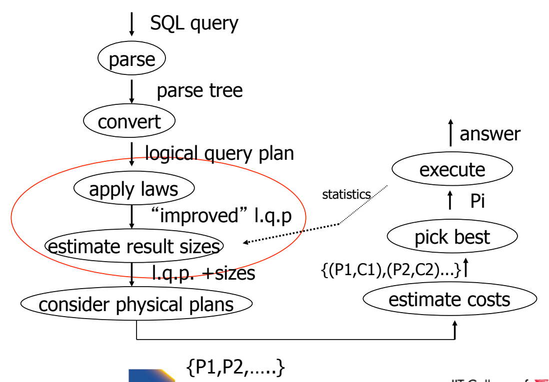
CS 525: Advanced Database Organisation

09: Query Optimization Logical

Boris Glavic

Slides: adapted from a course taught by Hector Garcia-Molina, Stanford InfoLab



Query Optimization

- Relational algebra level
- Detailed query plan level

Query Optimization

- Relational algebra level
- Detailed query plan level
  - Estimate Costs
    - without indexes
    - with indexes
  - Generate and compare plans

Relational algebra optimization

- Transformation rules (preserve equivalence)
- What are good transformations?
  - Heuristic application of transformations

Query Equivalence

- Two queries q and q' are equivalent:
  - If for every database instance I
    - Contents of all the tables
  - Both queries have the same result

$q \equiv q'$ iff $\forall I: q(I) = q'(I)$

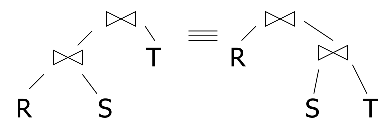
Rules: Natural joins & cross products & union

$R \bowtie S \quad = \quad S \bowtie R$

$(R \bowtie S) \bowtie T \quad = R \bowtie (S \bowtie T)$

Note:

- Carry attribute names in results, so order is not important
- Can also write as trees, e.g.:



Rules: Natural joins & cross products & union

$R \bowtie S \quad = \quad S \bowtie R$

$(R \bowtie S) \bowtie T \quad = R \bowtie (S \bowtie T)$

R x S = S x R

(R x S) x T = R x (S x T)

R U S = S U R

R U (S U T) = (R U S) U T

Rules: Selects

$\sigma_{p1 \wedge p2}(R) =$

$\sigma_{p1 \vee p2}(R) =$

Rules: Selects

$\sigma_{p1 \wedge p2}(R) = \quad \sigma_{p1} \; [\, \sigma_{p2} (R)]$

$\sigma_{p1 \vee p2}(R) = \quad [\, \sigma_{p1} (R)] \cup [\, \sigma_{p2} (R)]$

Bags vs. Sets

R = {a,a,b,b,b,c}

S = {b,b,c,c,d}

RUS = ?

Bags vs. Sets

R = {a,a,b,b,b,c}
S = {b,b,c,c,d}
RUS = ?

- Option 1 SUM
  RUS = {a,a,b,b,b,b,b,c,c,c,d}
- Option 2 MAX
  RUS = {a,a,b,b,b,c,c,d}

Option 2 (MAX) makes this rule work:

$\sigma_{p1 \vee p2}(R) = \sigma_{p1}(R) \cup \sigma_{p2}(R)$

Example: R={a,a,b,b,b,c}

P1 satisfied by a,b; P2 satisfied by b,c

Option 2 (MAX) makes this rule work:

$\sigma_{p1 \vee p2}(R) = \sigma_{p1}(R) \cup \sigma_{p2}(R)$

Example: R={a,a,b,b,b,c}

P1 satisfied by a,b; P2 satisfied by b,c

$\sigma_{p1 \vee p2}(R)$ = {a,a,b,b,b,c}

$\sigma_{p1}(R)$ = {a,a,b,b,b}

$\sigma_{p2}(R)$ = {b,b,b,c}

$\sigma_{p1}(R) \cup \sigma_{p2}(R)$ = {a,a,b,b,b,c}

"Sum" option makes more sense:

Senators (......) Rep (......)

