Das View Selection Problem

Benjamin Daeumlich

Humboldt-Universität zu Berlin

Institut für Informatik

09. Juli 2007

Einführung

- Views (Sichten) werden zum Umschreiben von Querys verwendet
- materialisierte View:
  - im Gegensatz zu klassischer View keine virtuelle Tabelle sondern eine reale Tabelle
  - wird auf Sekundärspeicher gespeichert
- Anwendung:
  - zur Beschleunigung der Query-Bearbeitung
  - im Bereich Data-Warehousing

- **Problem:**

  Welche Views müssen materialisiert werden, damit die Kosten der Ausführung von Queries möglichst gering sind?

- **Problem:**

  Welche Views müssen materialisiert werden, damit die Kosten der Ausführung von Queries möglichst gering sind?

- **Frage:**

  Welche Views kommen in Frage?

- **Problem:**

  Welche Views müssen materialisiert werden, damit die Kosten der Ausführung von Queries möglichst gering sind?

- **Frage:**

  Welche Views kommen in Frage?

- **intuitive Antwort:**

  Views müssen Teilausdrücke der Queries sein.

Beispiel:

- Logistikunternehmen
  - beliefert verschiedene Städte
  - feste Zeitpläne zwischen Paaren von Städten
- Daten in Datenbanktabelle: `T(source, d, dest)`
- Fahrer sollen Touren zugewiesen bekommen, so dass zwei oder mehr Städte beliefert werden können
- Touren starten und enden in der selben Stadt

Beispiel:

- **Tour1:** „Eine Stadt in zwei Tagen“

```
Q1(X1) :- T(X1,D,X2), T(X2,D+1,X1).
```

Beispiel:

- **Tour1:** „Eine Stadt in zwei Tagen“

```
Q1(X1) :- T(X1,D,X2), T(X2,D+1,X1).
```

- **Tour2:** „Fünf Städte in fünf Tagen mit Pause“

```
Q2(X1) :- T(X1,1,X2), T(X2,1,X3), T(X3,2,X4),
          T(X4,2,X5), T(X5,4,X4), T(X4,5,X1).
```

Mögliche Views:

- **View1:**

```
C5(X1) :- T(X1,D1,X2), T(X2,D2,X3), T(X3,D3,X4),
          T(X4,D4,X5), T(X5,D5,X1).
```

  - nur zur Auswertung von `Q2` verwendbar
  - man müsste `C2, C3, C4, .... Cn` materialisieren, aber dafür reicht der Speicher vermutlich nicht aus

Mögliche Views:

- **View1:**

```
C5(X1) :- T(X1,D1,X2), T(X2,D2,X3), T(X3,D3,X4),
          T(X4,D4,X5), T(X5,D5,X1).
```

  - nur zur Auswertung von `Q2` verwendbar
  - man müsste `C2, C3, C4, .... Cn` materialisieren, aber dafür reicht der Speicher vermutlich nicht aus

- **View2:**

```
V10(X1) :- T(X1,D1,X2), T(X2,D2,X3),
           ... , T(X9,D9,X10).
```

  - `Cm` $\subseteq$ `V10` für $2 \leq m \leq 10$

Benutzung zum Umschreiben von `Q1`:

```
Q1(X1) :- V10(X1), T(X1,D,X2), V10(X2),
          T(X2,D+1,X1).
```

- $(V_{10} \bowtie T) \bowtie (V_{10} \bowtie T)$ effizienter als $T \bowtie T$, da Zwischenergebnis $V_{10} \bowtie T$ kleiner als T
- allgemein: V_n
- ursprüngliche Annahme, dass Views Teilausdrücke von Queries sein müssen ist dadurch widerlegt

Definition

Workload

Ein Workload ist eine Abfolge von Queries $\mathcal{Q} = \mathcal{Q}_1, \mathcal{Q}_2, ..., \mathcal{Q}_m$ wobei jedem Query $\mathcal{Q}_i$ ein Gewicht ω_i zugeordnet wird.
Es gilt: $\sum_{i=1}^{m} \omega_i = 1$

Viewkonfiguration

Eine Viewkonfiguration $\mathcal{V}$ ist eine Menge von Views.

