CSE 562
Database Systems

Query Processing:
Physical Plan Enumeration & Selection

Some slides are based or modified from originals by
Database Systems: The Complete Book,
Pearson Prentice Hall 2nd Edition

cse@buffalo

Outline – Query Optimization

- Overview
- Relational algebra level
  - Algebraic Transformations
- Detailed query plan level
  - Estimate Costs
    - Estimating size of results
    - Estimating # of IOs
  - Generate and compare plans

Generate & Compare Plans

Given a logical plan:

1. Enumerate physical alternatives (straightforward)
2. Estimate costs
3. Pick best one

⇨ **Problem:** takes too long!

Observation:
plans share pieces ("sub-plans")…

Generate & Compare Plans: Improved

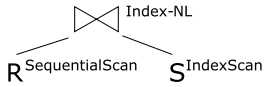
Combine Enumeration and Selection

1. Enumerate small sub-plans & estimate costs
2. Prune (remove) "sub-optimal" alternatives
3. Enumerate ways to assemble sub-plans into larger sub-plans & estimate costs
4. Prune again (keep only "optimal" sub-plans)

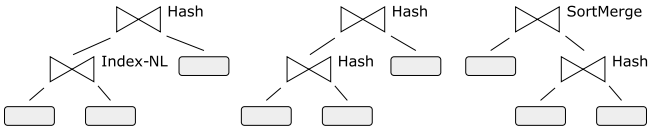
… Keep building larger "optimal" sub-plans
… Eventually generate "optimal" overall plan

Dynamic Programming Approach

- Level 1 sub-plan: join of 2 relations (plus access methods)



- Level 2 sub-plan: join of 3 relations

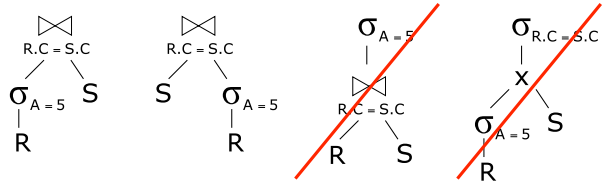


- …
- Level n sub-plan: join of n+1 relations

Generate Logical Sub-Plans

- Only generate logical sub-plans that conform to heuristic rules

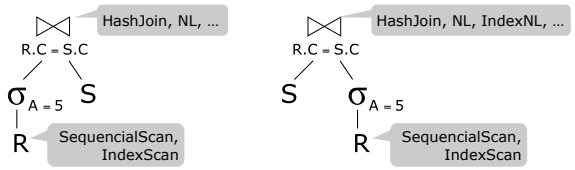
  Example:



Generate Physical Sub-Plans

- For each heuristically-chosen logical sub-plan, try all combinations of physical alternatives

  Example:



Pruning

- Estimate cost of each physical sub-plan
- For each sub-plan with identical input and output, keep only:
  - Optimal plan overall
  - Optimal plan for each "interesting order"
    - Ordered on some field
- Discard the rest

Example

```
SELECT   *
FROM     R, S, T
WHERE    R.A = S.A AND S.B = T.B
```

- T(R) = 30,000 B(R) = 300
- T(S) = 100,000 B(S) = 1000
- T(T) = 20,000 B(T) = 200

- V(S,B) = 25,000
- V(T,B) = 10,000

Example (cont.)

```
SELECT   *
FROM     R, S, T
WHERE    R.A = S.A AND S.B = T.B
```

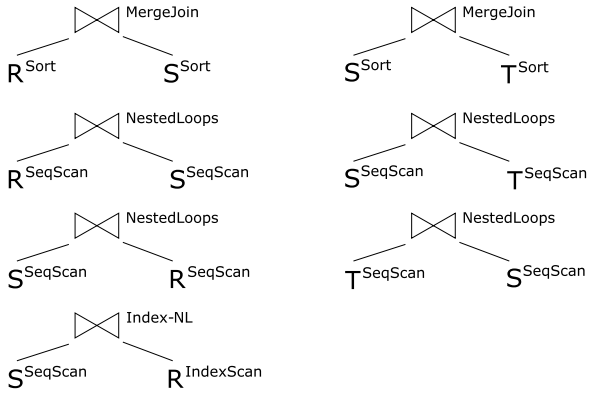
- Memory Size: M = 102
- Index on R.A (all non-leaves fit in memory)
- S.A foreign key onto R.A

Example (cont.)