T1 = $\pi_{yr,state}$ Senators; T2 = $\pi_{yr,state}$ Reps

T1

Yr	State
97	CA
99	CA
98	AZ

T2

Yr	State
99	CA
99	CA
98	CA

Union?

Executive Decision

-> Use "SUM" option for bag unions
-> Some rules cannot be used for bags

Rules: Project

Let: X = set of attributes
Y = set of attributes
XY = X U Y

$\pi_{xy}(R)$ =

Rules: Project

Let: X = set of attributes
Y = set of attributes
XY = X U Y

$\pi_{xy}(R) = \pi_x[\pi_y(R)]$

Rules: Project

Let: X = set of attributes
Y = set of attributes
XY = X U Y

$\pi_{xy}(R) = \pi_x[\pi_y(R)]$

Rules: $\sigma + \bowtie$ combined

Let p = predicate with only R attribs
q = predicate with only S attribs
m = predicate with only R,S attribs

$\sigma_p(R \bowtie S) =$

$\sigma_q(R \bowtie S) =$

Rules: $\sigma + \bowtie$ combined

Let p = predicate with only R attribs
q = predicate with only S attribs
m = predicate with only R,S attribs

$\sigma_p(R \bowtie S) = [\sigma_p(R)] \bowtie S$

$\sigma_q(R \bowtie S) = R \bowtie [\sigma_q(S)]$

Rules: $\sigma + \bowtie$ combined (continued)

Some Rules can be Derived:

$\sigma_{p \wedge q}(R \bowtie S) =$

$\sigma_{p \wedge q \wedge m}(R \bowtie S) =$

$\sigma_{p \vee q}(R \bowtie S) =$

Do one:

$\sigma_{p \wedge q}(R \bowtie S) = [\sigma_p(R)] \bowtie [\sigma_q(S)]$

$\sigma_{p \wedge q \wedge m}(R \bowtie S) =$

$\sigma_m \left[(\sigma_p R) \bowtie (\sigma_q S)\right]$

$\sigma_{p \vee q}(R \bowtie S) =$

$\left[(\sigma_p R) \bowtie S\right] \cup \left[R \bowtie (\sigma_q S)\right]$

--> Derivation for first one:

$\sigma_{p \wedge q}(R \bowtie S) =$

$\sigma_p [\sigma_q (R \bowtie S)] =$

$\sigma_p [R \bowtie \sigma_q (S)] =$

$[\sigma_p (R)] \bowtie [\sigma_q (S)]$

Rules: π,σ combined

Let x = subset of R attributes
z = attributes in predicate P
(subset of R attributes)

$\pi_x[\sigma_p (R)] =$

Rules: π,σ combined

Let x = subset of R attributes
z = attributes in predicate P
(subset of R attributes)

$\pi_x[\sigma_p (R)] = \{\sigma_p [\pi_x (R)]\}$

Rules: π,σ combined

Let x = subset of R attributes
z = attributes in predicate P
(subset of R attributes)

$\pi_x[\sigma_p (R)] = \pi_x \{\sigma_p [$ ~~π_x~~ $\pi_{xz} (R)]\}$

Rules: π, $\bowtie$ combined

Let x = subset of R attributes
y = subset of S attributes
z = intersection of R,S attributes

$\pi_{xy} (R \bowtie S) =$

Rules: π, $\bowtie$ combined

Let x = subset of R attributes
y = subset of S attributes
z = intersection of R,S attributes

$\pi_{xy} (R \bowtie S) =$

$\pi_{xy}\{[\pi_{xz} (R)] \bowtie [\pi_{yz} (S)]\}$

$\pi_{xy} \{\sigma_p \ (R \bowtie S)\} \ =$

$\pi_{xy} \{\sigma_p \ (R \bowtie S)\} \ =$

$\pi_{xy} \{\sigma_p \ [\pi_{xz'} \ (R) \bowtie \pi_{yz'} \ (S)]\}$

$z' \ = z \cup \{\text{attributes used in P}\}$

<u>Rules</u> for σ, π combined with X

similar...

e.g., $\sigma_p \ (R \ X \ S) \ = \ ?$

<u>Rules</u> $\sigma, \cup$ combined:

$\sigma_p(R \cup S) \ = \ \sigma_p(R) \cup \sigma_p(S)$

$\sigma_p(R - S) \ = \ \sigma_p(R) - S \ = \ \sigma_p(R) - \sigma_p(S)$

Which are "good" transformations?

