Ökonomische Modelle für VDBMS

Was haben wir bisher über Allokation und Anfrageoptimierung gelernt? (Planwirtschaft!)

- Man nimmt ein Riesenmodell und entscheidet, welche Partitionen repliziert werden und wo die einzelnen Kopien der Partitionen gespeichert werden.
- Man nimmt ein Riesenkostenmodell und entscheidet, wie eine Anfrage bei gegebener Allokation ausgeführt wird.

Motivation für etwas Besseres:

- Die Systeme sind so groß, daß man sie nicht mehr durch Modelle beherrschen kann. (Gigamodelle sind nicht praktikabel.)
- Dynamische Anpassung an die Gegebenheiten.
  (Selbst wenn die Modelle richtig sind, kriegt man nicht ihre Eingabeparameter; und selbst wenn man die Eingabeparameter bekäme, ändern sie sich zu schnell und man steht vor der Entscheidung, wie häufig man die Modelle anwerfen muß.)

Der Weg: (Magie der Marktwirtschaft)

Die Welt ist ein Marktplatz. Jede Site ist ein Marktstand, der versucht durch Ausführung von Anfragen (Umsatz), seinen Gewinn zu maximieren.

Literatur:

- Nikolau et al.
- Stonebraker et al., The VLDB Journal, Januar 1996 (Mariposa)

Grundprinzipien von Mariposa

1. Es gibt eine Währung, z.B. $.
2. Jede Anfrage kriegt ein Budget in dieser Währung.
3. Jede Anfrage wird von einem Broker gemanaged. Der Broker verhandelt mit den Sites, damit die Anfrage innerhalb des Budgets ausgeführt werden kann.
4. Die Sites bieten wie auf einer Auktion um die Ausführung von Teilen der Anfrage. Wie im richtigen Leben stehen die Sites vor dem folgenden Dilemma:
   - Wenn ihr Preis zu hoch ist, dann kriegen sie keine Aufträge und verhungern.
   - Wenn ihr Preis zu niedrig ist, machen sie Verlust oder nur sehr wenig Gewinn.
5. Wichtig: der Broker versucht seinen Gewinn ebenfalls zu maximieren, indem er das billigste Angebot findet und den Rest des Budgets als Gewinn einstreicht.

Genauer Ablauf einer Anfrage

1. Budget einer Anfrage: Preis-Antwortzeit Kurve.
   (Macht der Benutzer.)

2. Finde sehr guten zentralen Plan für Join Ordering.
   (Vorsicht: Das führt zu 2-Phasen Optimierung und liefert nicht immer insgesamt optimale Pläne.)

3. Query Fragmenter: hier wird Partitionierungsschema berücksichtigt und die Operatoren werden umgeschrieben.
   (Vorsicht: Wir haben so etwas vor der eigentlichen Optimierung im Query Rewrite auf logischer Ebene gemacht, damit der kostenbasierte Optimierer das in der Bestimmung des besten Planes berücksichtigen kann.)

4. Der Broker gibt die einzelnen Operatoren zur Versteigerung frei. D.h. jede Site kann sich um jeden der einzelnen Operatoren bewerben, indem sie Gebot der Form $\langle$ Preis, Antwortzeit, Expiration Date $\rangle$ abgibt.

5. Broker sammelt die einzelnen Gebote und versucht daraus einen Plan zu basteln, der seinen Gewinn maximiert.

6. Falls Anfrage im Rahmen des Budgets ausgeführt werden kann, schließt der Broker die entsprechenden Verträge mit den (Winner-) Sites der Auktion ab und initiert die Ausführung der Anfrage und streicht seinen Gewinn ein. Ansonsten scheitert die Anfrage und niemand macht Umsätze. Der Benutzer muß in diesem Fall sein Budget erhöhen oder einfach auf die Anfrage verzichten.

Hinweise:

1. Netzwerk ist Ressource wie jede andere: sie kostet $. Broker muß $ für Netzwerk in seinen Kalkulationen berücksichtigen. (Falls Netzwerk kein Monopolist kann er sogar Kommunikation zur Auktion freigeben.)
2. Granularität (Operatoren oder einzelne Teilbäume) in der Broker Sachen zur Auktion gibt ist ein Freiheitsgrad dieses Ansatzes. Es ist nicht klar, wie das die Ergebnisse beeinträchtigt.

Details der Auktion

Vier Themen:

1. Der Broker: Wie batzt er die Gebote zusammen?
2. Werbung: Wie findet man passende Sites?
3. Die Sites: Wie bestimmen sie ihre Gebote?
4. Preiswerte Protokolle: Wie kauft man Schnürrsenkel ohne Makler?

Der Broker

Problem:

Gegeben die Budgetkurve der Anfrage und ein Riesenberg Gebote. Wie maximiert der Broker seinen Gewinn?

Antwort: Mariposa verwendet folgende Heuristik:

1. Baue Lösung mit minimaler Antwortzeit.
   (Diese Lösung wird sehr hohe Kosten haben, da sie nur die teuersten Gebote mit der geringsten Antwortzeit berücksichtigt.)
2. Ersetze teure Gebote, die nichts bringen, da sie nicht auf dem *kritischen Pfad* liegen.
   $\rightarrow$ Lösung wird billiger bei gleicher Antwortzeit.
3. Sortiere restlichen (bisher noch nicht berücksichtigten Gebote) nach absteigendem Kostengradient:

$$\text{Kostengradient}(G, L) = \frac{\text{Kostenersparnis durch } G}{\text{Verzögerung in } L \text{ durch Verwendung von } G}$$

4. Probiere die restlichen Gebote, Gebot für Gebot nach absteigendem Kostengradienten, ob sie den Gewinn des Brokers erhöhen.

