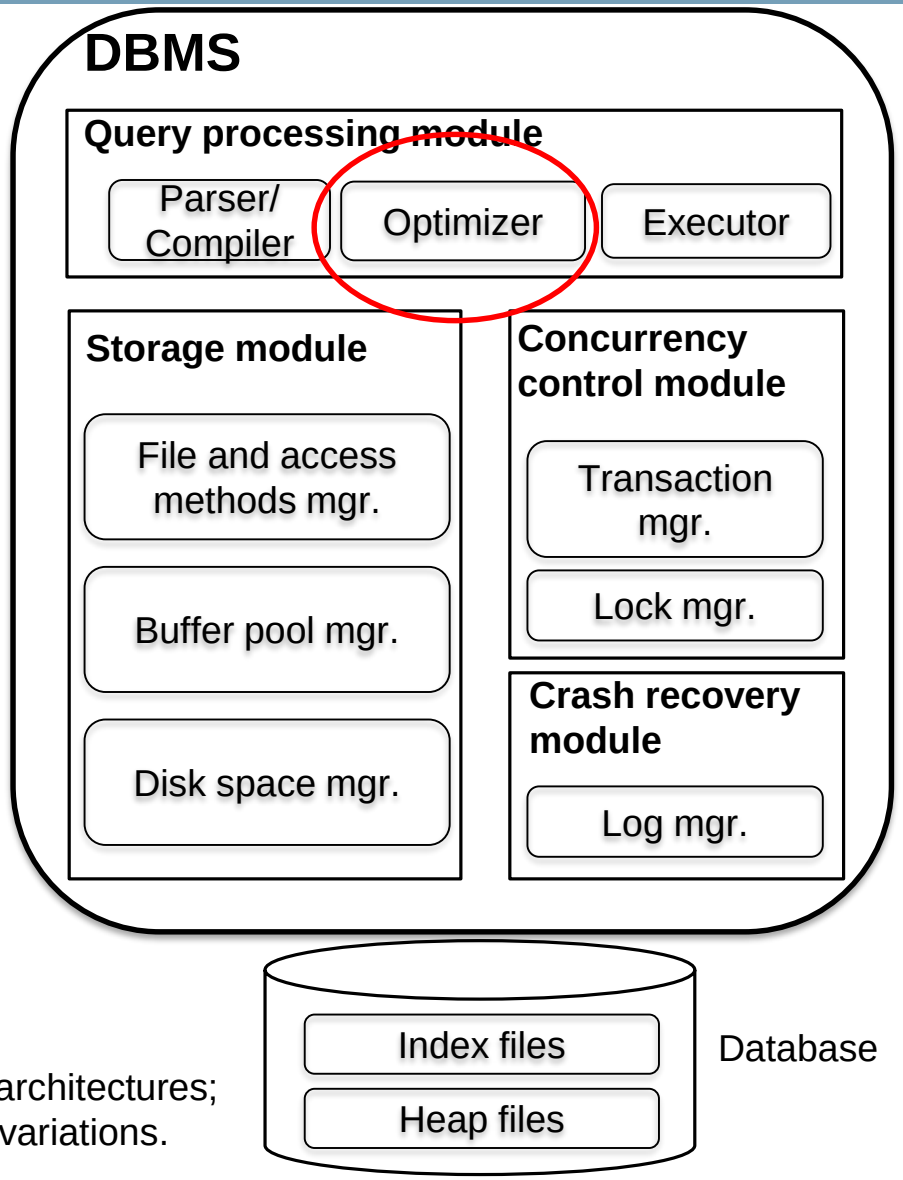
INFO20003 Database Systems

Dr Renata Borovica-Gajic

Lecture 14
Query Optimization Part II

Remember this? Components of a DBMS



TODAY
Plan enumeration

This is one of several possible architectures; each system has its own slight variations.

