Anfrageoptimierung in Datenbanken – Theorie und Praxis

Vorlesung im WS 2017/18, HU Berlin,

Prof J.C. Freytag & Prof. N. Schweikardt

Theorieteil

Prof. N. Schweikardt

Kapitel 1: Grundbegriffe

- $\mathbb{N} := \{0,1,2,3,\dots\}$
- $\mathbb{N}_{\geq 1} := \mathbb{N} \setminus \{0\}$

Für $m,n \in \mathbb{N}$ ist

- $[m,n] := \{ i \in \mathbb{N} : m \leq i \leq n \}$
- $[n] := [1,n]$

Ist M eine Menge, so schreiben wir $X \subseteq_e M$, um auszudrücken, dass X eine endliche Teilmenge von M ist.

Wir schreiben $\mathcal{P}(M)$ oder 2^M, um die Potenzmenge von M zu bezeichnen.

Für $k \in \mathbb{N}$ ist

$$\binom{M}{k} := \{ X \subseteq M : |X| = k \}$$

die Menge aller k-elementigen Teilmengen von M,

und $\binom{M}{\leq k} := \bigcup_{i=0}^{k} \binom{M}{k}$

Für ein Tupel $t = (t_1, \dots, t_k)$ und Zahlen $i_1, \dots, i_m \in [k]$ ist $\pi_{i_1,\dots,i_m}(t) := (t_{i_1}, \dots, t_{i_m})$.

Abkürzungen:

- "f.a." steht für "für alle"
- "ex." steht für "es existiert" bzw "es gibt"
- "s.d." steht für "so dass"
- "DB" steht für "Datenbank"

1.1 Datenbanken

Für den gesamten Theorieteil sei

dom eine fest gewählte, abzählbar unendliche Menge. Elemente in **dom** werden auch Konstanten genannt; wir bezeichnen sie oft mit Kleinbuchstaben wie a, b, c.

Ein (Datenbank-) Schema σ ist eine endliche Menge von Relationssymbolen (auch: Relationsnamen), wobei jedem $R \in \sigma$ eine feste Stelligkeit $ar(R) \in \mathbb{N}$ zuordnen.

Eine Datenbank vom Schema σ (kurz: σ-DB) besteht aus einer Relation $R^D \subseteq_e \text{dom}^{ar(R)}$ für jedes $R \in \sigma$.

Für $\sigma = \{R_1, \dots, R_s\}$ schreiben wir auch $D = (R_1^D, \dots, R_s^D)$, um eine σ-DB zu bezeichnen.

Der active domain $adom(D)$ von D ist die kleinste Menge $A \subseteq \text{dom}$, für die gilt: $R^D \subseteq A^{ar(R)}$ f.a. $R \in \sigma$.

1.2 Konjunktive Anfragen

Sei var eine fest gewählte, abzählbar unendliche Menge mit $\text{var} \cap \text{dom} = \emptyset$.

Elemente in var nennen wir Variablen; wir bezeichnen sie oft mit Großbuchstaben wie $A, B, C, X, Y, \ldots$

Ein Term ist ein Element aus $\text{var} \cup \text{dom}$.
Ein freies Tupel der Stelligkeit $k \in \mathbb{N}$ ist ein Element aus $(\text{var} \cup \text{dom})^k$

Sei σ ein Schema.
Ein σ-Atom α ist von der Form $R(u_1, \ldots, u_r)$ mit $R \in \sigma$, $r = ar(R)$ und $u_1, \ldots, u_r \in \text{var} \cup \text{dom}$.
Wir setzen $\text{vars}(\alpha) := \{u_1, \ldots, u_r\} \cap \text{var}$ und $\text{cons}(\alpha) := \{u_1, \ldots, u_r\} \cap \text{dom}$.

Eine (regelbasierte) konjunktive Anfrage (kurz: CQ, für "conjunctive query") Q vom Schema σ ist von der Form

$$Q(v_1, \dots, v_k) \leftarrow \alpha_1, \dots, \alpha_m \qquad (*)$$

mit
- $k \geq 0$, $m \geq 1$, $v_1, \dots, v_k \in \text{var} \cup \text{dom}$,
- für jedes $i \in [m]$ ist α_i ein σ-Atom und
- $\{v_1, \dots, v_k\} \cap \text{var} \subseteq \bigcup_{i \in [m]} \text{vars}(\alpha_i)$.

Eine Anfrage der Form $(*)$ wird auch Regel genannt; das "Atom" $Q(v_1, \dots, v_k)$ heißt Kopf der Regel; "$\alpha_1, \dots, \alpha_m$" heißt Rumpf der Regel; die Zahl k ist die Stelligkeit der Anfrage.

Mit $\text{vars}(Q)$ (bzw. $\text{cons}(Q)$) bezeichnen wir die Menge aller Variablen (bzw. Konstanten), die in Q vorkommen. D.h.:

- $\text{vars}(Q) = \bigcup_{i \in [m]} \text{vars}(\alpha_i) = \left(\{v_1, \dots, v_k\} \cap \text{var}\right) \cup \bigcup_{i \in [m]} \text{vars}(\alpha_i)$
- $\text{cons}(Q) = \left(\{v_1, \dots, v_k\} \cap \text{dom}\right) \cup \bigcup_{i \in [m]} \text{cons}(\alpha_i)$.