Frage: Das ist doch im Grunde Site Selection. Kann man das auch durch Dynamic Programming lösen? Welchen Aufwand hat dieser Algorithmus? (Übung!)

Welche Sites spricht der Broker an?

Sites können ihre Dienste auf zwei Arten anbieten:

Gelbe Seiten: Site trägt ein, welche Dienste es bereit ist auszuführen.
(Beispiel: Scans durch Partitionen A, B, C, Joins, Aggregationen usw.)

Werbekampagnen: Site macht bekannt, daß sie bestimmten Dienst bis zum Expiration Date für zu einem bestimmten Gebot macht. D.h. Site gibt generelles Gebot für jede Art von Anfrage ab.

Vorteil: Broker und Site sparen sich das Einholen des Gebotes für jede individuelle Anfrage.

Nachteil: Site kann nicht mehr so flexibel reagieren.

Wie bestimmen Sites ihre Gebote?

Naive Strategie:

- Verwende unsere Standardkostenformeln, um CPU und Disk Verbrauch der Operation zu bestimmen.
- Berechne so die Antwortzeit der Operation, und berechne den Preis nach einem festen Satz für CPU und Disk IO Verbrauch.
- Diese Vorgehensweise entspricht sehr unserem klassischen Kostenmodell!

Berücksichtige Last der Site:

Eine Site möchte ja niedrige Preise haben, wenn sie nicht ausgelastet ist und relativ hohe Preise haben, wenn sie ausgelastet ist. (Das heißt "Marktwirtschaft"!) Deswegen bestimme Gebot wie folgt:

$$\boxed{\text{Gebot} = \text{Naives Gebot} \times \text{load factor}}$$

Wie kommen unterschiedliche Gebote zustande?

Auswahl anderer Joinmethoden oder Zugriffspfade, die die eine oder andere, stärker oder schwächer beanspruchte Ressource verwenden. Hauptspeicherallokation.

Das *Purchase Order* Protokoll

Das Problem: Richtige Auktionen sind für viele Anfragen zu teuer. (Sotherby's versteigert ja auch keine Zahnbürsten.)

Die Lösung: Der Broker beauftragt eine Site zur Ausführung einer Operation, ohne sie vorher nach einem Gebot zu fragen. Die Site führt die Operation aus, und liefert das Ergebnis zusammen mit einer Rechnung.

Bemerkung: Es kann sein, daß der Broker Verlust macht, da er die Kosten für die Operation nicht im voraus kennt und nicht weiß, wann die Site das Ergebnis liefert. Kann man der Site vertrauen? (C'est la vie.)

Wieso sollte das klappen?

Zwei Fragen:

1. Klappt Marktwirtschaft im richtigen Leben?
2. Was ist mit Monopolen?

Es weiß wirklich niemand, ob das je richtig klappen wird.
Aber gegen Monopole kann man etwas machen:

Replikation

Allokation in Mariposa

Neben der Bearbeitung von Anfragen, können Sites auch Geschäfte mit Daten machen. An- und Verkauf von Partitionen und/oder Kopien von Partitionen; An- und Verkauf von Updateströmen.

Gründe, die zum Ankauf einer Kopie einer Partition führen können.

- Site hat sehr viel freien Speicherplatz.
- Site ist noch nicht ausgelastet und möchte bei Anfragen mit dieser Partition mitmachen.

Gründe, die zum Verkauf einer Kopie einer Partition führen können.

- die Site ist überlastet und kann die Nachfrage ohnehin nicht befriedigen
- Verkauf einer Kopie bringt auf jeden Fall mal Kohle
- die Site macht zusätzlichen Gewinn durch Verkauf des Update Streams

Allokation, Klappe II

Gründe, die zum vollständigen Überlassen der eigenen Kopie führen können.

- Die Site hat keinen Platz mehr.
- Die Site muß hohe Kosten für Updateströme zahlen, und macht folglich Verluste mit der Kopie.
- Die Site glaubt, daß sie höhere Gewinne mit anderen Partitionen machen könnte.
- Wiederum bringt der Verkauf direkt Kohle blank auf die Kralle.

Jede Site führt Statistiken über Updates und Anfragehäufigkeiten und rechnet sich ihren Gewinn aus abhängig davon ob sie eine Kopie einer Partition hat oder nicht.

$\rightarrow$ Auch bei Allokation gilt der Grundsatz:

Jede Site versucht ihren Gewinn zu maximieren.

Dynamische Partitionierung in Mariposa
(Jetzt wird es richtig wild)

- In Mariposa kann eine Site auch beschließen eine (horizontale) Partition ihrer Partition zu verkaufen.
- Die Site macht das wiederum aufgrund eines ökonomischen Modells und mit der Absicht, ihren Gewinn zu maximieren.
- Solche Modelle können allerdings recht unangenehm werden.
- In dem Mariposa Paper ist eine einfache Taktik beschrieben.
  (Wirklich keine Ahnung, wie gut die ist.)

Implementierungsaspekte von Mariposa

- Regelsystem zur Steuerung der Aktionen einer Site.
  (Die sind ja recht kompliziert und irgendjemand muß die ja häcken.)
- Migration von Partitionen: Formatkonvertierung, Einpacken und Recyclen von Indexen.
  (D.h. Index migrieren mit der Partition mit. Da gibt es ein recht hübsches Paper von Paul Aoki auf der CIKM'96.)

. . . und noch etwas:

- Mariposa wird jetzt auch kommerziell.