Datenbanksysteme WS 2017/18
Prof. Dr.-Ing. Sebastian Michel
Manuel Hoffmann, Marc Dörr
TU Kaiserslautern, FB Informatik – Lehrgebiet Informationssysteme
http://dbis.informatik.uni-kl.de
Übungsblatt 4: Ausgabe 21.11.2017, Abgabe 27.11.2017 16:00 MEZ

Aufgabe 1: Effektivität von Heuristiken (1 P.)

Schreiben Sie in einer Programmiersprache Ihrer Wahl ein Programm, das die Effektivität von Join-Ordering-Heuristiken misst. Dazu brauchen Sie jeweils eine Funktion für jeden der drei Greedy Heuristiken, sowie eine Funktion, die den besten Plan zurückgibt und eine Funktion, die den schlechtesten Plan zurück gibt. Für letztere beide durchsuchen Sie den kompletten Suchraum. Die Effektivität messen wir in verursachten C_{out}-Kosten.

Vergleichen Sie mit Ihrem Programm, wie gut die gefundenen Pläne für Relationen $|R_1| = 100, |R_2| = 500, |R_3| = 200, |R_4| = 10$ mit Selektivitäten $j_{1,2} = 1/200$, $j_{2,4} = 1/5$, $j_{3,4} = 1/20$ sind. Kreuzprodukte seien nicht zugelassen.

Aufgabe 2: Join-Ordering (1 P.)

a) **Links-tiefe Bäume vs. Buschige Bäume**
Gegeben seien die Relationen R_1, R_2, R_3 und R_4, mit $|R_1| = 10$, $|R_2| = 100$, $|R_3| = 20$, $|R_4| = 40$, sowie die Join-Selektivitäten $j_{1,2} = 0{,}10$, $j_{2,3} = 0{,}20$, $j_{2,4} = 0{,}10$.

   i) Zeichnen Sie den Anfragegraphen.

   ii) Bestimmen Sie alle links-tiefen Join-Bäume ohne Kreuzprodukte.

   iii) Bestimmen Sie die Kosten C_{out} für die in (ii) bestimmten Join-Bäume, in denen R_4 als letztes (d.h. oben) gejoint wird.

   iv) Gibt es einen buschigen Join-Baum, auch ohne Kreuzprodukte, der kein linearer Baum ist und geringere Kosten hat?

b) **Greedy Heuristiken**
Geben Sie ein Beispiel mit 3 Relationen an, für das der Greedy-Algorithmus-1 weit neben der optimalen Lösung liegt, Greedy-Algorithmus-3 hingegen die optimale Lösung findet.

Aufgabe 3: DP-Algorithmus für Ketten-Anfragen (1 P.)

a) Spezifizieren Sie einen einfachen DP-Algorithmus, der den optimalen Anfragebaum für Kettenanfragen ohne Kreuzprodukte in $O(n^3)$ findet (Pseudocode).

b) Implementieren Sie den Algorithmus in einer Programmiersprache Ihrer Wahl und führen Sie ihn auf folgendem Problem aus: Für 6 Relationen als Kette R_1—0,1—R_2—0,7—R_3—0,2—R_4—0,3—R_5—0,4—R_6 mit Kardinalitäten (von R_1 nach R_6) 20, 10, 20, 20, 10, 20 und Selektivitäten wie in der Kette zwischen den Relationen angegeben.
Geben Sie Ihren Code und die berechnete Lösung an.

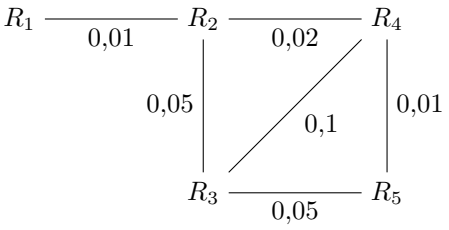
Aufgabe 4: Join-Ordering: Dynamische Programmierung (1 P.)

a) Erzeugen Sie manuell die DP-Tabelle für die Relationen A, B und C mit Kardinalitäten $|A| = 10$, $|B| = 20$, $|C| = 100$ und Selektivitäten $f_{A,B} = 0.5$, $f_{B,C} = 0.1$ unter der Kostenfunktion C_{out}. Kreuzprodukte seien zugelassen. Markieren Sie die finalen Einträge der Tabelle und behalten Sie auch die eliminierten Einträge bei, also die Einträge, die durch günstigere ersetzt worden sind.

b) Gegeben die folgende DP-Tabelle mit Zwischenergebnissen und der Anfragegraph, in welchem die einzelnen Selektivitäten als Kantenbeschriftung notiert sind:

Relationen	T	$C_{out}(T)$	$\|T\|$
$\{R_1, R_2\}$	$(R_1 \bowtie R_2)$	2	2
$\{R_1, R_3\}$	$(R_1 \bowtie R_3)$	200	200
$\{R_1, R_4\}$	$(R_1 \bowtie R_4)$	500	500
$\{R_1, R_5\}$	$(R_1 \bowtie R_5)$	500	500
$\{R_2, R_3\}$	$(R_2 \bowtie R_3)$	20	20
$\{R_2, R_4\}$	$(R_2 \bowtie R_4)$	20	20
$\{R_2, R_5\}$	$(R_2 \bowtie R_5)$	1000	1000
$\{R_3, R_4\}$	$(R_3 \bowtie R_4)$	100	100
$\{R_3, R_5\}$	$(R_3 \bowtie R_5)$	50	50
$\{R_4, R_5\}$	$(R_4 \bowtie R_5)$	25	25
$\{R_1, R_2, R_3\}$	$((R_1 \bowtie R_2) \bowtie R_3)$	4	2
$\{R_1, R_2, R_4\}$	$((R_1 \bowtie R_2) \bowtie R_4)$	4	2
$\{R_1, R_2, R_5\}$	$((R_1 \bowtie R_2) \bowtie R_5)$	102	100
$\{R_1, R_3, R_4\}$	$((R_3 \bowtie R_4) \bowtie R_1)$	1100	1000
$\{R_1, R_3, R_5\}$	$((R_3 \bowtie R_5) \bowtie R_1)$	550	500
$\{R_1, R_4, R_5\}$	$((R_4 \bowtie R_5) \bowtie R_1)$	275	250
$\{R_2, R_3, R_4\}$	$((R_2 \bowtie R_3) \bowtie R_4)$	22	2
$\{R_2, R_3, R_5\}$	$((R_2 \bowtie R_3) \bowtie R_5)$	70	50
$\{R_2, R_4, R_5\}$	$((R_2 \bowtie R_4) \bowtie R_5)$	30	10
$\{R_3, R_4, R_5\}$	$((R_4 \bowtie R_5) \bowtie R_3)$	27,5	2,5

- $|R_1| = 10$
- $|R_2| = 20$
- $|R_3| = 20$
- $|R_4| = 50$
- $|R_5| = 50$



Berechnen Sie den optimalen buschigen Join-Baum für $\{R_1, R_2, R_3, R_4\}$ mit dem in der Vorlesung vorgestellten DP-Algorithmus.