

CMU SCS

**Carnegie Mellon Univ.
Dept. of Computer Science
15-415 - Database Applications**

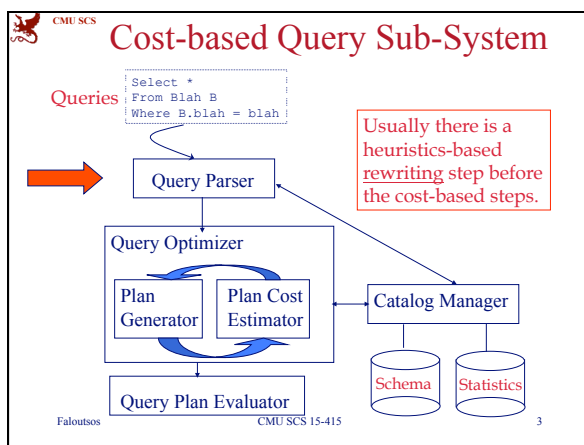
Lecture #15: Query Optimization
(R&G ch. 15; Sys. R q-opt paper)

CMU SCS

Overview - detailed

- Why q-opt?
- Equivalence of expressions
- Cost estimation
- Plan generation
- Plan evaluation

Faloutsos CMU SCS 15-415 2



CMU SCS

Why Q-opt?

- SQL: ~declarative
- good q-opt -> big difference
 - eg., seq. Scan vs
 - B-tree index, on P=1,000 pages

Faloutsos CMU SCS 15-415 4

CMU SCS

Q-opt steps

- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
- estimate cost; pick best

Faloutsos CMU SCS 15-415 5

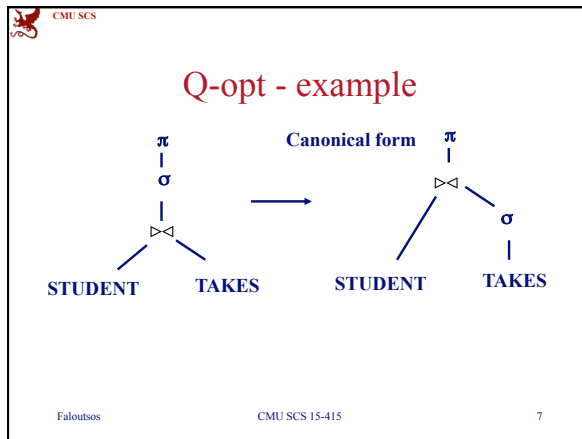
CMU SCS

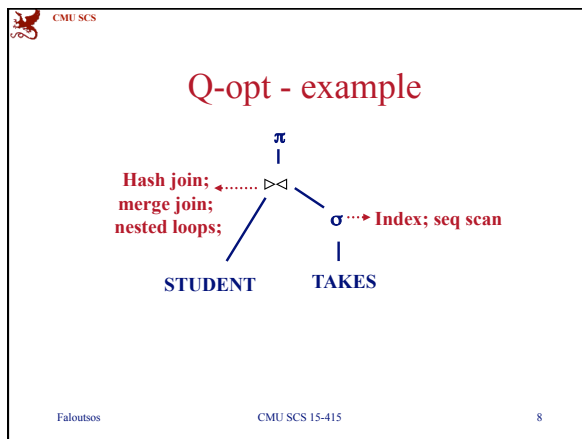
Q-opt - example

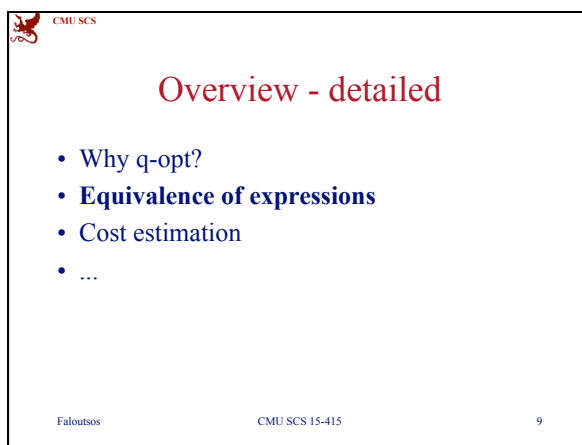
```
select name
from STUDENT, TAKES
where c-id='415' and
STUDENT.ssn=TAKES.ssn
```

$$\begin{array}{c} \pi \\ | \\ \sigma \\ | \\ \triangleleft \\ \swarrow \quad \searrow \\ \text{STUDENT} \quad \text{TAKES} \end{array}$$

Faloutsos CMU SCS 15-415 6









Equivalence of expressions

- A.k.a.: syntactic q-opt
- in short: perform selections and projections early
- More details: see transf. rules in text

Faloutsos CMU SCS 15-415 10



Equivalence of expressions

- Q: How to prove a transf. rule?

$$\sigma_p(R1 \bowtie R2) \stackrel{?}{=} \sigma_p(R1) \bowtie \sigma_p(R2)$$
- A: use RTC, to show that LHS = RHS, eg:

$$\sigma_p(R1 \cup R2) \stackrel{?}{=} \sigma_p(R1) \cup \sigma_p(R2)$$

Faloutsos CMU SCS 15-415 11



Equivalence of expressions

$$\sigma_p(R1 \cup R2) \stackrel{?}{=} \sigma_p(R1) \cup \sigma_p(R2)$$

$$t \in LHS \Leftrightarrow$$

$$t \in (R1 \cup R2) \wedge P(t) \Leftrightarrow$$

$$(t \in R1 \vee t \in R2) \wedge P(t) \Leftrightarrow$$

$$(t \in R1 \wedge P(t)) \vee (t \in R2 \wedge P(t)) \Leftrightarrow$$

Faloutsos CMU SCS 15-415 12

CMU SCS

Equivalence of expressions

$$\sigma_p(R1 \cup R2) \stackrel{?}{=} \sigma_p(R1) \cup \sigma_p(R2)$$

...