Eine <u>Belegung</u> für Q ist eine Abbildung

$\beta : \text{vars}(Q) \cup \text{dom} \to \text{dom}$ mit

$\beta(a) = a$ f.a. $a \in \text{dom}$.

Eine Belegung β für Q ist ein
<u>Homomorphismus von Q auf eine σ-DB D</u>
(kurz: $\beta : Q \to D$), falls für jedes
Atom α der Form $R(u_1, \dots, u_r)$ im Rumpf von Q
gilt: $\big(\beta(u_1), \dots, \beta(u_r) \big) \in D^R$.

Das <u>Anfrageergebnis</u> (kurz: Ergebnis, Resultat)
von Q auf D ist die Menge

$$Q(D) := \left\{ \big(\beta(v_1), \dots, \beta(v_k)\big) : \begin{array}{l} \beta \text{ ist ein} \\ \text{Homomorphismus} \\ \text{von } Q \text{ auf } D \end{array} \right\}$$

<u>Beobachtung 1.1</u>

$$Q(D) \subseteq \big(\underbrace{\text{adom}(D) \cup \text{cons}(Q)}_{=:\ \text{adom}(Q,D)} \big)^k$$

Beispiel 1.2

Sei $\sigma_E := \{E\}$ mit $ar(E) = 2$.

Eine σ_E-DB $\mathcal{D}$ besteht aus einer 2-stelligen Relation $E^{\mathcal{D}} \subseteq_e dom \times dom$.

Wir können sie z.B. als "soziales Netzwerk" interpretieren, bei dem ein Tupel $(a,b) \in E^{\mathcal{D}}$ besagt "Person a hat die Nachrichten von Person b abonniert" — kurz: Person a ist ein "Follower" von Person b. Wir können uns $\mathcal{D}$ also vorstellen als Datenbank, die aus einer Tabelle $E^{\mathcal{D}}$ der Form

Follower	Author
...	...

(b) Gesucht: Eine Anfrage Q_Δ, s.d. auf jeder σ_E-DB $\mathcal{D}$ gilt:

$Q_\Delta(\mathcal{D})$ besteht aus genau denjenigen Tripeln (a,b,c), für die gilt: a ist Follower von b,
b ist Follower von c und
c ist Follower von a

In SQL:

```
SELECT DISTINCT E1.Follower, E2.Follower, E3.Follower
FROM E AS E1, E AS E2, E AS E3
WHERE E1.Author = E2.Follower AND
      E2.Author = E3.Follower AND
      E3.Author = E1.Follower
```

Als CQ:

$$Q_\Delta(A,B,C) \leftarrow E(A,B),\ E(B,C),\ E(C,A)$$

(a) Gesucht: Eine Anfrage $Q_{(b)}$, s.d. auf jeder σ_E-DB D gilt: $Q_{(b)}(D)$ besteht aus genau denjenigen 1-Tupeln (a), für die gilt: a ist Follower von Sascha Lobo und von Mesut Özil

Als CQ:

$$Q_{(b)}(A) \leftarrow E(A, \text{Sascha Lobo}),\ E(A, \text{Mesut Özil})$$

In SQL:

```
SELECT DISTINCT E1.Follower
FROM E AS E1, E AS E2
WHERE E1.Author = 'Sascha Lobo' AND
      E2.Author = 'Mesut Özil" AND
      E1.Follower = E2.Follower
```

1.3 Das Auswertungsproblem

Das Auswertungsproblem für CQ ist das Berechnungsproblem mit

Eingabe: Ein Schema σ, eine σ-DB D und eine CQ Q vom Schema σ

Ausgabe: $Q(D)$

Beobachtung 1.3

Das Auswertungsproblem für CQ lässt sich mit dem folgenden einfachen, aber nicht gerade effizienten Algorithmus lösen:

Bei Eingabe von σ, D, Q

1) Berechne $s := |\mathrm{vars}(Q)|$ und $\{X_1, \ldots, X_s\} = \mathrm{vars}(Q)$
2) Berechne $N := \mathrm{adom}(Q,D)$ und $\{a_1, \ldots, a_N\} = \mathrm{adom}(Q,D)$
3) Ergebnis $:= \emptyset$
4) Für jedes $(b_1, \ldots, b_s) \in \{a_1, \ldots, a_N\}^s$ tue Folgendes:

   Sei β die Belegung für Q mit $\beta(X_i) = b_i$ f.a. $i \in [s]$.

   Teste, ob für jedes Atom α der Form $R(u_1, \ldots, u_r)$ im Rumpf von Q gilt: $(\beta(u_1), \ldots, \beta(u_r)) \in R^D$.

