzw. hinterem Feld aller Verfahren. Außerdem hat dieser Algorithmus den Nachteil, dass eine solche Zuordnung nur mit vorhandenen Statistikdaten erfolgen kann. Diese sind aber erst nach einigen Durchläufen von Anfragen verfügbar und ändern sich im zeitlichen Verlauf. Somit sind regelmässig Umplanungen nötig. Eine Round-Robin-Verteilung (RR) ist denkbar, wenn Anfragen sequentiell ausgeführt werden. Bei paralleler Ausführung mehrerer Anfragen zeigt sich jedoch eine schlechte Skalierung. Wenn alle gespeicherten Daten gleichmässig angefragt werden, ist eine Gleichverteilung der Daten auf alle beteiligten Rechner eine sinnvolle Optimierung, sowohl bei sequentieller als auch paralleler Ausführung von Anfragen.

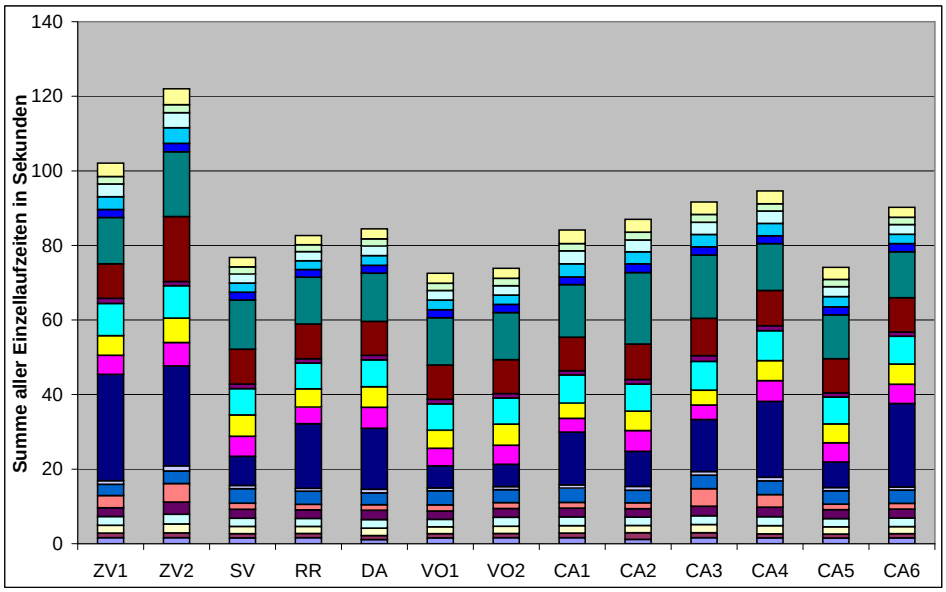

Abbildung 4: Laufzeiten Einzelanfragen, sequentiell

Eine Optimierung unter Zuhilfenahme von Schemainformationen kann ohne vorhandene Statistikdaten erfolgen, ist unabhängig von ungleichmässiger Nutzung der Daten und benötigt nur eine Umplanung, wenn ein Schema geändert wird. Mit diesen Informationen sind vergleichbare oder bessere Ergebnisse als bei einer Optimierung nach Statistikdaten zu erwarten, wie in mehreren weiteren Szenarien bestätigt werden konnte. Da diese Informationen schon bei der Definition eines Szenarios verfügbar ist und nur selten geändert wird, lässt sich mit einer Optimierung nach Schemainformationen schon vor dem Füllen der Tabellen eine performante Verteilung ermitteln. Für parallele Anfragen sollte zusätzlich auf eine möglichst große Lokalität von Verbundpfaden geachtet werden.

Die Experiment haben gezeigt, dass eine schemabasierte Optimierung noch nicht die optimale Verteilung findet. Eine Verteilung nach CPU-Last kann, wenn auch nicht deterministisch, eine bessere Lösung bieten. Aus diesem Grund ist es sinnvoll, beide Verfahren zu kombinieren. Ausgehend von einer initialen schemabasierten Verteilung ist ein Greedy-Strategie unter Nutzung von gemessenen CPU-Laufzeiten denkbar, um eine weitere Verbesserung zu erreichen.

Literatur

[BD] R. Burns und R. Dorin. The SAP NetWeaver BI Accelerator - Transforming Business Intelligence, White Paper, Winter Corporation, 2006, http://www.wintercorp.com/whitepapers/whitepapers.asp.

[BM00] J. Bernardino und H. Madeira. A new technique to speedup queries in data warehousing. In *Symposium on Advances in Databases and Information Systems - 4th East-European Conf. on Advances in Databases and Information Systems*, 2000.

[BPT97] E. Baralis, S. Paraboschi und E. Teniente. Materialized view selection in a multidimensional database. In *Proc. 23rd Int. Conf. on Very Large Data Bases*, 1997.

[CCS] E.F. Codd, S.B. Codd und C.T. Salley. Providing OLAP (On-line Analytical Processing) to User Analysts: An IT Mandate, White Paper, Arbor Software Corporation, 1993.

[CM04] M. Costa und H. Madeira. Handling big dimensions in distributed data warehouses using the DWS technique. In *DOLAP '04: Proc. of 7th ACM int. workshop on Data warehousing and OLAP*, Seiten 31–37, New York, 2004. ACM Press.

[Dur04] R. Durrett. *Probability: Theory and Examples.* 3rd edition. Duxbury Press, 2004.

[GHQ95] A. Gupta, V. Harinarayan und D. Quass. Aggregate-query processing in data warehousing environments. In *Proc. 21th Int. Conf. on Very Large Data Bases*, 1995.

[KRT+98] R. Kimball, L. Reeves, W. Thornthwaite, M. Ross und W. Thornwaite. *The Data Warehouse Lifecycle Toolkit: Expert Methods for Designing, Developing and Deploying Data Warehouses*. John Wiley & Sons, Inc., New York, NY, 1998.

[LLR06] T. Legler, W. Lehner und A. Ross. Data mining with the SAP NetWeaver BI Accelerator. In *Proc. 32nd Int. Conf. on Very Large Data Bases*, Seiten 1059–1068, 2006.

[TPC] The TPC-H Benchmark, http://www.tpc.org/tpch/.