$$(t \in R1 \wedge P(t)) \vee (t \in R2 \wedge P(t)) \Leftrightarrow$$

$$(t \in \sigma_p(R1)) \vee (t \in \sigma_p(R2)) \Leftrightarrow$$

$$t \in \sigma_p(R1) \cup \sigma_p(R2) \Leftrightarrow$$

$$t \in RHS$$

QED

Faloutsos CMU SCS 15-415 13

CMU SCS

Equivalence of expressions

- Q: how to disprove a rule??

$$\pi_A(R1 - R2) \stackrel{?}{=} \pi_A(R1) - \pi_A(R2)$$

Faloutsos CMU SCS 15-415 14

CMU SCS

Equivalence of expressions

- Selections
 - perform them early
 - break a complex predicate, and push

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

- simplify a complex predicate
 - ('X=Y and Y=3') $\rightarrow$ 'X=3 and Y=3'

Faloutsos CMU SCS 15-415 15



Equivalence of expressions

- Projections
 - perform them early (but carefully...)
 - Smaller tuples
 - Fewer tuples (if duplicates are eliminated)
 - project out all attributes except the ones requested or required (e.g., joining attr.)

Faloutsos CMU SCS 15-415 16



Equivalence of expressions

- Joins
 - Commutative, associative

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

$$(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$$
 - Q: n-way join - how many diff. orderings?

Faloutsos CMU SCS 15-415 17



Equivalence of expressions

- Joins - Q: n-way join - how many diff. orderings?
- A: Catalan number $\sim 4^n$
 - Exhaustive enumeration: too slow.

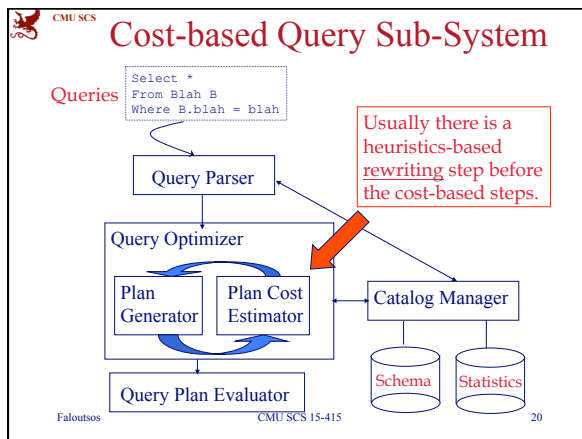
Faloutsos CMU SCS 15-415 18

CMU SCS

Q-opt steps

- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
- **estimate cost**; pick best

Faloutsos CMU SCS 15-415 19



CMU SCS

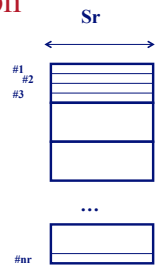
Cost estimation

- Eg., find ssn's of students with an 'A' in 415 (using seq. scanning)
- How long will a query take?
 - CPU (but: small cost; decreasing; tough to estimate)
 - Disk (mainly, # block transfers)
- How many tuples will qualify?
- (what statistics do we need to keep?)

Faloutsos CMU SCS 15-415 21

Cost estimation

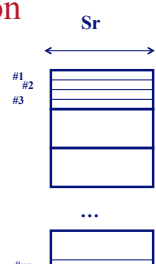
- Statistics: for each relation 'r' we keep
 - nr : # tuples;
 - Sr : size of tuple in bytes



Faloutsos CMU SCS 15-415 22

Cost estimation

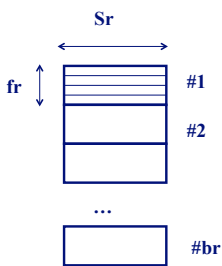
- Statistics: for each relation 'r' we keep
 - ...
 - $V(A,r)$: number of distinct values of attr. 'A'
 - (recently, histograms, too)



Faloutsos CMU SCS 15-415 23

Derivable statistics

- blocking factor = max# records/block (=??)
- br: # blocks (=??)
- $SC(A,r)$ = selection cardinality = avg# of records with A=given (=??)



Faloutsos CMU SCS 15-415 24

CMU SCS

Derivable statistics

- blocking factor = max# records/block ($= B / S_r$; B: block size in bytes)
- br: # blocks ($= nr / (\text{blocking-factor})$)

Faloutsos CMU SCS 15-415 25

CMU SCS

Derivable statistics

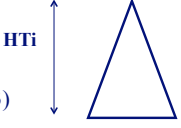
- $SC(A,r)$ = selection cardinality = avg# of records with A=given ($= nr / V(A,r)$) (assumes uniformity...) – eg: 10,000 students, 10 colleges – how many students in SCS?

Faloutsos CMU SCS 15-415 26

CMU SCS

Additional quantities we need:

- For index 'i':
 - f_i : average fanout ($\sim 50-100$)
 - HT_i : # levels of index 'i' ($\sim 2-3$)
 - $\sim \log(\#entries) / \log(f_i)$
 - LB_i : # blocks at leaf level



Faloutsos CMU SCS 15-415 27



CMU SCS

Statistics

- Where do we store them?
- How often do we update them?

Faloutsos CMU SCS 15-415 28



CMU SCS

Q-opt steps

- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
- **estimate cost**; pick best

Faloutsos CMU SCS 15-415 29



CMU SCS

Selections

- we saw simple predicates (A=constant; eg., 'name=Smith')
- how about more complex predicates, like
 - 'salary > 10K'
 - 'age = 30 **and** job-code="analyst"'
- what is their selectivity?