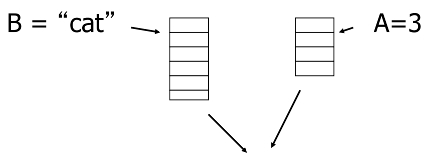
- □ $\sigma_{p1 \wedge p2} \ (R) \rightarrow \sigma_{p1} \ [\sigma_{p2} \ (R)]$
- □ $\sigma_p \ (R \bowtie S) \rightarrow [\sigma_p \ (R)] \bowtie \ S$
- □ $R \bowtie S \ \rightarrow \ S \bowtie R$
- □ $\pi_x \ [\sigma_p \ (R)] \rightarrow \pi_x \ \{\sigma_p \ [\pi_{xz} \ (R)]\}$

Conventional wisdom:
do projects early

<u>Example</u>: R(A,B,C,D,E) x={E}
P: (A=3) ∧ (B="cat")

$\pi_x \ \{\sigma_p \ (R)\}$ vs. $\pi_E \ \{\sigma_p\{\pi_{ABE}(R)\}\}$

But What if we have A, B indexes?



Intersect pointers to get
pointers to matching tuples
e.g., using bitmaps

Bottom line:

- No transformation is always good
- Usually good: early selections
  - Exception: expensive selection conditions
  - E.g., UDFs

More transformations

- Eliminate common sub-expressions
- Detect constant expressions
- Other operations: duplicate elimination

Pushing Selections

- Idea:
  - Join conditions equate attributes
  - For parts of algebra tree (scope) store which attributes have be the same
    - Called Equivalence classes
- Example: R(a,b), S(c,d)

$$\sigma_{b=3} (R \bowtie_{b=c} S) = \sigma_{b=3} (R) \bowtie_{b=c} \sigma_{c=3} (S)$$

Outer-Joins

- Not commutative
  - $R ⟕ S \neq S ⟕ R$
- p – condition over attributes in A
- A list of attributes from R

$\sigma_p (R ⟕_{A=B} S) \equiv \sigma_p (R) ⟕_{A=B} S$

Not $\sigma_p (R ⟕_{A=B} S) \equiv R ⟕_{A=B} \sigma_p (S)$

Summary Equivalences

- Associativity: $(R \odot S) \odot T \equiv R \odot (S \odot T)$
- Commutativity: $R \odot S \equiv S \odot R$
- Distributivity: $(R \odot S) \otimes T \equiv (R \otimes T) \odot (S \otimes T)$
- Difference between Set and Bag Equivalences
- Only some equivalence are useful

Outline - Query Processing

- Relational algebra level
  - transformations
  - good transformations
- Detailed query plan level
  - estimate costs
  - generate and compare plans

- Estimating cost of query plan

(1) Estimating size of results
(2) Estimating # of IOs

Estimating result size

- Keep statistics for relation R
  - T(R) : # tuples in R
  - S(R) : # of bytes in each R tuple
  - B(R): # of blocks to hold all R tuples
  - V(R, A) : # distinct values in R for attribute A

Example

R

A	B	C	D
cat	1	10	a
cat	1	20	b
dog	1	30	a
dog	1	40	c
bat	1	50	d

A: 20 byte string
B: 4 byte integer
C: 8 byte date
D: 5 byte string

Example

R

A	B	C	D
cat	1	10	a
cat	1	20	b
dog	1	30	a
dog	1	40	c
bat	1	50	d

A: 20 byte string
B: 4 byte integer
C: 8 byte date
D: 5 byte string

T(R) = 5 S(R) = 37
V(R,A) = 3 V(R,C) = 5
V(R,B) = 1 V(R,D) = 4

Size estimates for W = R1 x R2

T(W) =

S(W) =

Size estimates for W = R1 x R2

T(W) = T(R1) × T(R2)

S(W) = S(R1) + S(R2)

Size estimate for W = $\sigma_{A=a}$ (R)

S(W) = S(R)

T(W) = ?

