CSE 444: Database Internals

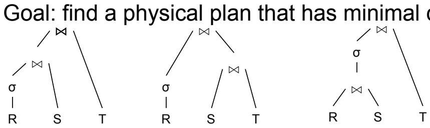
Lecture 9
Query Plan Cost Estimation

Announcements

- Lab 2 / part 1 due tonight 11pm
- Homework 2 due Wednesday 11pm

Query Optimization Summary

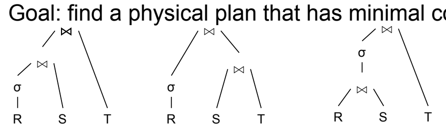
Goal: find a physical plan that has minimal cost



What is the cost of a plan?
For each operator, cost is function of CPU, IO, network bw
Cost of plan is total for all operators
In this class, we look only at IO

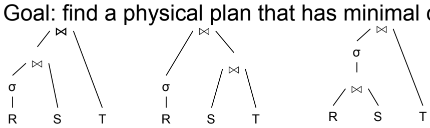
Query Optimization Summary

Goal: find a physical plan that has minimal cost



Query Optimization Summary

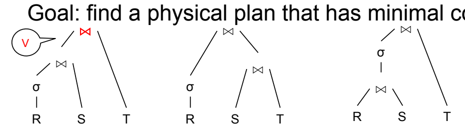
Goal: find a physical plan that has minimal cost



Know how to compute cost if know cardinalities

Query Optimization Summary

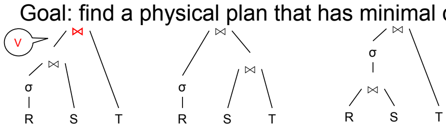
Goal: find a physical plan that has minimal cost



Know how to compute cost if know cardinalities

Query Optimization Summary

Goal: find a physical plan that has minimal cost

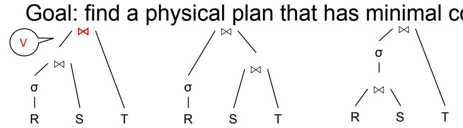


Know how to compute cost if know cardinalities

- Eg. Cost(V ⋈ T) = 3B(V) + 3B(T)
- B(V) = T(V) / PageSize
- T(V) = T(σ(R) ⋈ S)

Query Optimization Summary

Goal: find a physical plan that has minimal cost



Know how to compute cost if know cardinalities

- Eg. Cost(V ⋈ T) = 3B(V) + 3B(T)
- B(V) = T(V) / PageSize
- T(V) = T(σ(R) ⋈ S)

Cardinality estimation problem: e.g. estimate T(σ(R) ⋈ S)

Database Statistics

- Collect statistical summaries of stored data
- Estimate size (=cardinality) in a bottom-up fashion
  - This is the most difficult part, and still inadequate in today's query optimizers
- Estimate cost by using the estimated size
  - Hand-written formulas, similar to those we used for computing the cost of each physical operator

Database Statistics

- Number of tuples (cardinality) T(R)
- Indexes, number of keys in the index V(R,a)
- Number of physical pages B(R)
- Statistical information on attributes
  - Min value, Max value, V(R,a)
- Histograms

- Collection approach: periodic, using sampling

Size Estimation Problem

Q = SELECT list
FROM R1, …, Rn
WHERE $cond_1$ AND $cond_2$ AND . . . AND $cond_k$

Given T(R1), T(R2), …, T(Rn)
Estimate T(Q)

How can we do this ? Note: doesn't have to be exact.

Size Estimation Problem

Q = SELECT list
FROM R1, …, Rn
WHERE $cond_1$ AND $cond_2$ AND . . . AND $cond_k$

Remark: T(Q) ≤ T(R1) × T(R2) × … × T(Rn)

Size Estimation Problem

Q = SELECT list
FROM R1, …, Rn
WHERE $cond_1$ AND $cond_2$ AND . . . AND $cond_k$

Remark: $T(Q) \leq T(R1) \times T(R2) \times \ldots \times T(Rn)$

Key idea: each condition reduces the size of T(Q) by some factor, called selectivity factor

Selectivity Factor

- Each condition cond reduces the size by some factor called selectivity factor
- Assuming independence, multiply the selectivity factors

Example

R(A,B)
S(B,C)
T(C,D)

Q = SELECT *
FROM R, S, T
WHERE R.B=S.B and S.C=T.C and R.A<40

T(R) = 30k, T(S) = 200k, T(T) = 10k

Selectivity of R.B = S.B is 1/3
Selectivity of S.C = T.C is 1/10
Selectivity of R.A < 40 is ½

Q: What is the estimated size of the query output T(Q) ?