Faloutsos CMU SCS 15-415 30

CMU SCS

Selections – complex predicates

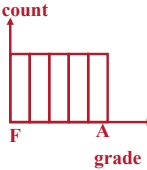
- selectivity $\text{sel}(P)$ of predicate P :
 $\Rightarrow$ fraction of tuples that qualify
 $\text{sel}(P) = \text{SC}(P) / \text{nr}$

Faloutsos CMU SCS 15-415 31

CMU SCS

Selections – complex predicates

- eg., assume that $V(\text{grade}, \text{TAKES})=5$ distinct values
- simple predicate $P: A=\text{constant}$
 - $\text{sel}(A=\text{constant}) = 1/V(A,r)$
 - eg., $\text{sel}(\text{grade}='B') = 1/5$
- (what if $V(A,r)$ is unknown??)

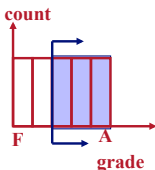


Faloutsos CMU SCS 15-415 32

CMU SCS

Selections – complex predicates

- range query: $\text{sel}(\text{grade} \geq 'C')$
 - $\text{sel}(A \geq a) = (A_{\text{max}} - a) / (A_{\text{max}} - A_{\text{min}})$

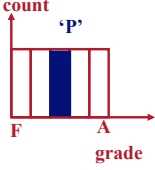


Faloutsos CMU SCS 15-415 33

CMU SCS

Selections - complex predicates

- negation: $\text{sel}(\text{grade} \neq \text{'C'})$
 - $\text{sel}(\text{not } P) = 1 - \text{sel}(P)$
 - (Observation: selectivity $\approx$ probability)



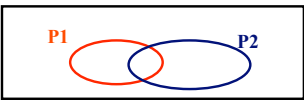
Faloutsos CMU SCS 15-415 34

CMU SCS

Selections – complex predicates

conjunction:

- $\text{sel}(\text{grade} = \text{'C'} \text{ and } \text{course} = \text{'415'})$
- $\text{sel}(P1 \text{ and } P2) = \text{sel}(P1) * \text{sel}(P2)$
- INDEPENDENCE ASSUMPTION



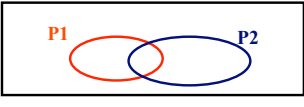
Faloutsos CMU SCS 15-415 35

CMU SCS

Selections – complex predicates

disjunction:

- $\text{sel}(\text{grade} = \text{'C'} \text{ or } \text{course} = \text{'415'})$
- $\text{sel}(P1 \text{ or } P2) = \text{sel}(P1) + \text{sel}(P2) - \text{sel}(P1 \text{ and } P2)$
- $= \text{sel}(P1) + \text{sel}(P2) - \text{sel}(P1) * \text{sel}(P2)$
- INDEPENDENCE ASSUMPTION, again

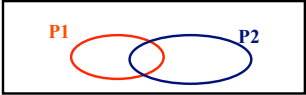


Faloutsos CMU SCS 15-415 36

CMU SCS

Selections – complex predicates

disjunction: in general
 $\text{sel}(P1 \text{ or } P2 \text{ or } \dots Pn) =$
 $1 - (1 - \text{sel}(P1)) * (1 - \text{sel}(P2)) * \dots (1 - \text{sel}(Pn))$



Faloutsos CMU SCS 15-415 37

CMU SCS

Selections – summary

- $\text{sel}(A=\text{constant}) = 1/V(A,r)$
- $\text{sel}(A > a) = (A_{\max} - a) / (A_{\max} - A_{\min})$
- $\text{sel}(\text{not } P) = 1 - \text{sel}(P)$
- $\text{sel}(P1 \text{ and } P2) = \text{sel}(P1) * \text{sel}(P2)$
- $\text{sel}(P1 \text{ or } P2) = \text{sel}(P1) + \text{sel}(P2) - \text{sel}(P1) * \text{sel}(P2)$
- $\text{sel}(P1 \text{ or } \dots \text{ or } Pn) = 1 - (1 - \text{sel}(P1)) * \dots * (1 - \text{sel}(Pn))$
- UNIFORMITY and INDEPENDENCE ASSUMPTIONS

Faloutsos CMU SCS 15-415 38

CMU SCS

Result Size Estimation for Joins

- Q: Given a join of R and S, what is the range of possible result sizes (in #of tuples)?
 - Hint: what if $R_cols \cap S_cols = \emptyset$?
 - $R_cols \cap S_cols$ is a key for R (and a Foreign Key in S)?

Faloutsos CMU SCS 15-415 39

CMU SCS

Result Size Estimation for Joins

- General case: $R_cols \cap S_cols = \{A\}$ (and A is key for neither)
 - match each R-tuple with S-tuples

$$est_size \leq NTuples(R) * NTuples(S) / NKeys(A, S)$$

$$\leq nr * ns / V(A, S)$$
 - symmetrically, for S:

$$est_size \leq NTuples(R) * NTuples(S) / NKeys(A, R)$$

$$\leq nr * ns / V(A, R)$$
 - Overall:

$$est_size = NTuples(R) * NTuples(S) / \max\{NKeys(A, S), NKeys(A, R)\}$$

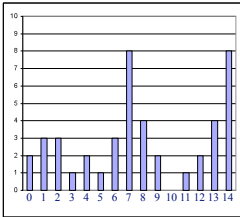
Faloutsos CMU SCS 15-415 40

CMU SCS

On the Uniform Distribution Assumption

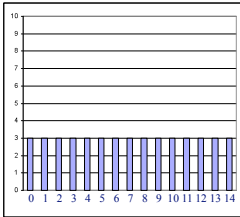
- Assuming uniform distribution is rather crude