Kostenfunktionen

$\mathcal{C}(\mathcal{R}, \mathcal{V}, \mathcal{Q}_i)$: Kosten für die Ausführung von Query $\mathcal{Q}_i$ bei gegebenem Schema $\mathcal{R}$ und Viewkonfiguration $\mathcal{V}$, benutzt zur Abschätzung die durch E approximierten Größen

$E(\mathcal{R}, V)$: Berechnet die Größe einer View V basierend auf den zu $\mathcal{R}$ gehörenden Statistiken $\Sigma_{\mathcal{R}}$.

$\mathcal{C}(\mathcal{R}, \mathcal{V}, \mathcal{Q}) = \sum_{i=1}^{m} \mathcal{C}(\mathcal{R}, \mathcal{V}, \mathcal{Q}_i) \times \omega_i$: Gesamtkosten für die Ausführung des Workload $\mathcal{Q}$ bei gegebenem Schema $\mathcal{R}$ und Viewkonfiguation $\mathcal{V}$

Definition

Sei $\mathcal{R}$ ein Datenbankschema, $\mathcal{B}$ der verfügbare Speicherplatz, $\mathcal{Q}$ ein Workload, $\mathcal{C}(\mathcal{R}, \mathcal{V}, \mathcal{Q})$ eine Kostenabschätzungsfunktion.

Das Problem, eine auf $\mathcal{R}$ definierte Viewkonfiguration $\mathcal{V}$ zu finden, deren Größe durch $\mathcal{B}$ beschränkt ist und die die Kostenfunktion $\mathcal{C}(\mathcal{R}, \mathcal{V}, \mathcal{Q})$ minimiert, nennt man View Selection Problem.

Folgerungen:

- View Selection Problem ist Optimierungsproblem
- zugehöriges Entscheidungsproblem: Gibt es eine Viewkonfiguration $\mathcal{V}$, so dass der Wert der Kostenabschätzungsfunktion $\mathcal{C}(\mathcal{R}, \mathcal{V}, \mathcal{Q})$ kleiner als eine Zahl k ist.
- Eingabegrößen beinhalten keine Datenbankinstanz

Das View Selection Problem unter der „Complete Statistics Assumption“

Grundannahmen:

- Workload: SPJ-Queries
- Orakel kann genau sagen, wie groß jede View ist
- die Funktion E ist Monoton
  d.h.: $V_1 \subseteq V_2 \rightarrow E(\mathcal{R}, V_1) \leq E(\mathcal{R}, V_2)$
- Orakel kann in der Praxis durch viele Arbeitsschritte und die dadurch aufgestellten Statisiken approximiert werden

Komplexität:

- Größe einer View: exponentiell begrenzt durch das größte Teilergebnis aller Queries im Workload
- Anzahl der Views: exponentiell begrenzt in der Größe von $\mathcal{R}$ und $\mathcal{Q}$
- Anzahl der Views ohne Projektion: quadratisch begrenzt in der Größe von $\mathcal{R}$ und $\mathcal{Q}$
- Entscheidbarkeit: in 4-fach exponentieller Zeit

Das View Selection Problem unter der „Partial Statistics Assumption“

Grundannahmen:

- praxisrelevant
- Ergebnisse stark abhängig von Funktion E:
  - E bestimmt „künstlich“ große Views: Problem trivial (nur Relationen als Views materialisieren oder gar nichts, wenn Zugriff auf Relationen stets gewährleistet)
  - E bestimmt „künstlich“ kleine Views: Problem komplex
- Annahme: Benutzung von E-Funktionen aus der Praxis

Praxisrelevante E-Funktionen:

- nur multiplikative Schätzfunktionen
  - $\Sigma_{\mathcal{R}} = \{c_1, ..., c_p, N_1, ..., N_q\}$
  - $E(\mathcal{R}, V) = c_1^{\gamma_1} \times ... \times c_p^{\gamma_p} \times N_1^{\delta_1} \times ... \times N_q^{\delta_q}$
  - aber: so gibt es Konstruktionen, bei denen die erzeugte Viewkonfiguration nicht optimal ist
- Annahme großer Kardinalitäten
  - festgesetzter Mindestwert L für $E(\mathcal{R}, V)$
  - wenn Wert von $E(\mathcal{R}, V)$ kleiner als L: Wert entweder 0 oder L
- weitere Einschränkungen

Komplexität:

- Größe einer View: linear abhängig vom größten Teilergebnis aller Queries im Workload
- Anzahl der Views: polynomiell begrenzt in der Größe von $\mathcal{R}$ und $\mathcal{Q}$
- Entscheidbarkeit: in exponentieller Zeit

Vergleich:

	CSA	**PSA**
Größe einer View	exponentiell begrenzt durch das größte Teilergebnis aller Queries im Workload	linear abhängig vom größten Teilergebnis aller Queries im Workload
Anzahl der Views	exponentiell begrenzt in der Größe von $\mathcal{R}$ und $\mathcal{Q}$	polynomiell begrenzt in der Größe von $\mathcal{R}$ und $\mathcal{Q}$
Entscheidbarkeit	in 4-fach exponentieller Zeit	in exponentieller Zeit

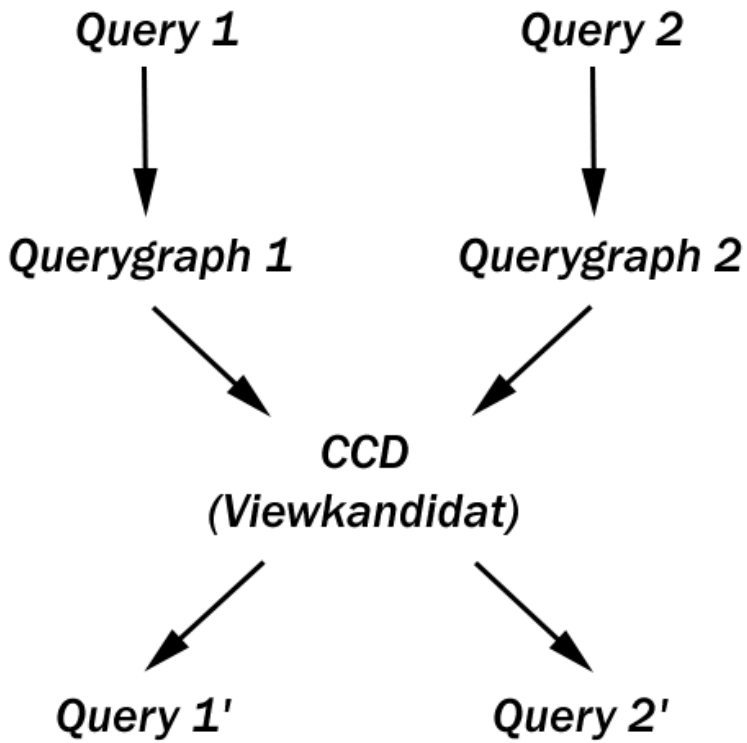
Lösungsansatz

- Ziel: Viewkandidaten erzeugen, basierend auf den Queries im Workload
- dadurch: Verkleinerung des Suchraumes
- durch Konzept des „Closest Common Derivator“ (CCD)
  - maximale Gemeinsamkeit zwischen zwei Queries
  - d.h.: so viele Operationen wie möglich (SPJ)

