CSE 444: Database Internals

Lecture 12 Query Optimization (part 3)

CSE 444 - Spring 2016

1

3

5

Acknowledgments

Today's lecture focuses on how to actually implement the Selinger optimizer

(Many slides from Sam Madden at MIT)

CSE 444 - Spring 2016

Selinger Optimizer

Goal:

• How to order a series joins over N tables A,B,C,... E.g. A.a = B.b AND A.c = D.d AND B.e = C.f

Problem:

• ... too ... many ... plans ...

CSE 444 - Spring 2016

Reminders

- I'm not Magda
- Lab 2 is due on Friday by 11pm
- Lab 3 released this Friday (transactions, yay!)
- HW 5 due next week
- Quiz on 4/25 (next Monday)

CSE 444 - Spring 2016

2

EXEMPTION Substitution of the product of the prod

Selinger Optimizer

Goal:

• How to order a series joins over N tables A,B,C,... E.g. A.a = B.b AND A.c = D.d AND B.e = C.f

Problem:

• N! ways to order joins; e.g. ABCD, ACBD,

• $C_{N-1} = \frac{1}{N} \binom{2(N-1)}{N-1}$ plans/ordering; e.g. (((AB)C)D), ((AB)(CD)))

- Multiple implementations (hash, nested loops)
- Naïve approach does not scale
 E.g. N = 20, #join orders 20! = 2.4 x 10¹⁸; many more plans

CSE 444 - Spring 2016

6

Selinger Optimizer

- Only left-deep plans: (((AB)C)D) eliminate C_{N-1}.
- Push down selections
- Don't consider Cartesian products
- Dynamic programming algorithm

CSE 444 - Spring 2016

7

9

11

Dynamic Programming

OrderJoins: R = set of relations to join For d = 1 to |R|: For S in {all size-d subsets of R}: Pick a∈S with lowest cost (S-a)⋈a ↑

What is the cost? * Cost to scan a * Cost to produce S-a

* Cost to join (S-a) with a

CSE 444 - Spring 2016

Dynamic Programming

OrderJoins:

 $\begin{array}{l} \mathsf{R} \; = \; set \; of \; relations \; to \; join \\ \begin{array}{l} \mathsf{For} \; d = 1 \; to \; |\mathsf{R}|: \\ \end{array} \\ \begin{array}{l} \mathsf{For} \; S \; in \; \{ all \; size-d \; subsets \; of \; \mathsf{R} \}: \\ \begin{array}{l} \mathsf{optjoin}(\mathsf{S}) \; = \; (\mathsf{S}-a) \; join \; a, \\ \end{array} \\ where \; a \; is \; the \; single \; relation \; that \; minimizes: \\ cost(\mathsf{optjoin}(\mathsf{S}-a)) + \\ min.cost \; to \; join \; (\mathsf{S}-a) \; with \; a \; + \\ min.access \; cost \; for \; a \\ \end{array}$

Note: **optjoin**(S-a) is cached from previous iterations

CSE 444 - Spring 2016

Dynamic Programming

OrderJoins: R = set of relations to join For d = 1 to |R|: For S in {all size-d subsets of R}: Pick a∈S with lowest cost (S-a)Ma

CSE 444 - Spring 2016

8

10

Dynamic Programming

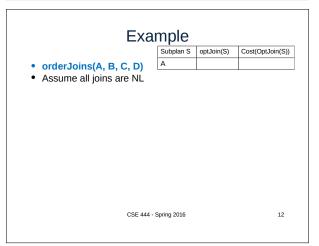
OrderJoins: R = set of relations to joinFor d = 1 to |R|: For S in {all size-d subsets of R}: Pick a \in S with lowest cost (S-a) \bowtie a

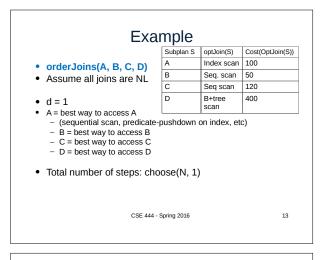
What is the cost?

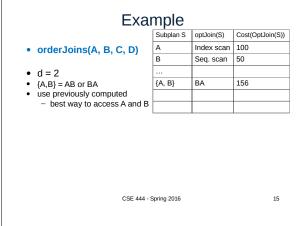
* Cost to scan a

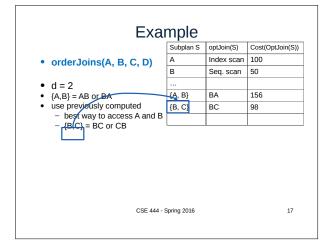
- * Cost to produce S-a ← Calculated in previous iteration
- * Cost to join (S-a) with a