   Falls ja, so Ergebnis $:=$ Ergebnis $\cup \{(\beta(v_1), \ldots, \beta(v_k))\}$ (wobei der Kopf von Q von der Form $Q(v_1, \ldots, v_k)$ ist).
5) Gib Ergebnis aus.

Unter Verwendung von Beobachtung 1.1 kann man sich leicht davon überzeugen, dass dieser Algorithmus tatsächlich $Q(D)$ ausgibt.
Die Laufzeit wird im Wesentlichen dominiert durch die Anzahl N^s der Schleifendurchläufe in Zeile 4).

<u>Satz 1.4</u> (Satz von Chandra und Merlin, 1977)

Das Auswertungsproblem für 0-stellige konjunktive Anfragen ist NP-vollständig.

<u>Beweisidee:</u>

<u>Zugehörigkeit zu NP:</u>

Bei Eingabe von σ, D, Q, führe die Schritte 1), 2), 3) aus Beobachtung 1.3 durch.
An Stelle von Schritt 4) tue Folgendes:

4) Rate ein Tupel $(b_1, \dots, b_s) \in \{a_1, \dots, a_N\}^s$.

Betrachte die Belegung β mit $\beta(x_i) = b_i$ f.a. $i \in [s]$.
Teste, ob für jedes Atom α der Form $R(u_1, \dots, u_r)$ im Rumpf von Q gilt: $(\beta(u_1), \dots, \beta(u_r)) \in R^D$.

Falls ja, so Ergebnis $:= \{()\}$.

5) Gib Ergebnis aus

Man kann sich leicht davon überzeugen, dass dies ein nichtdeterministischer Polynomialzeit-Algorithmus ist, der das Auswertungsproblem für CQ löst.

NP-Härte:

Wir reduzieren das NP-vollständige Problem

CLIQUE

Eingabe: Ein ungerichteter endlicher Graph $G = (V(G), E(G))$ und eine Zahl $k \in \mathbb{N}$.

Frage: Besitzt G eine Clique der Größe k, d.h. gibt es Knoten $v_1, \dots, v_k \in V(G)$, s.d. f.a. $i,j \in [k]$ mit $i \neq j$ gilt: $\{v_i, v_j\} \in E(G)$?

aufs Auswertungsproblem für CQ:

Für eine Eingabe (G, k) für's CLIQUE-Problem konstruieren wir eine Eingabe (σ, D, Q) für's Auswertungsproblem für CQ, s.d. gilt:

G besitzt eine Clique der Größe k $\iff$ $Q(D) \ni \{()\}$.

$\Updownarrow$ Def. Semantik von CQs

es gibt einen Homomorphismus $\beta : Q \to D$

Dazu wählen wir $\sigma := \sigma_E = \{E\}$,

$D := (E^D)$ mit $E^D := \{ (a,b) : \{a,b\} \in E(G) \}$ und

$Q := Q_{\min\{k, |V(G)|+1\}}$, wobei für jedes $k' \geq 1$ die Anfrage $Q_{k'}$ die CQ mit Kopf $Q_{k'}()$ ist, deren Rumpf aus allen Atomen $E(x_i, x_j)$ mit $1 \leq i < j \leq k'$ besteht, wobei $x_1, \dots, x_{k'}$ k' verschiedene Variablen sind.

Beispiel: $Q_3() \leftarrow E(x_1,x_2), E(x_2,x_3), E(x_1,x_2)$

Man sieht leicht, dass gilt

$Q_{k'}(D) \ni \{()\} \iff G$ besitzt eine Clique der Größe k'.

Somit gilt auch:

$Q(D) \ni \{()\} \iff G$ besitzt eine Clique der Größe k.

Außerdem kann (σ, D, Q) bei Eingabe von (G,k) deterministisch in Zeit polynomiell in der Größe von G und (der Binärdarstellung von) k erzeugt werden.

Somit haben wir eine Polynomialzeit-Reduktion vom CLIQUE-Problem auf's Auswertungsproblem für CQ konstruiert.

$\square$

Folgerung 1.5

Falls $P \neq NP$ ist, so gibt es keinen Algorithmus, der das Auswertungsproblem für 0-stellige CQ-Anfragen deterministisch in Zeit $(k+n)^{O(1)}$ löst, wobei k die Größe der Anfrage und n die Größe der Datenbank ist

Bemerkung 1.6

Unter Verwendung einer stärkeren Annahme aus der Parametrisierten Komplexitätstheorie lässt sich sogar Folgendes zeigen:

Satz von Papadimitriou und Yannakakis, 1997

Falls FPT $\neq$ W[1], so gibt es keinen Algorithmus, der das Auswertungsproblem für 0-stellige CQ-Anfragen deterministisch in Zeit in Zeit $f(k) \cdot n^c$ löst, wobei c irgendeine natürliche Zahl und f irgendeine berechenbare Funktion ist, und k die Größe der Anfrage und n die Größe der Datenbank bezeichnet.

FPT und W[1] sind Komplexitätsklassen, die in der parametrisierten Komplexitätstheorie Rollen spielen, die mit den Rollen von P und NP in der klassischen Komplexitätstheorie vergleichbar sind.