Optimizations Based on Semijoins

Semi-join based optimizations

• $R \bowtie S = \Pi_{A_1,\ldots,A_n} (R >> S)$
• Where the schemas are:
  – Input: $R(A_1,\ldots,A_n)$, $S(B_1,\ldots,B_m)$
  – Output: $T(A_1,\ldots,A_n)$

• Example:
  
  $Q \cdot R_1(A,B), R_2(B,C), R_3(C,D)$

  A full reducer is:

  $R_2(B,C) \Rightarrow R_2(B,C), R_1(A,B)$
  $R_3(C,D) \Rightarrow R_3(C,D), R_2(B,C)$
  $R_2(B,C) \Rightarrow R_2(B,C), R_3(C,D)$
  $R_1(A,B) \Rightarrow R_1(A,B), R_2(B,C)$

• Example:
  
  $Q \cdot R_1(A,B), R_2(B,C), R_3(A,C)$

  Doesn’t have a full reducer (we can reduce forever)

  **Theorem** a query has a full reducer iff it is “acyclic”
Optimizations Based on Semijoins

• Semijoins in [Chaudhuri’98]

CREATE VIEW DepAvgSal AS (  
SELECT E.did, Avg(E.Sal) AS avgSal  
FROM Emp E  
GROUP BY E.did)  
SELECT E.id, E.sal  
FROM Emp E, Dept D, DepAvgSal V  
WHERE E.did = D.did AND E.did = V.did  
AND E.age < 30 AND D.budget > 100k  
AND E.sal > V.avgSal

Optimizations Based on Semijoins

• First idea:

CREATE VIEW LimitedAvgSal AS (  
SELECT E.did, Avg(E.Sal) AS avgSal  
FROM Emp E, Dept D  
WHERE E.did = D.did AND D.budget > 100k  
GROUP BY E.did)  
SELECT E.id, E.sal  
FROM Emp E, Dept D, LimitedAvgSal V  
WHERE E.did = D.did AND E.did = V.did  
AND E.age < 30 AND D.budget > 100k  
AND E.sal > V.avgSal

Optimizations Based on Semijoins

• Better: full reducer

CREATE VIEW PartialResult AS  
(SELECT E.id, E.sal, E.did  
FROM Emp E, Dept D  
WHERE E.did = D.did AND E.age < 30  
AND D.budget > 100k)  
CREATE VIEW Filter AS  
(SELECT DISTINCT P.did FROM PartialResult P)  
CREATE VIEW LimitedAvgSal AS  
(SELECT E.did, Avg(E.Sal) AS avgSal  
FROM Emp E, Filter F  
WHERE E.did = F.did GROUP BY E.did)

Optimizations Based on Semijoins

Modern Query Optimizers

• Volcano
  – Rewrite rules
  – Extensible

• Starburst
  – Keeps query blocks
  – Interblock, intrablock optimizations

Size Estimation

The problem: Given an expression E, compute T(E) and V(E, A)

• This is hard without computing E
• Will ‘estimate’ them instead
Size Estimation

Estimating the size of a projection

• Easy: $T(\Pi_A(R)) = T(R)$
• This is because a projection doesn’t eliminate duplicates

Size Estimation

Estimating the size of a selection

• $S = \sigma_{A < c}(R)$
  – $T(S)$ can be anything from 0 to $T(R) - V(R,A) + 1$
  – Estimate: $T(S) = T(R) / V(R,A)$
  – When $V(R,A)$ is not available, estimate $T(S) = T(R)/10$

• $S = \sigma_{A = c}(R)$
  – $T(S)$ can be anything from 0 to $T(R)$
  – Estimate: $T(S) = (c - Low(R,A)) / (High(R,A) - Low(R,A))$
  – When Low, High unavailable, estimate $T(S) = T(R)/3$

Size Estimation

Estimating the size of a natural join, $R \times_A S$

• When the set of $A$ values are disjoint, then $T(R \times_A S) = 0$
• When $A$ is a key in $S$ and a foreign key in $R$, then $T(R \times_A S) = T(R)$
• When $A$ has a unique value, the same in $R$ and $S$, then $T(R \times_A S) = T(R) \cdot T(S)$

Size Estimation

Assumptions:

• Containment of values: if $V(R,A) \subseteq V(S,A)$, then the set of $A$ values of $R$ is included in the set of $A$ values of $S$
  – Note: this indeed holds when $A$ is a foreign key in $R$, and a key in $S$
• Preservation of values: for any other attribute $B$, $V(R \times_A S, B) = V(R, B)$ (or $V(S, B)$)

Size Estimation

Assume $V(R,A) \subseteq V(S,A)$

• Then each tuple $t$ in $R$ joins some tuple(s) in $S$
  – How many?
    – On average $T(S)/V(S,A)$
    – $t$ will contribute $T(S)/V(S,A)$ tuples in $R \times_A S$
• Hence $T(R \times_A S) = T(R) \cdot T(S) / V(S,A)$

In general: $T(R \times_A S) = T(R) \cdot T(S) / \max(V(R,A), V(S,A))$

Size Estimation

Example:

• $T(R) = 10000$, $T(S) = 20000$
• $V(R,A) = 100$, $V(S,A) = 200$
• How large is $R \times_A S$?

Answer: $T(R \times_A S) = 10000 \cdot 20000/200 = 1M$
Size Estimation

Joins on more than one attribute:

\[ T(R \mid_{A,B} S) = \frac{T(R)}{\max(V(R,A),V(S,A)) \times \max(V(R,B),V(S,B))} \]

Histograms

- Statistics on data maintained by the RDBMS
- Makes size estimation much more accurate (hence, cost estimations are more accurate)

Employee(ssn, name, salary, phone)

- Maintain a histogram on salary:

<table>
<thead>
<tr>
<th>Salary</th>
<th>0-20k</th>
<th>20k-40k</th>
<th>40k-60k</th>
<th>60k-80k</th>
<th>80k-100k</th>
<th>&gt;100k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuples</td>
<td>200</td>
<td>500</td>
<td>2000</td>
<td>1000</td>
<td>500</td>
<td>50</td>
</tr>
</tbody>
</table>

  - \( T(Employee) = 25000 \), but now we know the distribution

Ranks(rankName, salary)

- Estimate the size of Employee \mid Salary Ranks

<table>
<thead>
<tr>
<th>Employee</th>
<th>0-20k</th>
<th>20k-40k</th>
<th>40k-60k</th>
<th>60k-80k</th>
<th>80k-100k</th>
<th>&gt;100k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuples</td>
<td>200</td>
<td>500</td>
<td>2000</td>
<td>1000</td>
<td>500</td>
<td>50</td>
</tr>
</tbody>
</table>

Histograms

- Eqwidth
- Eqdepth

End of “Normal” Lectures

- What’s next?
- Two lectures on advanced topics:
  - Queries with uncertainties
  - Security issues in data sharing
- Projects presentations