Coverage

- Plan enumeration and costing
- System R strategy

Readings: Chapter 15, Ramakrishnan & Gehrke, Database Systems

Enumeration of Alternative Plans

- There are two main cases:
  - Single-relation plans
  - Multiple-relation plans

- For queries over a single relation:
  - Each available access path (file scan / index) is considered, and the one with the least estimated cost is chosen
  - The different operations are essentially carried out together (e.g., if an index is used for a selection, projection is done for each retrieved tuple)

Cost Estimates for Single-Relation Plans

- Index I on primary key matches selection:
  - *Cost is Height(I)+1 for a B+ tree, about 2.2 for hash index*
- Clustered index I matching one or more selects:
  - *(NPages(I)+NPages(R)) * product of RF's of matching selects.*
- Non-clustered index I matching one or more selects:
  - *(NPages(I)+NTuples(R)) * product of RF's of matching selects*
- Sequential scan of file:
  - *NPages(R)*

  - ***Note:*** *Must also charge for duplicate elimination if required*

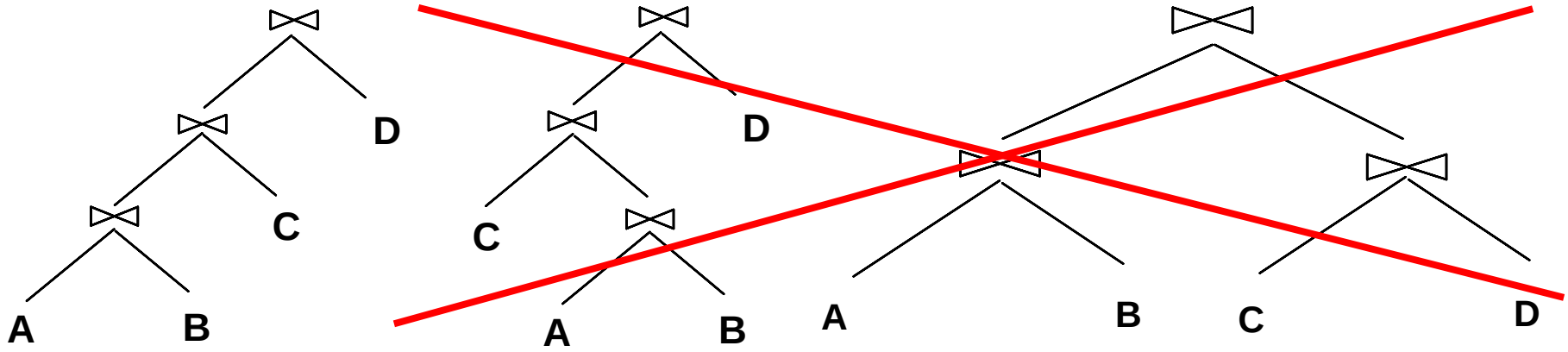
Example

```
SELECT  S.sid
FROM  Sailors S
WHERE  S.rating=8
```

- Reminder: Sailors has 500 pages, 40000 tuples
- If we have an index on *rating*:
  - Cardinality: (1/NKeys(I)) * NTuples(S) = (1/10)*40000 tuples retrieved
  - Clustered index: cost = (1/NKeys(I)) * (NPages(I)+NPages(S)) = (1/10) * (50+500) = 55 pages retrieved.
  - Unclustered index: cost = (1/NKeys(I)) * (NPages(I)+NTuples(S)) = (1/10) * (50+40000) = 4005 pages retrieved
- If we have an index on *sid*:
  - Would have to retrieve all tuples/pages. With a clustered index, the cost is 50+500, with unclustered index, 50+40000
- Doing a file scan:
  - We retrieve all file pages (500)

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
  - Left-deep trees allow us to generate all *fully pipelined* plans
    - Intermediate results are not written to temporary files
    - Not all left-deep trees are fully pipelined (e.g., SM join)



Plan Enumeration – The Hard Way

1. Select order of relations (the only degree of freedom for left-deep plans)
   - maximum possible orderings = N! (but no X-products)
2. For each join, select join algorithm
3. For each input relation, select access method

Q: How many plans for a query over N relations?