Distribution D



Faloutsos

Uniform distribution approximating D



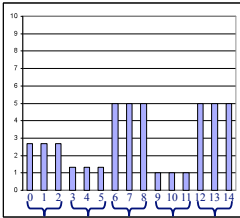
CMU SCS 15-415 41

CMU SCS

Histograms

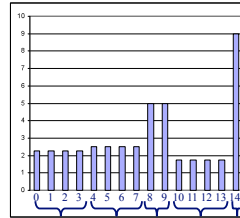
- For better estimation, use a *histogram*

Equiwidth histogram



Faloutsos

Equidepth histogram ~ quantiles



CMU SCS 15-415 42

CMU SCS

Q-opt steps

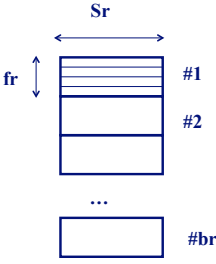
- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- **generate alt. plans**
 - **single relation**
 - multiple relations
- estimate cost; pick best

Faloutsos CMU SCS 15-415 43

CMU SCS

plan generation

- Selections – eg.,
select *
from TAKES
where grade = 'A'
- Plans?

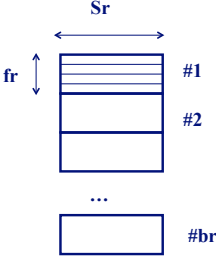


Faloutsos CMU SCS 15-415 44

CMU SCS

plan generation

- Plans?
 - seq. scan
 - binary search
 - (if sorted & consecutive)
 - index search
 - if an index exists



Faloutsos CMU SCS 15-415 45

plan generation

seq. scan – cost?

- br (worst case)
- br/2 (average, if we search for primary key)

Faloutsos CMU SCS 15-415 46

plan generation

binary search – cost?

if sorted and consecutive:

- $\sim \log(\text{br}) +$
- $\text{SC}(\text{A}, \text{r}) / \text{fr}$ (=blocks spanned by qual. tuples)

Faloutsos CMU SCS 15-415 47

plan generation

estimation of selection cardinalities $\text{SC}(\text{A}, \text{r})$:

non-trivial – we saw it earlier

Faloutsos CMU SCS 15-415 48

plan generation

method#3: index – cost?

- levels of index +
- blocks w/ qual. tuples

case#1: primary key

case#2: sec. key – clustering index

case#3: sec. key – non-clust. index

Faloutsos CMU SCS 15-415 49

plan generation

method#3: index – cost?

- levels of index +
- blocks w/ qual. tuples

case#1: primary key – cost:

$HT_i + 1$

Faloutsos CMU SCS 15-415 50

plan generation

method#3: index – cost?

- levels of index +
- blocks w/ qual. tuples

case#2: sec. key – clustering index

$HT_i + SC(A,r)/fr$

Faloutsos CMU SCS 15-415 51

CMU SCS

plan generation

method#3: index – cost?

- levels of index +
- blocks w/ qual. tuples

case#3: sec. key – non-clust. index

HTi + SC(A,r)

(actually, pessimistic...)

Faloutsos CMU SCS 15-415 52

CMU SCS

plan generation

method#3: index – cost?

- levels of index +
- blocks w/ qual. tuples

(actually, pessimistic...)

better estimates:
Cardenas' formula

Faloutsos CMU SCS 15-415 53

CMU SCS

Cardena's formula

- q: # qual records
- Q: # qual. blocks
- N: # records total
- B: # blocks total
- Q=??

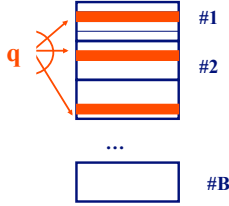
Alfonso Cardenas
(UCLA)

Faloutsos CMU SCS 15-415 54

CMU SCS

Cardena's formula

- Pessimistic:
 - $Q = q$

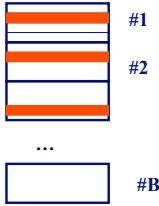


Faloutsos CMU SCS 15-415 55

CMU SCS

Cardena's formula

- Pessimistic:
 - $Q = q$
- More realistic
 - $Q = q$ if $q \leq B$
 - $Q = B$ otherwise

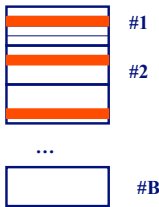


Faloutsos CMU SCS 15-415 56

CMU SCS

Cardena's formula

- Cardenas' formula

$$Q = B [1 - (1 - 1/B)^q]$$


Faloutsos CMU SCS 15-415 57

CMU SCS

Plans for single relation - summary

- no index: scan (dup-elim; sort)
- with index:
 - single index access path
 - multiple index access path
 - sorted index access path
 - index-only access path

Faloutsos CMU SCS 15-415 58

CMU SCS

Citation

- P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price. *Access path selection in a relational database management system*. In SIGMOD Conference, pages 23--34, 1979.

Faloutsos CMU SCS 15-415 59

CMU SCS

Frequently cited database publications

<http://www.informatik.uni-trier.de/~ley/db/about/top.html>

#	Publication
608	Peter P. Chen: The Entity-Relationship Model - Toward a Unified View of Data. ACM Trans. Database Syst. 1(1): 9-36 (1976)
580	E. F. Codd: A Relational Model of Data for Large Shared Data Banks. Commun. ACM 13(6): 377-387(1970)
371	Patricia G. Selinger, Morton M. Astrahan, Donald D. Chamberlin, Raymond A. Lorie, Thomas G. Price: Access Path Selection in a Relational Database Management System. SIGMOD Conference 1979: 23-34
366	Jeffrey D. Ullman: Principles of Database and Knowledge Base Systems, Volume I. Computer Science Press 1988, ISBN 0-7167-8158-1
...	...