Example

R(A,B)
S(B,C)
T(C,D)

Q = SELECT *
FROM R, S, T
WHERE R.B=S.B and S.C=T.C and R.A<40

T(R) = 30k, T(S) = 200k, T(T) = 10k

Selectivity of R.B = S.B is 1/3
Selectivity of S.C = T.C is 1/10
Selectivity of R.A < 40 is ½

Q: What is the estimated size of the query output T(Q) ?

A: T(Q) = 30k * 200k * 10k * 1/3 * 1/10 * ½ = 10^{12}

Selectivity Factors for Conditions

- A = c /* $\sigma_{A=c}(R)$ */
  - Selectivity = 1/V(R,A)

Selectivity Factors for Conditions

- A = c /* $\sigma_{A=c}(R)$ */
  - Selectivity = 1/V(R,A)
- A < c /* $\sigma_{A<c}(R)$*/
  - Selectivity = (c - Low(R, A))/(High(R,A) - Low(R,A))

Selectivity Factors for Conditions

- A = c /* $\sigma_{A=c}(R)$ */
  - Selectivity = 1/V(R,A)
- A < c /* $\sigma_{A<c}(R)$ */
  - Selectivity = (c - Low(R, A))/(High(R,A) - Low(R,A))
- A = B /* $R \bowtie_{A=B} S$ */
  - Selectivity = 1 / max(V(R,A),V(S,A))
  - (will explain next)

Assumptions

- *Containment of values*: if V(R,A) <= V(S,B), then all values R.A occur in S.B
  - Note: this indeed holds when A is a foreign key in R, and B is a key in S
- *Preservation of values*: for any other attribute C, $V(R \bowtie_{A=B} S, C) = V(R, C)$ (or V(S, C))
  - Note: we don't need this to estimate the size of the join, but we need it in estimating the next operator

Selectivity of $R \bowtie_{A=B} S$

Assume V(R,A) <= V(S,B)

- A tuple t in R joins with T(S)/V(S,B) tuple(s) in S
- Hence $T(R \bowtie_{A=B} S) = T(R)\,T(S) / V(S,B)$

$T(R \bowtie_{A=B} S) = T(R)\,T(S) / \max(V(R,A),V(S,B))$

Size Estimation for Join

Example:

- T(R) = 10000, T(S) = 20000
- V(R,A) = 100, V(S,B) = 200
- How large is $R \bowtie_{A=B} S$?

(In class…)

Complete Example

Supplier(sid, sname, scity, sstate)
Supply(sid, pno, quantity)

```
SELECT sname
FROM Supplier x, Supply y
WHERE x.sid = y.sid
   and y.pno = 2
   and x.scity = 'Seattle'
   and x.sstate = 'WA'
```

- Some statistics
  - T(Supplier) = 1000 records
  - T(Supply) = 10,000 records
  - B(Supplier) = 100 pages
  - B(Supply) = 100 pages
  - V(Supplier,scity) = 20, V(Suppliers,state) = 10
  - V(Supply,pno) = 2,500
  - Both relations are clustered
- M = 11

Computing the Cost of a Plan

- Estimate cardinality in a bottom-up fashion
  - Cardinality is the size of a relation (nb of tuples)
  - Compute size of *all* intermediate relations in plan
- Estimate cost by using the estimated cardinalities

T(Supplier) = 1000 B(Supplier) = 100 V(Supplier,scity) = 20 M = 11
T(Supply) = 10,000 B(Supply) = 100 V(Supplier,state) = 10
V(Supply,pno) = 2,500

Physical Query Plan 1

(On the fly) π sname

Selection and project on-the-fly
-> No additional cost.

(On the fly)
σ scity='Seattle' ∧sstate='WA' ∧ pno=2

(Nested loop) sno = sno

Total cost of plan is thus cost of join:
= B(Supplier)+B(Supplier)*B(Supply)
= 100 + 100 * 100
= 10,100 I/Os

Supplier (File scan)

Supply (File scan)

T(Supplier) = 1000 B(Supplier) = 100 V(Supplier,scity) = 20 M = 11
T(Supply) = 10,000 B(Supply) = 100 V(Supplier,state) = 10
V(Supply,pno) = 2,500

Physical Query Plan 2

(On the fly) π sname (d)

(Sort-merge join) sno = sno (c)