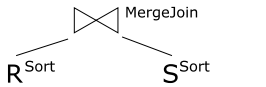
For simplicity:

- Assume: join tuple size is sum of sizes of component tuples
- Assume: always write out intermediate results
- Consider the following strategies:
  - Nested-loops
  - Index nested loops
  - Sort-Merge join

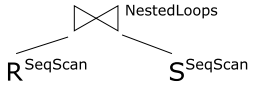
Level-1 Sub-Plans (No X-Products)



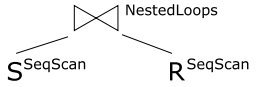
Cost of Level-1 Sub-Plans



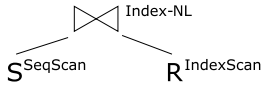
Cost = 3·(B(R) + B(S)) = 3900



$\text{Cost} = B(R) + \left\lceil \frac{B(R)}{M-2} \right\rceil \cdot B(S) = 3300$

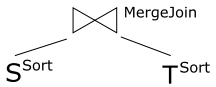


$\text{Cost} = B(S) + \left\lceil \frac{B(S)}{M-2} \right\rceil \cdot B(R) = 4000$

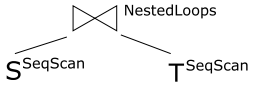


Cost = B(S) + T(S) + T(S) = 201000

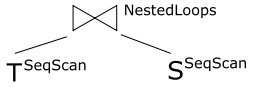
Cost of Level-1 Sub-Plans (cont.)



Cost = 3·(B(S) + B(T)) = 3600

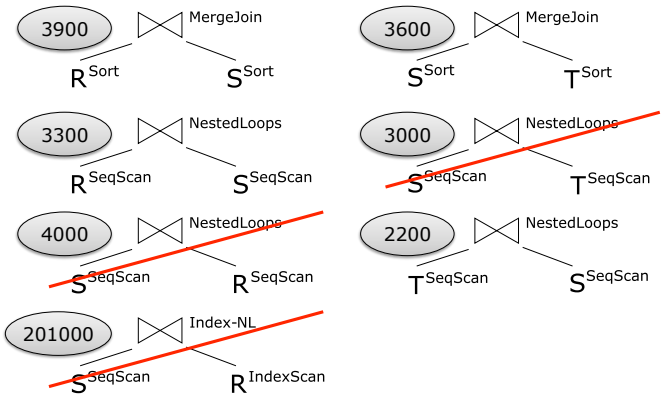


$\text{Cost} = B(S) + \left\lceil \frac{B(S)}{M-2} \right\rceil \cdot B(T) = 3000$



$\text{Cost} = B(T) + \left\lceil \frac{B(T)}{M-2} \right\rceil \cdot B(S) = 2200$

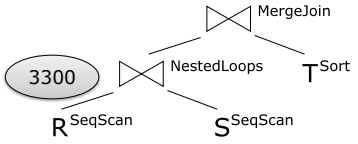
Pruning Level-1 Sub-Plans



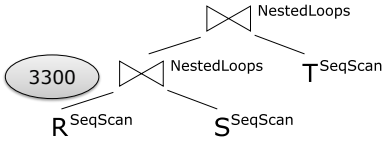
Level-2 Sub-Plans

- What is the size or R⋈S?
- Recall: S.A foreign key onto R.A
- T(R⋈S) = T(S) = 100,000
- B(R⋈S) = 2 x B(S) = 2,000

Cost of Level-2 Sub-Plans



Cost = 3300 + B(R⋈S)
+ 3·(B(R⋈S) + B(T))
= 11900

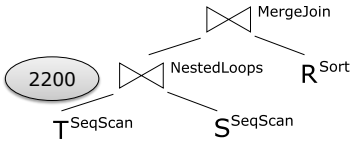


Cost = 3300 + B(R⋈S)
+ B(R⋈S) + $\left\lceil \frac{B(R⋈S)}{M-2} \right\rceil$ ·B(T)
= 11300

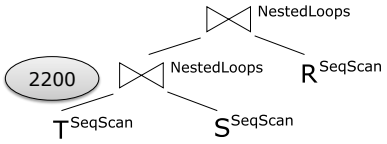
Level-2 Sub-Plans

- What is the size or S⋈T?
- T(S⋈T) = $\frac{T(S)T(T)}{\max\{V(S,B),V(T,B)\}}$ = 80,000
- B(S⋈T) = 80,000/50 = 1,600

Cost of Level-2 Sub-Plans

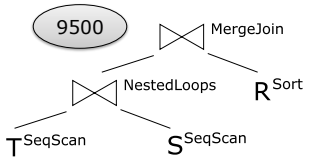


Cost = 2200 + B(S⋈T)
+ 3·(B(S⋈T) + B(R))
= 9500



Cost = 2200 + B(S⋈T)
+ B(S⋈T) + $\left\lceil \frac{B(S⋈T)}{M-2} \right\rceil$ ·B(R)
= 10200

Winner



- Selected plan: Cost = 9500
- Worst plan: Cost > 201000
- Benefit from optimizer: 20x speedup!