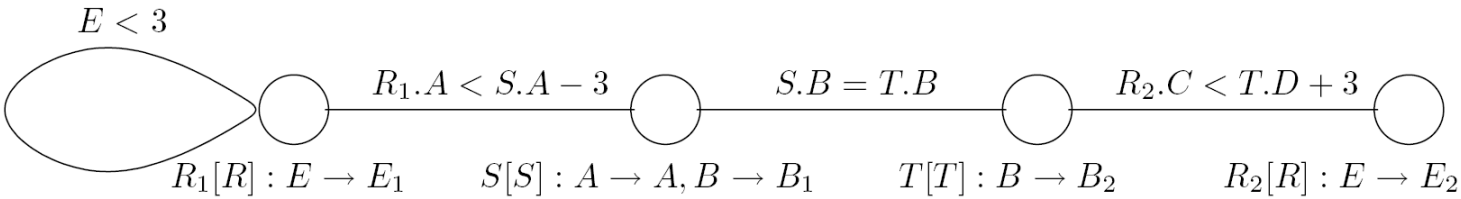
Herangehensweise:

- Bestimmung eines CCD:
  - geschieht mittels Querygraphen
  - aus denen entsteht durch Anwendung gewisser Regeln ein CCD
  - CCD entspricht dem Viewkandidaten
  - wichtig: CCD kann mehr Knoten haben als die Ausgangsqueries, dies hat sich in der Praxis als vorteilhaft erwiesen
- Queries lassen sich dann nachweislich durch die aus ihnen generierten CCDs umschreiben

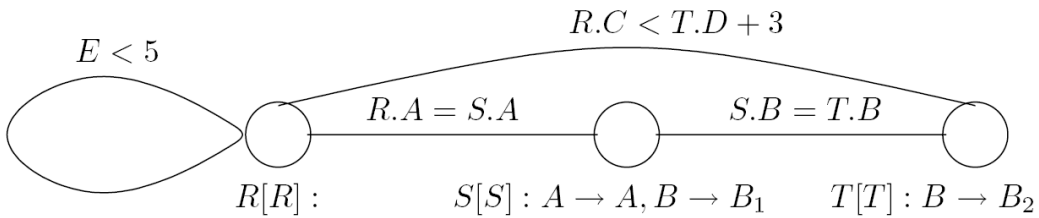
Herangehensweise:



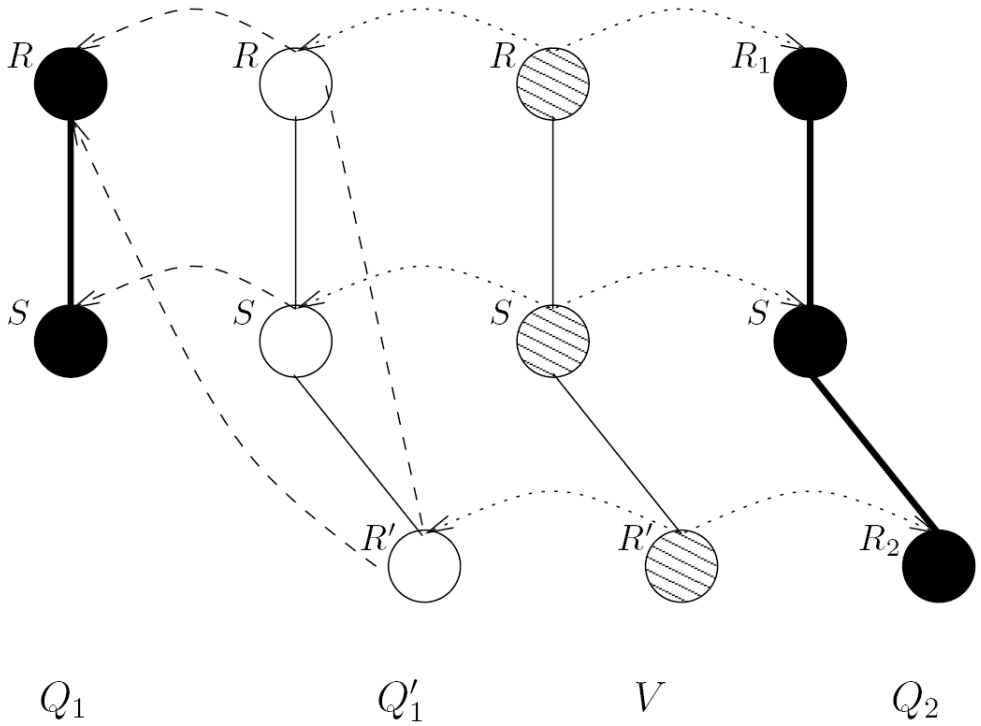
Query 1:



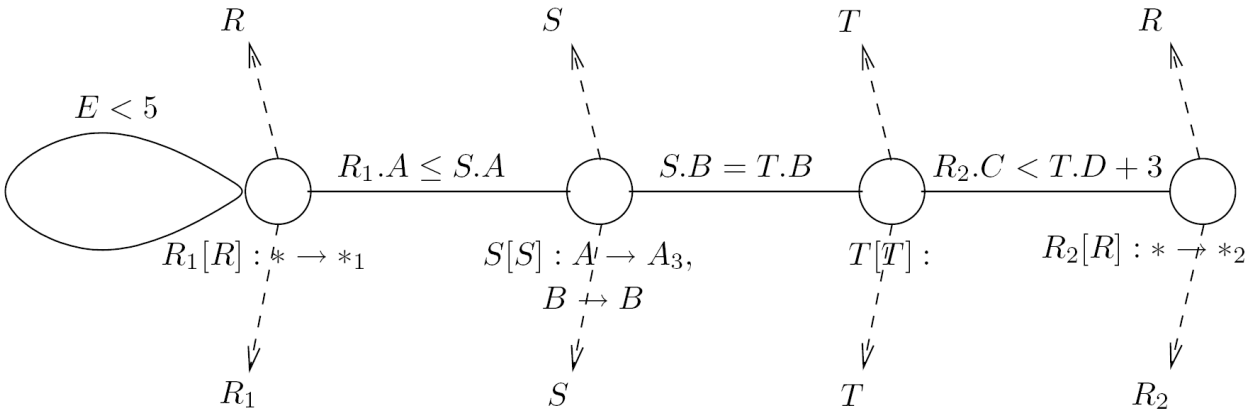
Query 2:



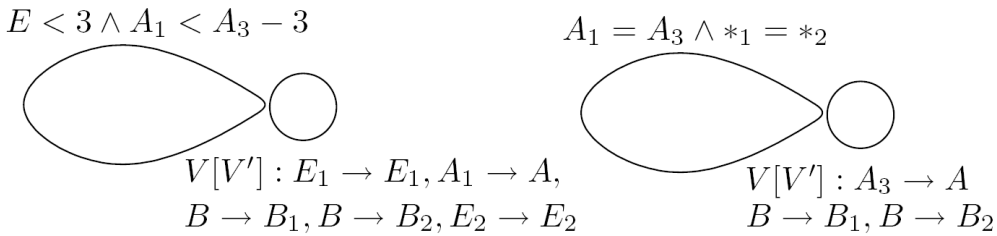
Erstellung des CCD aus 2 Querygraphen:



CCD (Viewkandidat):



Umschreibung von Query 1 und Query 2 mittels V:



Referenzen

 *Rada Chirkova, Alon Y. Halevy, Dan Suciu: A Formal Perspective on the View Selection Problem*

 *Wugang Xu, Mimitri Theodoratos, Calisto Zuzarte: Computing Closest Common Subexpressions for View Selection Problems*

 *Wugang Xu, Mimitri Theodoratos: Constructing Search Spaces for Materialized View Selection*

 *Howard Karloff, Milena Mihail: On the complexity of the view-selection problem*

 *Rada Chirkova: The view-selection problem has an exponential-time lower bound for conjunctive queries and views*

Fragen ???