(Scan write to T1)
(a) σ scity='Seattle' ∧sstate='WA'

(Scan write to T2)
(b) σ pno=2

Supplier (File scan)

Supply (File scan)

Total cost
= 100 + 100 * 1/20 * 1/10 (a)
+ 100 + 100 * 1/2500 (b)
+ 2 (c)
+ 0 (d)

Total cost ≈ **204 I/Os**

M = 11

Plan 2 with Different Numbers

What if we had:
10K pages of Supplier
10K pages of Supply

π sname (d)

(Sort-merge join) sno = sno (c)

(Scan write to T1)
(a) σ scity='Seattle' ∧sstate='WA'

(Scan write to T2)
(b) σ pno=2

Supplier (File scan)

Supply (File scan)

Total cost
= 10000 + 50 (a)
+ 10000 + 4 (b)
+ 3*50 + 4 (c)
+ 0 (d)

Total cost ≈ 20,208 I/Os

Need to do a two-pass sort algorithm

T(Supplier) = 1000 B(Supplier) = 100 V(Supplier,scity) = 20 M = 11
T(Supply) = 10,000 B(Supply) = 100 V(Supplier,state) = 10
V(Supply,pno) = 2,500

Physical Query Plan 3

(On the fly) (d) π sname

(On the fly)
(c) σ scity='Seattle' ∧sstate='WA'

(b) sno = sno (Index nested loop)

(Use hash index) 4 tuples
(a) σ pno=2

Supply (Hash index on pno) Assume: clustered

Supplier (Hash index on sno) Clustering does not matter

Total cost
= 1 (a)
+ 4 (b)
+ 0 (c)
+ 0 (d)

Total cost ≈ **5 I/Os**

Histograms

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

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50
min(age) = 19, max(age) = 68

$\sigma_{age=48}$(Empolyee) = ? $\sigma_{age>28 \text{ and } age<35}$(Empolyee) = ?

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50
min(age) = 19, max(age) = 68

$\sigma_{age=48}$(Empolyee) = ? $\sigma_{age>28 \text{ and } age<35}$(Empolyee) = ?

Estimate = 25000 / 50 = 500 Estimate = 25000 * 6 / 50 = 3000

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50
min(age) = 19, max(age) = 68

$\sigma_{age=48}$(Empolyee) = ? $\sigma_{age>28 \text{ and } age<35}$(Empolyee) = ?

Age:	0..20	20..29	30-39	40-49	50-59	> 60
Tuples	200	800	5000	12000	6500	500

Histograms

Employee(ssn, name, age)

T(Employee) = 25000, V(Empolyee, age) = 50
min(age) = 19, max(age) = 68

$\sigma_{age=48}$(Empolyee) = ? $\sigma_{age>28 \text{ and } age<35}$(Empolyee) = ?

Age:	0..20	20..29	30-39	40-49	50-59	> 60
Tuples	200	800	5000	12000	6500	500

Estimate = 1200 Estimate = 1*80 + 5*500 = 2580

Types of Histograms

- How should we determine the bucket boundaries in a histogram ?

Types of Histograms

- How should we determine the bucket boundaries in a histogram ?
- Eq-Width
- Eq-Depth
- Compressed
- V-Optimal histograms

Employee(ssn, name, age)

Histograms

Eq-width:

Age:	0..20	20..29	30-39	40-49	50-59	> 60
Tuples	200	800	5000	12000	6500	500

Eq-depth:

Age:	0..33	33..38	38-43	43-45	45-54	> 54
Tuples	1800	2000	2100	2200	1900	1800

Compressed: store separately highly frequent values: (48,1900)

V-Optimal Histograms

- Defines bucket boundaries in an optimal way, to minimize the error over all point queries
- Computed rather expensively, using dynamic programming
- Modern databases systems use V-optimal histograms or some variations

Difficult Questions on Histograms

- Small number of buckets
  - Hundreds, or thousands, but not more
  - WHY ?
- *Not* updated during database update, but recomputed periodically
  - WHY ?
- Multidimensional histograms rarely used
  - WHY ?

Difficult Questions on Histograms

- Small number of buckets
  - Hundreds, or thousands, but not more
  - WHY? All histograms are kept in main memory during query optimization; plus need fast access
- *Not* updated during database update, but recomputed periodically
  - WHY? Histogram update creates a write conflict; would dramatically slow down transaction throughput
- Multidimensional histograms rarely used
  - WHY? Too many possible multidiimensional histograms, unclear which ones to choose