Faloutsos CMU SCS 15-415 60

CMU SCS

Statistics for Optimization

- NCARD (T) - cardinality of relation T in tuples
- TCARD (T) - number of pages containing tuples from T
- $P(T) = \text{TCARD}(T) / (\# \text{ of non-empty pages in the segment})$
 - If segments only held tuples from one relation there would be no need for $P(T)$
- ICARD(I) - number of distinct keys in index I
- NINDX(I) - number of pages in index I

Faloutsos CMU SCS 15-415 61

CMU SCS

Predicate Selectivity Estimation

attr = value	$F = 1/\text{ICARD}(\text{attr index})$ – if index exists $F = 1/10$ otherwise
attr1 = attr2	$F = 1/\max(\text{ICARD}(I1), \text{ICARD}(I2))$ or $F = 1/\text{ICARD}(Ii)$ – if only index i exists, or $F = 1/10$
val1 < attr < val2	$F = (\text{value2} - \text{value1}) / (\text{high key} - \text{low key})$ $F = 1/4$ otherwise
expr1 or expr2	$F = F(\text{expr1}) + F(\text{expr2}) - F(\text{expr1}) * F(\text{expr2})$
expr1 and expr2	$F = F(\text{expr1}) * F(\text{expr2})$
NOT expr	$F = 1 - F(\text{expr})$

Faloutsos CMU SCS 15-415 62

CMU SCS

Costs per Access Path Case

Unique index matching equal predicate	$I + 1 + W$
Clustered index I matching ≥ 1 preds	$F(\text{preds}) * (\text{NINDX}(I) + \text{TCARD}) + W * \text{RSICARD}$
Non-clustered index I matching ≥ 1 preds	$F(\text{preds}) * (\text{NINDX}(I) + \text{NCARD}) + W * \text{RSICARD}$
Segment scan	$\text{TCARD}/P + W * \text{RSICARD}$

Faloutsos CMU SCS 15-415 63

CMU SCS

Q-opt steps

- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
 - single relation
 - **multiple relations**
 - Main idea
 - Dynamic programming – reminder
 - Example
- estimate cost; pick best

Faloutsos CMU SCS 15-415 64

CMU SCS

n-way joins

- $r_1 \text{ JOIN } r_2 \text{ JOIN } \dots \text{ JOIN } r_n$
- typically, break problem into 2-way joins
 - choose between NL, sort merge, hash join, ...

Faloutsos CMU SCS 15-415 65

CMU SCS

Queries Over Multiple Relations

- As number of joins increases, number of alternative plans grows rapidly → *need to restrict search space*
- Fundamental decision in System R: only left-deep join trees are considered. Advantages?

66

Queries Over Multiple Relations

- As number of joins increases, number of alternative plans grows rapidly → *need to restrict search space*
- Fundamental decision in System R: *only left-deep join trees* are considered. Advantages?
 - *fully pipelined* plans.
 - Intermediate results not written to temporary files.
 - Not all left-deep trees are fully pipelined (e.g., SM join).

67

Queries over Multiple Relations

- Enumerate the orderings (= left deep tree)
- enumerate the plans for each operator
- enumerate the access paths for each table

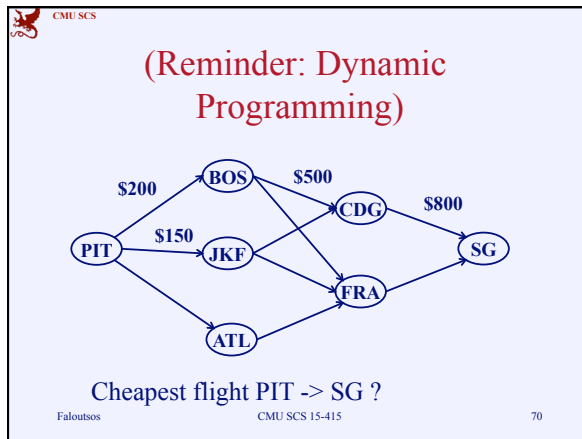
Dynamic programming, to save cost estimations