Example

R

A	B	C	D
cat	1	10	a
cat	1	20	b
dog	1	30	a
dog	1	40	c
bat	1	50	d

V(R,A)=3
V(R,B)=1
V(R,C)=5
V(R,D)=4

W = $\sigma_{z=val}$(R) T(W) =

Example

R

A	B	C	D
cat	1	10	a
cat	1	20	b
dog	1	30	a
dog	1	40	c
bat	1	50	d

V(R,A)=3
V(R,B)=1
V(R,C)=5
V(R,D)=4

W = $\sigma_{z=val}$(R) $T(W) = \frac{T(R)}{V(R,Z)}$

Assumption:

Values in select expression Z = val
are uniformly distributed
over possible V(R,Z) values.

Alternate Assumption:

Values in select expression Z = val
are uniformly distributed
over domain with DOM(R,Z) values.

Example

R

A	B	C	D
cat	1	10	a
cat	1	20	b
dog	1	30	a
dog	1	40	c
bat	1	50	d

Alternate assumption
V(R,A)=3 DOM(R,A)=10
V(R,B)=1 DOM(R,B)=10
V(R,C)=5 DOM(R,C)=10
V(R,D)=4 DOM(R,D)=10

$W = \sigma_{z=val}(R) \quad T(W) = ?$

C=val ⇒ T(W) = (1/10)1 + (1/10)1 + ...
= (5/10) = 0.5

B=val ⇒ T(W)= (1/10)5 + 0 + 0 = 0.5

A=val ⇒ T(W)= (1/10)2 + (1/10)2 + (1/10)1
= 0.5

Example

R

A	B	C	D
cat	1	10	a
cat	1	20	b
dog	1	30	a
dog	1	40	c
bat	1	50	d

Alternate assumption
V(R,A)=3 DOM(R,A)=10
V(R,B)=1 DOM(R,B)=10
V(R,C)=5 DOM(R,C)=10
V(R,D)=4 DOM(R,D)=10

$W = \sigma_{z=val}(R) \quad T(W) = \frac{T(R)}{DOM(R,Z)}$

Selection cardinality

SC(R,A) = average # records that satisfy equality condition on R.A

$$SC(R,A) = \begin{cases} \frac{T(R)}{V(R,A)} \\ \frac{T(R)}{DOM(R,A)} \end{cases}$$

What about $W = \sigma_{z \geq val}(R)$?

T(W) = ?

What about $W = \sigma_{z \geq val}(R)$?

T(W) = ?

- Solution # 1:
  T(W) = T(R)/2

What about W = $\sigma_{z \geq val}$ (R) ?

T(W) = ?

- Solution # 1:
  T(W) = T(R)/2

- Solution # 2:
  T(W) = T(R)/3

- Solution # 3: Estimate values in range

Example R

	Z

Min=1 ↕ Max=20

V(R,Z)=10

W= $\sigma_{z \geq 15}$ (R)

- Solution # 3: Estimate values in range

Example R

	Z

Min=1 ↕ Max=20

V(R,Z)=10

W= $\sigma_{z \geq 15}$ (R)

$$f = \frac{20-15+1}{20-1+1} = \frac{6}{20}$$ (fraction of range)

T(W) = f $\times$ T(R)