Back-of-envelope calculation:

- With 3 join algorithms, I indexes per relation:
  # plans ≈ $[N!] * [3^{(N-1)}] * [(I + 1)^N]$
- Suppose N = 3, I = 2: # plans ≈ $3! * 3^2 * 3^3 = 1458$ plans

- For each candidate plan, must estimate cost

* Query optimization is NP-complete

Plan Enumeration Example

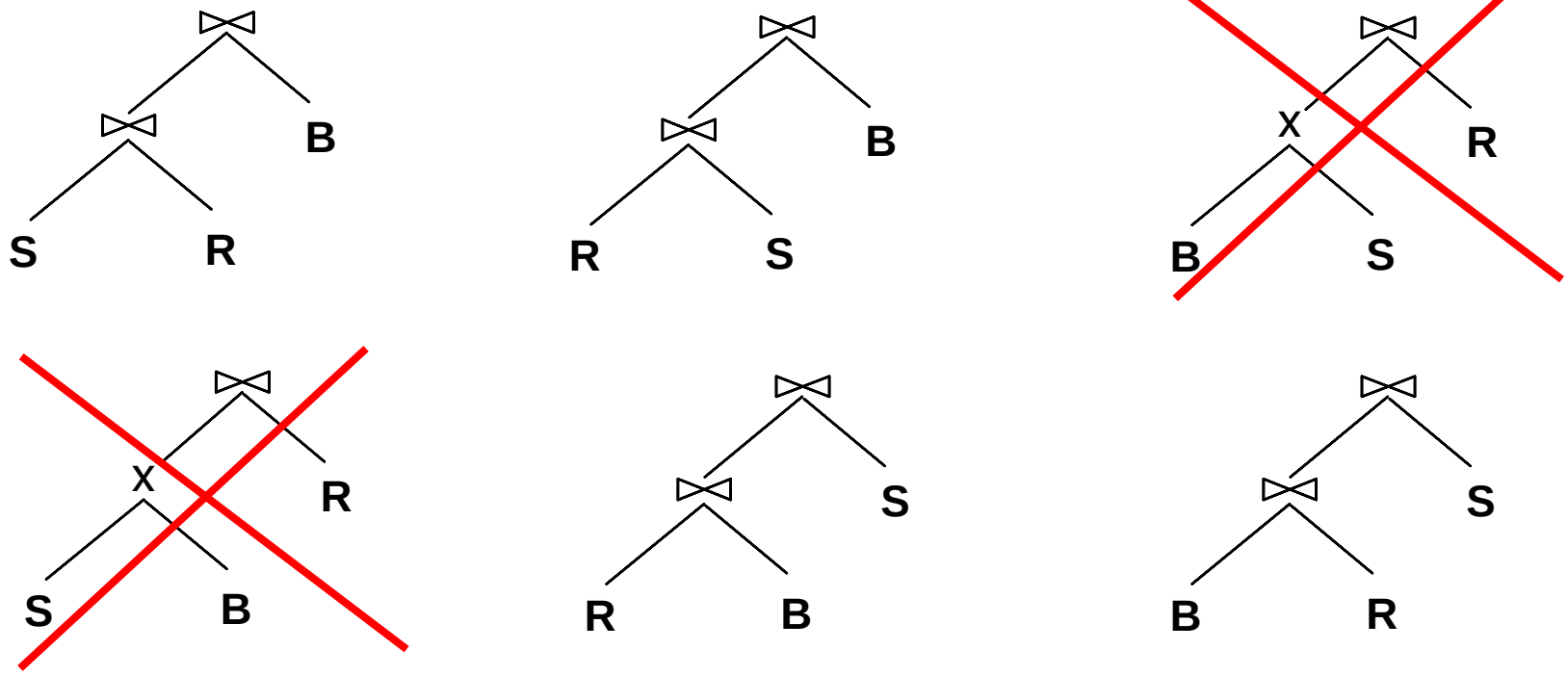
```
SELECT  S.sname, B.bname, R.day
FROM  Sailors S, Reserves R, Boats B
WHERE  S.sid = R.sid AND R.bid = B.bid
```

- Let's assume:
  - Two join algorithms to choose from:
    - Hash-Join
    - NL-Join (page-oriented or Index-NL-Join)
  - Unneeded columns removed at each stage
  - Non-clustered B+Tree index on R.sid; no other indexes
  - R.sid index has 50 pages
  - S has 500 pages, 80 tuples/page
  - R has 1000 pages, 100 tuples/page
  - B has 10 pages
  - 100 R $\bowtie$ S tuples fit on a page

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:

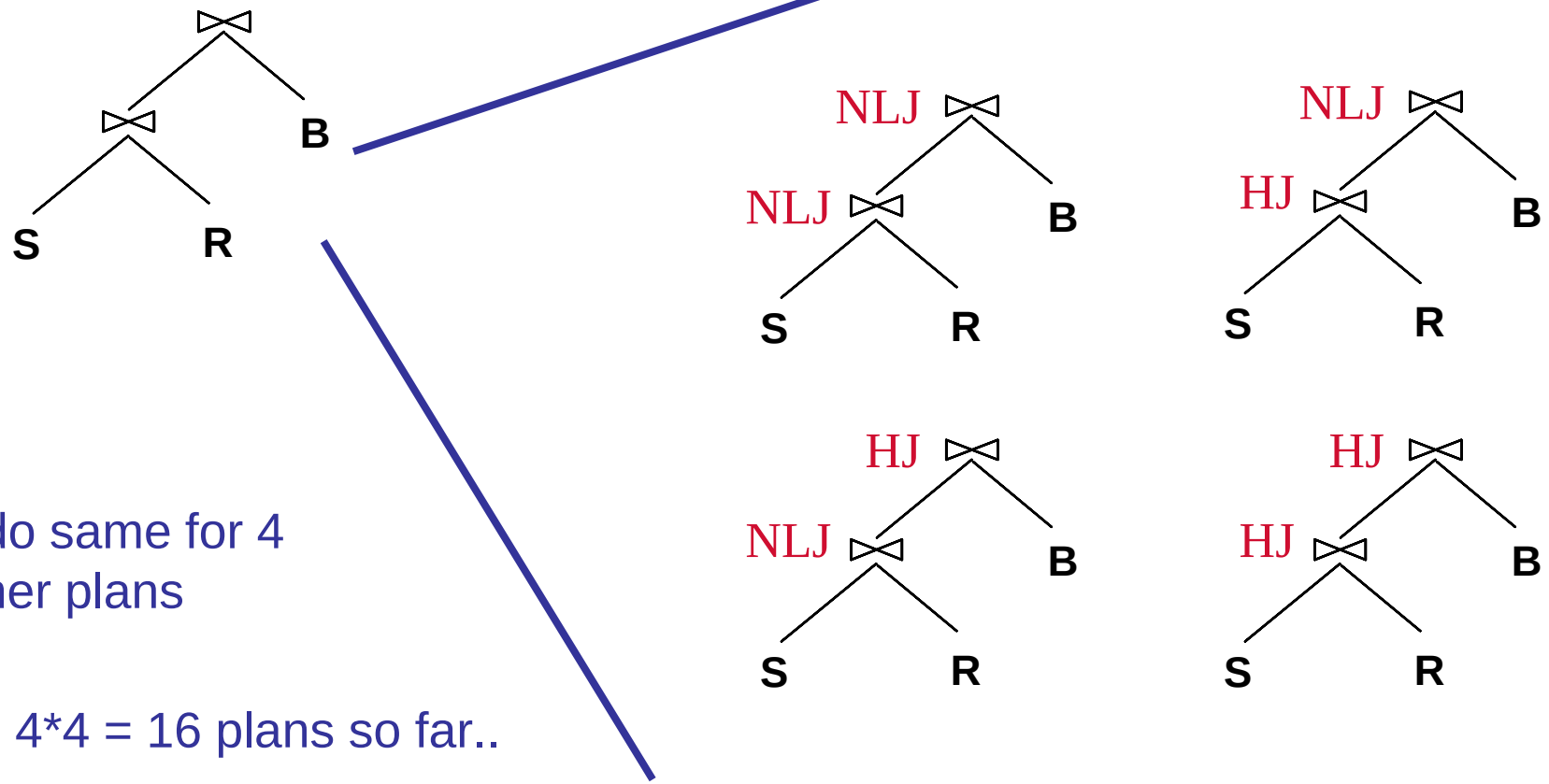


* Prune plans with cross-products immediately!