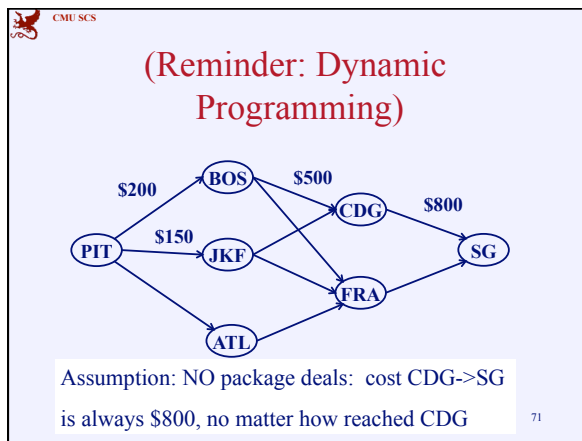
Faloutsos CMU SCS 15-415 68

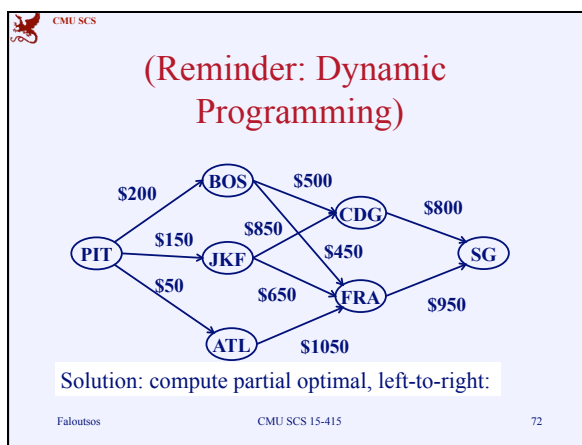
Q-opt steps

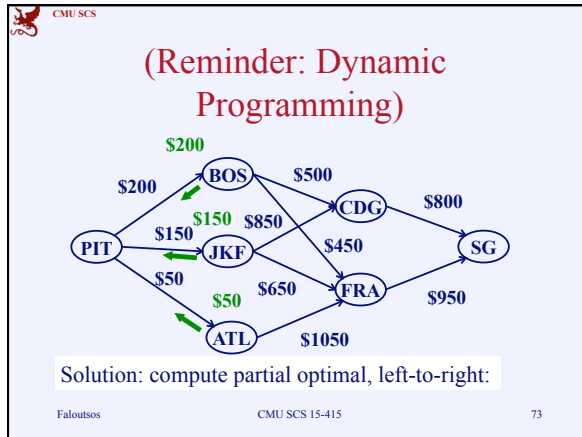
- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
 - single relation
 - multiple relations
 - Main idea
 - **Dynamic programming** – reminder
 - Example
- estimate cost; pick best