Equivalently:

f$\times$V(R,Z) = fraction of distinct values

$$T(W) = [f \times V(Z,R)] \times \frac{T(R)}{V(Z,R)} = f \times T(R)$$

Size estimate for W = R1 $\bowtie$ R2

Let x = attributes of R1
y = attributes of R2

Size estimate for W = R1 $\bowtie$ R2

Let x = attributes of R1
y = attributes of R2

Case 1 $X \cap Y = \emptyset$

Same as R1 x R2

Case 2 W = R1 ⋈ R2 X ∩ Y = A

R1 | A | B | C R2 | A | D

Case 2 W = R1 ⋈ R2 X ∩ Y = A

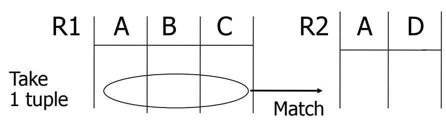
R1 | A | B | C R2 | A | D

Assumption:

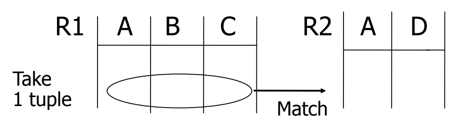
V(R1,A) ≤ V(R2,A) ⇒ Every A value in R1 is in R2

V(R2,A) ≤ V(R1,A) ⇒ Every A value in R2 is in R1

Computing T(W) when V(R1,A) ≤ V(R2,A)



Computing T(W) when V(R1,A) ≤ V(R2,A)



1 tuple matches with $\frac{T(R2)}{V(R2,A)}$ tuples...

so $T(W) = \frac{T(R2)}{V(R2, A)} \times T(R1)$

- V(R1,A) ≤ V(R2,A) $T(W) = \frac{T(R2)\ T(R1)}{V(R2,A)}$
- V(R2,A) ≤ V(R1,A) $T(W) = \frac{T(R2)\ T(R1)}{V(R1,A)}$

[A is common attribute]

In general W = R1 ⋈ R2

$$T(W) = \frac{T(R2)\ T(R1)}{\max\{ V(R1,A), V(R2,A) \}}$$

Case 2 with alternate assumption

Values uniformly distributed over domain

R1 | A | B | C   R2 | A | D

This tuple matches T(R2)/DOM(R2,A) so

$$T(W) = \frac{T(R2)\ T(R1)}{DOM(R2, A)} = \frac{T(R2)\ T(R1)}{DOM(R1, A)}$$

Assume the same

In all cases:

S(W) = S(R1) + S(R2) - S(A) ← size of attribute A

Using similar ideas, we can estimate sizes of:

Π_{AB} (R)

$\sigma_{A=a \wedge B=b}$ (R)

R $\bowtie$ S with common attribs. A,B,C

Union, intersection, diff,

Note: for complex expressions, need intermediate T,S,V results.

E.g. W = [$\sigma_{A=a}$ (R1)] $\bowtie$ R2

Treat as relation U

T(U) = T(R1)/V(R1,A)   S(U) = S(R1)

Also need V (U, *) !!

To estimate Vs

E.g., U = $\sigma_{A=a}$ (R1)

Say R1 has attribs A,B,C,D

V(U, A) =

V(U, B) =

V(U, C) =

V(U, D) =

Example

R1

A	B	C	D
cat	1	10	10
cat	1	20	20
dog	1	30	10
dog	1	40	30
bat	1	50	10

V(R1,A)=3

V(R1,B)=1

V(R1,C)=5

V(R1,D)=3

U = $\sigma_{A=a}$ (R1)

Example

R1

A	B	C	D
cat	1	10	10
cat	1	20	20
dog	1	30	10
dog	1	40	30
bat	1	50	10

V(R1,A)=3
V(R1,B)=1
V(R1,C)=5
V(R1,D)=3

$U = \sigma_{A=a}(R1)$

V(U,A) =1   V(U,B) =1   V(U,C) = $\frac{T(R1)}{V(R1,A)}$

V(D,U) ... somewhere in between

Possible Guess $U = \sigma_{A=a}(R)$

V(U,A) = 1
V(U,B) = V(R,B)

For Joins $U = R1(A,B) \bowtie R2(A,C)$

V(U,A) = min { V(R1, A), V(R2, A) }
V(U,B) = V(R1, B)
V(U,C) = V(R2, C)

Example:

$Z = R1(A,B) \bowtie R2(B,C) \bowtie R3(C,D)$

R1	T(R1) = 1000	V(R1,A)=50	V(R1,B)=100
R2	T(R2) = 2000	V(R2,B)=200	V(R2,C)=300
R3	T(R3) = 3000	V(R3,C)=90	V(R3,D)=500

Partial Result: $U = R1 \bowtie R2$

$T(U) = \frac{1000 \times 2000}{200}$

V(U,A) = 50
V(U,B) = 100
V(U,C) = 300

$Z = U \bowtie R3$

$T(Z) = \frac{1000 \times 2000 \times 3000}{200 \times 300}$

V(Z,A) = 50
V(Z,B) = 100
V(Z,C) = 90
V(Z,D) = 500

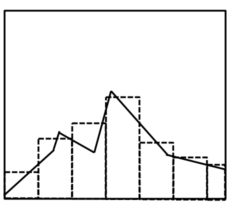
Approximating Distributions

- Summarize the distribution
  - Used to better estimate result sizes
  - Without the need to look at all the data
- Concerns
  - Error metric: How to measure preciseness
  - Memory consumption
  - Computational Complexity

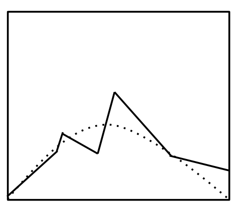
Approximating Distributions

- Parameterized distribution
  - E.g., gauss distribution
  - Adapt parameters to fit data
- Histograms
  - Divide domain into ranges (buckets)
  - Store the number of tuples per bucket
- Both need to be maintained

Histograms



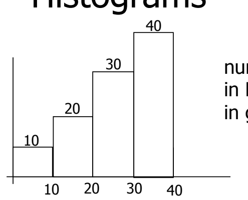
Parameterized Distribution



Maintaining Statistics

- Use separate command that triggers statistics collection
  - Postgres: `ANALYZE`
- During query processing
  - Overhead for queries
- Use Sampling?

Estimating Result Size using Histograms



number of tuples in R with A value in given range

$\sigma_{A=val}(R)$ = ?

Estimating Result Size using Histograms

- $\sigma_{A=val}(R)$ = ?
- |B| - number of values per bucket
- #B – number of records in bucket

$$\frac{\#B}{|B|}$$

Join Size using Histograms

- R ⋈ S
- Use

$$T(W) = \frac{T(R2)\ T(R1)}{\max\{ V(R1,A), V(R2,A) \}}$$

- Apply for each bucket

Join Size using Histograms

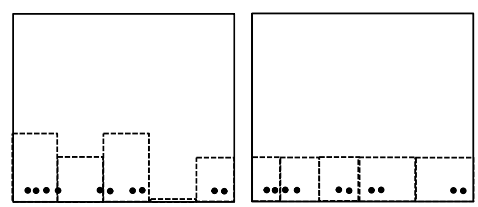
- V(R1,A) = V(R2,A) = bucket size |B|

$$T(W) = \sum_{buckets} \frac{\#B(R2)\ \#B(R1)}{|B|}$$

Equi-width vs. Equi-depth

- Equi-width
  - All buckets contain the same number of values
  - Easy, but inaccurate
- Equi-depth (used by most DBMS)
  - All buckets contain the same number of tuples
  - Better accuracy, need to sort data to compute

Equi-width vs. Equi-depth



Construct Equi-depth Histograms

- Sort input
- Determine size of buckets
  - #bucket / #tuples
- Example 3 buckets

```
1, 5,44, 6,10,12, 3, 6, 7
1, 3, 5, 6, 6, 7,10,12,44
[1-5] [6-8] [9-44]
```


Advanced Techniques

- Wavelets
- Approximate Histograms
- Sampling Techniques
- Compressed Histograms

Summary

- Estimating size of results is an "art"

- Don't forget:
  Statistics must be kept up to date…
  (cost?)

Outline

- Estimating cost of query plan
  - Estimating size of results ⟵ done!
  - Estimating # of IOs ⟵ next…
  - Operator Implementations

- Generate and compare plans