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:



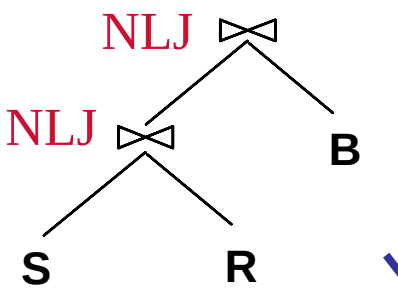
+ do same for 4 other plans

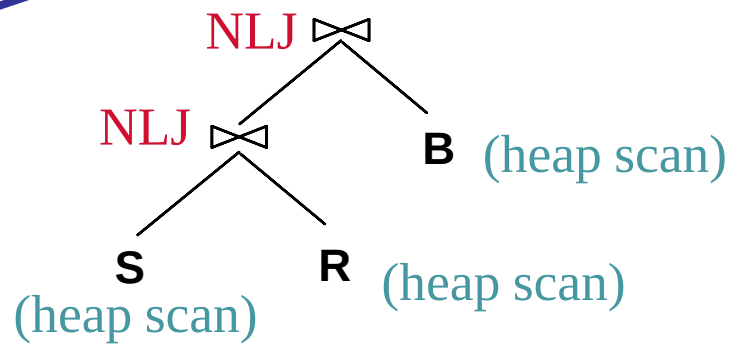
→ 4*4 = 16 plans so far..

Candidate Plans

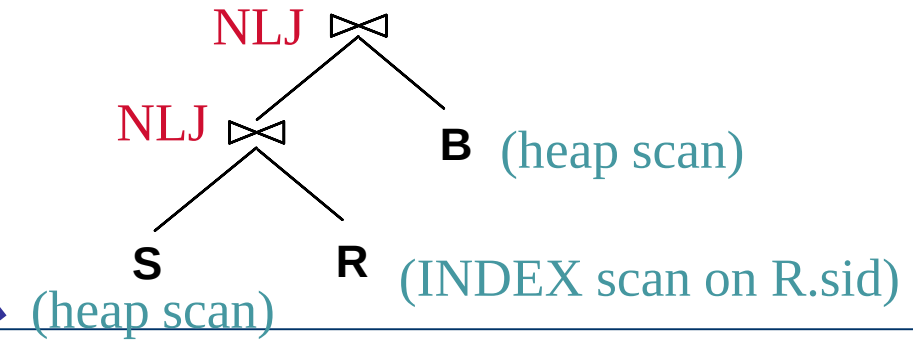
```
SELECT  S.sname, B.bname, R.day
FROM  Sailors S, Reserves R, Boats B
WHERE  S.sid = R.sid AND R.bid = B.bid
```

3. Enumerate access method choices:



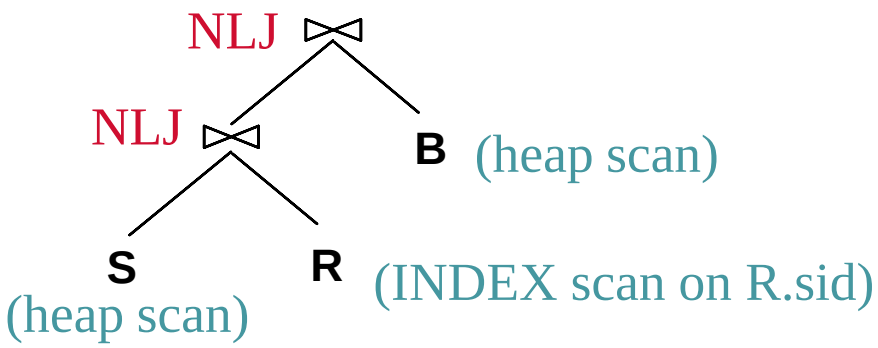


+ do same for other plans



Now estimate the cost of each plan

Example:



- Cost to scan S = 500
- Cost to join w/R = 40000 * (1/40000)(50+100,000) = 100,050
- Size of S ⋈ R = 100,000 tuples; 100,000/100 = 1000 pages
- Cost to join with B = 1000 * 10 = 10000

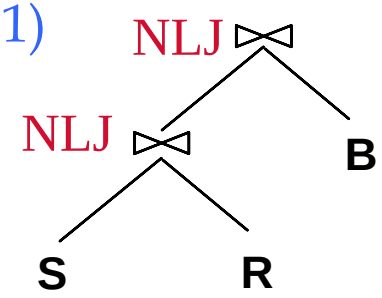
→ Total estimated cost = 500 + 100,050 + 10000 = 110,550