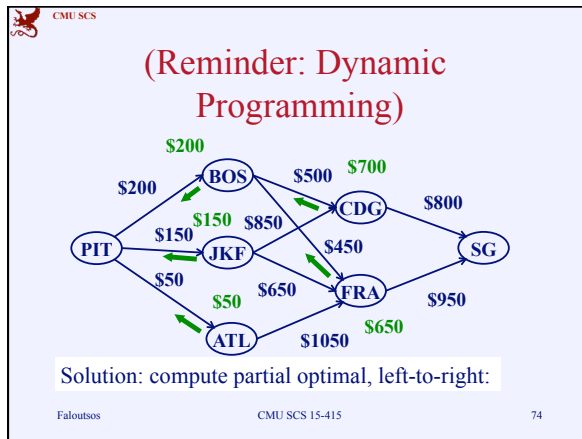
Faloutsos CMU SCS 15-415 69

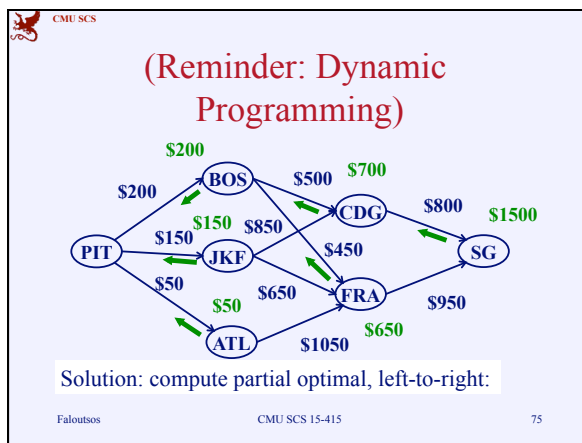


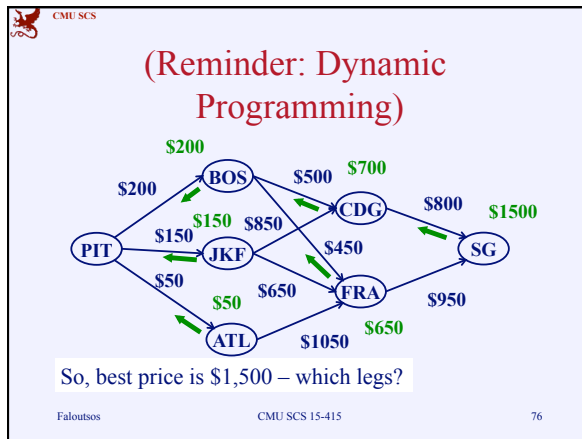


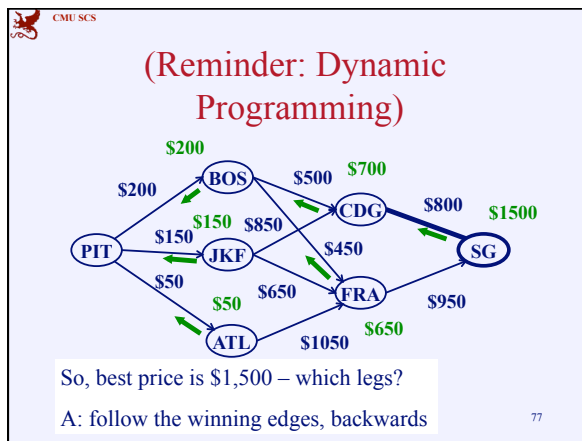


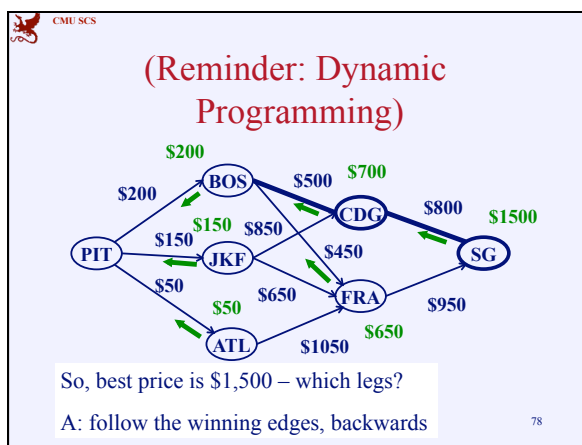


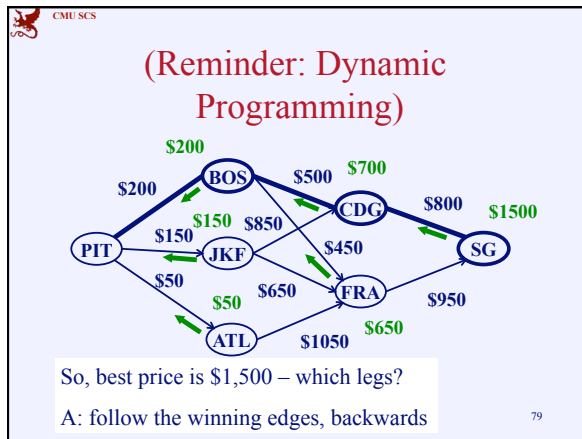


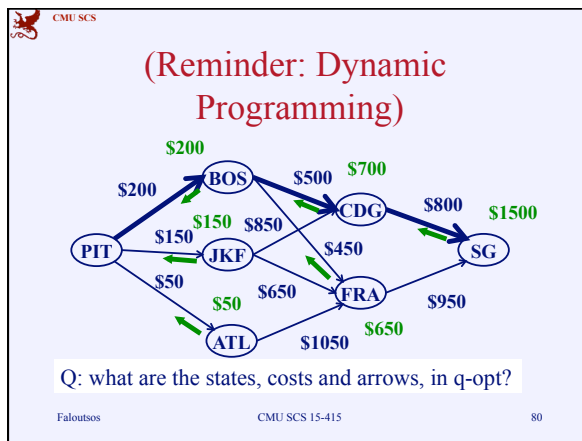


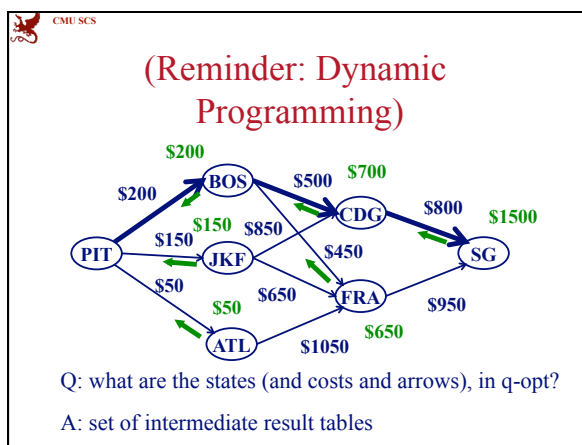


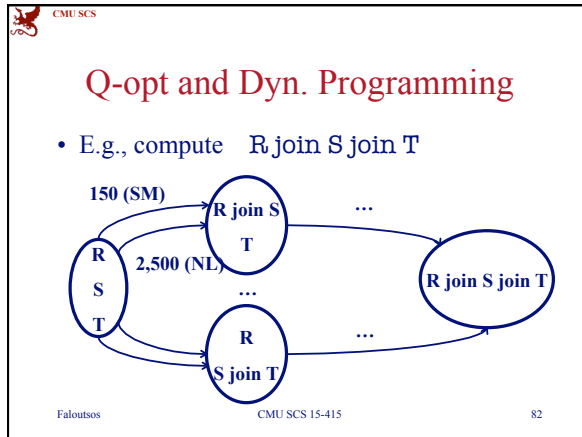


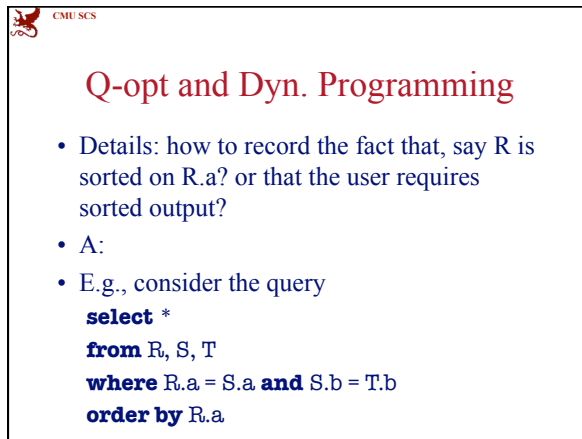


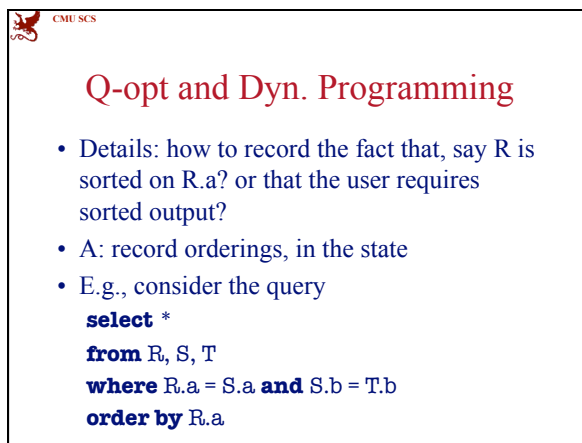


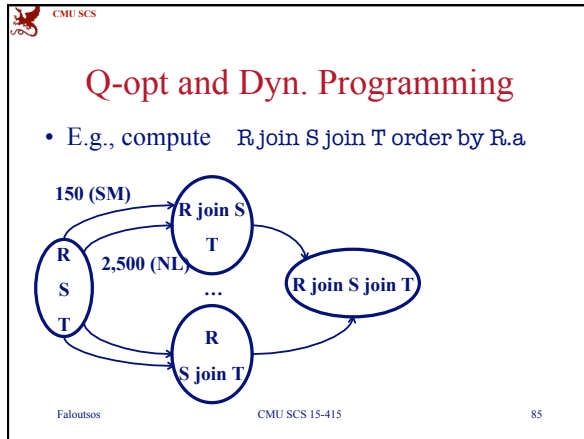


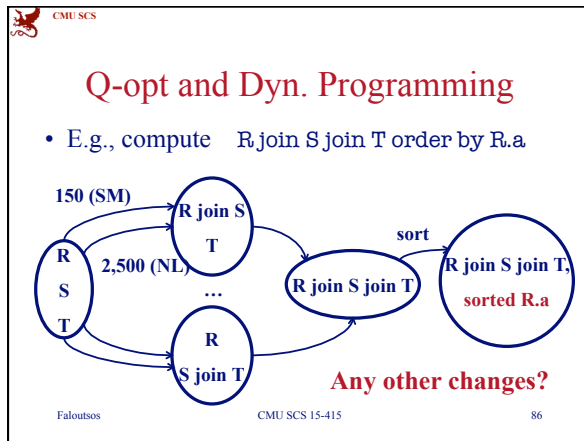


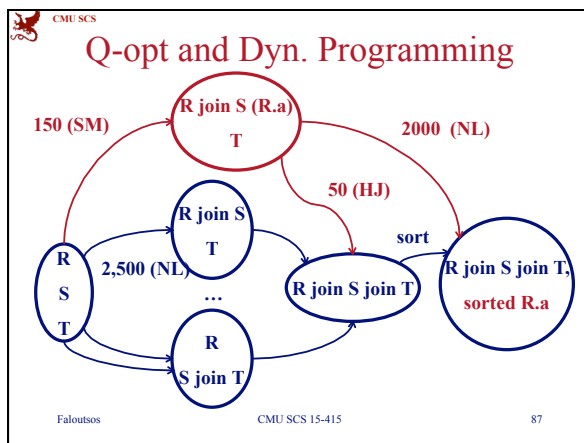












Q-opt steps

- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
 - single relation
 - multiple relations
 - Main idea
 - Dynamic programming – reminder
 - **Example**
- estimate cost; pick best

Faloutsos CMU SCS 15-415 88

Candidate Plans

```
SELECT S.sname, B.bname, R.day
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid
```

1. Enumerate relation orderings:

Prune plans with cross-products immediately!

Faloutsos CMU SCS 15-415 89

Candidate Plans

```
SELECT S.sname, B.bname, R.day
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid
```

2. Enumerate join algorithm choices:

+ do same for 4 other plans
→ 4*4 = 16 plans so far..

Faloutsos CMU SCS 15-415 90

Candidate Plans

```
SELECT S.sname, B.bname, R.day
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid
```

3. Enumerate access method choices:

+ do same for other plans

Faloutsos CMU SCS 15-415

Now estimate the cost of each plan

Example:

Faloutsos CMU SCS 15-415 92

Q-opt steps

- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
 - single relation
 - multiple relations
 - **nested subqueries**
- estimate cost; pick best

Faloutsos CMU SCS 15-415 93

CMU SCS

Q-opt steps

- Everything so far: about a single query block

Faloutsos CMU SCS 15-415 94

CMU SCS

Query Rewriting

- Re-write nested queries