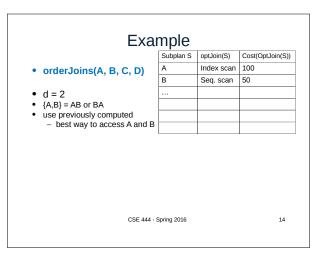
CSE 444 - Spring 2016

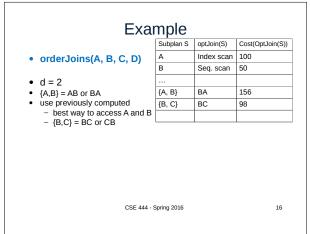


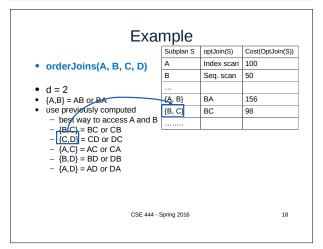


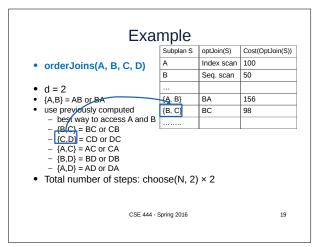


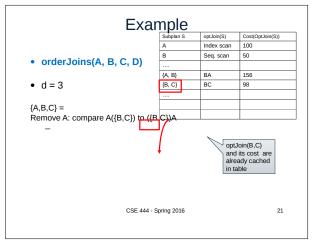


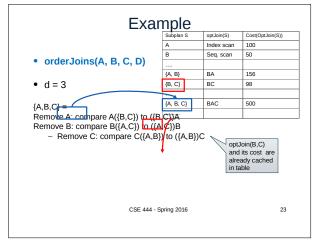


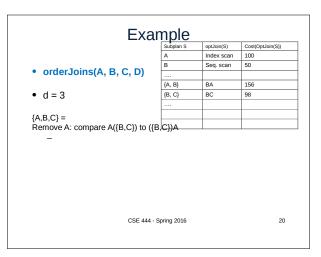


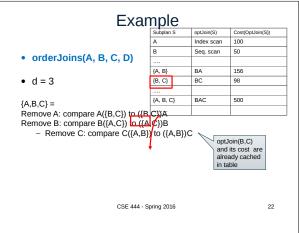


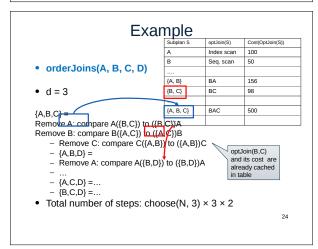


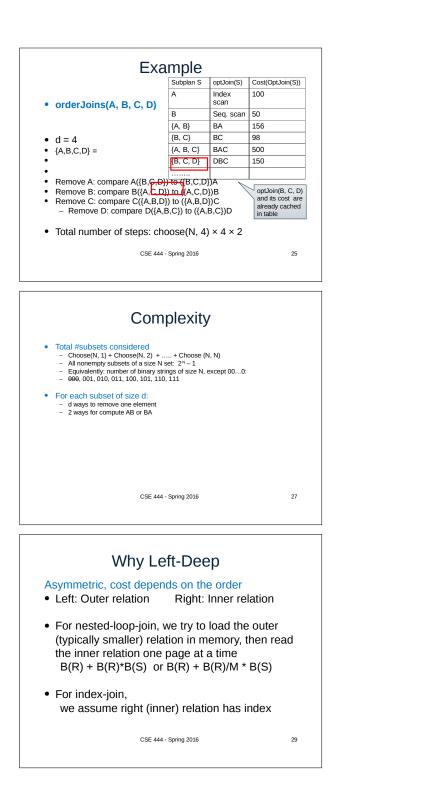




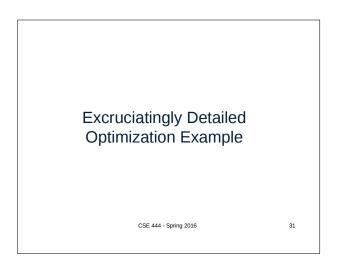








	Complexity	
	Complexity	
• Total #subsets cor	nsidered	
	hoose(N, 2) + + Choose (N, N) sets of a size N set: 2 ^N − 1	
	ber of binary strings of size N, except 000:	
,,,	_,,,,,	
	CSE 444 - Spring 2016	26
	COL 444 - Opining 2010	20
	Complexity	
Total #cubcote com	asidorod	
 Total #subsets con Choose(N, 1) + Cl 	hoose(N, 2) + + Choose (N, N)	
 Equivalently: num 	sets of a size N set: 2 ^N – 1 ber of binary strings of size N, except 000:	
 - 000, 001, 010, 013 For each subset of 		
 d ways to remove 2 ways for compute 		
 Total #plans consi 	dered	
 Equivalently: total 	Choose(N, 2) + + N Choose (N, N) number of 1's in all strings of size N ery 1 occurs 2 ^{N-1} times	
	ultiply by 2, to account for AB or BA	
	CSE 444 - Spring 2016	28
V	Vhy Left-Deep	
	left-deep trees?	
	standard join algorithms (nested loo	p, one-pass),
	: Uses smaller memory	(D Q)
(R, (S, T)):	can reuse the space for R while joining Need to hold R, compute (S, T), then jo pre relations	
3. Nested loop id	pin, consider top-down iterator next()
1. ((R, S), T),	Reads the chunks of (R, S) once, reads nultiple times	
	Reads the chunks of R once, reads cor	
		Ce.
	ple times, either more time or more spa	ce



Interesting Orders

- Some query plans produce data in sorted order E.g scan over a primary index, merge-join
 Called *interesting order*
- Next operator may use this order E.g. can be another merge-join
- For each subset of relations, compute multiple optimal plans, one for each interesting order
- Increases complexity by factor k+1, where k=number of interesting orders

CSE 444 - Spring 2016

32