- to: **de-correlate** and/or **flatten** them

Faloutsos CMU SCS 15-415 95

CMU SCS

Example: Decorrelating a Query

```
SELECT S.sid
FROM Sailors S
WHERE EXISTS
  (SELECT *
   FROM Reserves R
   WHERE R.bid=103
    AND R.sid=S.sid)
```

Equivalent uncorrelated query:

```
SELECT S.sid
FROM Sailors S
WHERE S.sid IN
  (SELECT R.sid
   FROM Reserves R
   WHERE R.bid=103)
```

- Advantage:** nested block only needs to be executed **once** (rather than once per S tuple)

Faloutsos CMU SCS 15-415 96

CMU SCS

Example: “Flattening” a Query

```
SELECT S.sid
FROM Sailors S
WHERE S.sid IN
  (SELECT R.sid
   FROM Reserves R
   WHERE R.bid=103)
```

Equivalent non-nested query:

```
SELECT S.sid
FROM Sailors S, Reserves R
WHERE S.sid=R.sid
AND R.bid=103
```

- **Advantage:** can use a join algorithm + optimizer can select among join algorithms & reorder freely

Faloutsos CMU SCS 15-415 97

CMU SCS

Structure of query optimizers:

System R:

- break query in query blocks
- simple queries (ie., no joins): look at stats
- n-way joins: left-deep join trees; ie., only one intermediate result at a time
 - pros: smaller search space; pipelining
 - cons: may miss optimal
- 2-way joins: NL and sort-merge

Faloutsos CMU SCS 15-415 98

CMU SCS

Structure of query optimizers:

More heuristics by Oracle, Sybase and Starburst (-> DB2)

In general: q-opt is very important for large databases.

(‘**explain select** <sql-statement>’ gives plan)

Faloutsos CMU SCS 15-415 99



CMU SCS

Q-opt steps

- bring query in internal form (eg., parse tree)
- ... into 'canonical form' (syntactic q-opt)
- generate alt. plans
- estimate cost; pick best

Faloutsos CMU SCS 15-415 100



CMU SCS

Conclusions

- Ideas to remember:
 - syntactic q-opt – do selections early
 - selectivity estimations (uniformity, indep.; histograms; join selectivity)
 - hash join (nested loops; sort-merge)
 - left-deep joins
 - dynamic programming

Faloutsos CMU SCS 15-415 101