Now You Try …

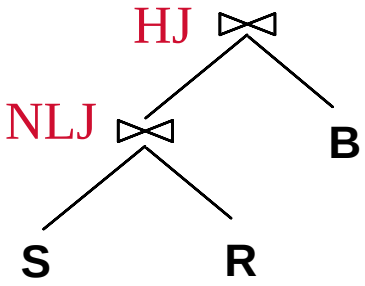
Estimate the cost of each of these plans:

S = Sailors, R = Reserves
B = Boats

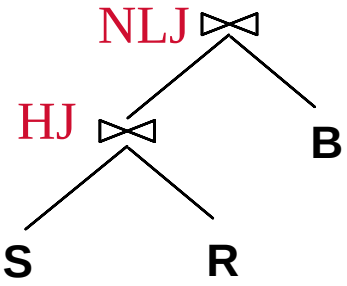
1)



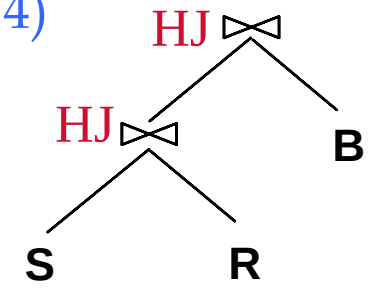
2)



3)



4)



Relevant stats:

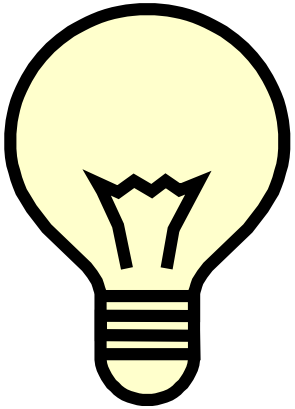
- S has 500 pages, 80 tuples/page
- R has 1000 pages, 100 tuples/page
- B has 10 pages
- 100 S ⋈ R tuples fit on a page

Join algorithms:

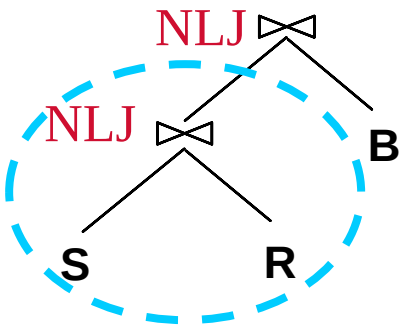
- NLJ = page-oriented NL Join
  - Scan left input + scan right input once per page in left input
- HJ = hash-join (assume 2 passes)
  - Scan both inputs + write both inputs in buckets + read all buckets

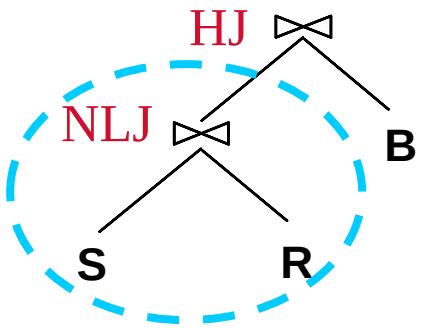
Notice Anything?

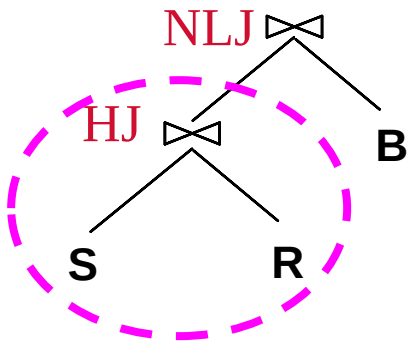
- Much of the computation is redundant
- Idea: when we estimate costs & result sizes of sub-plans, remember them.

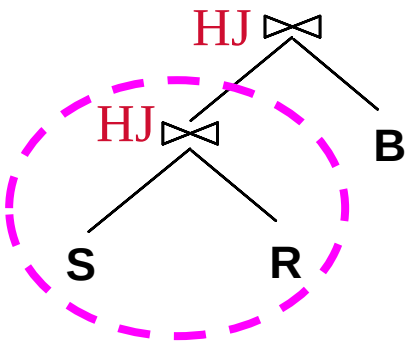


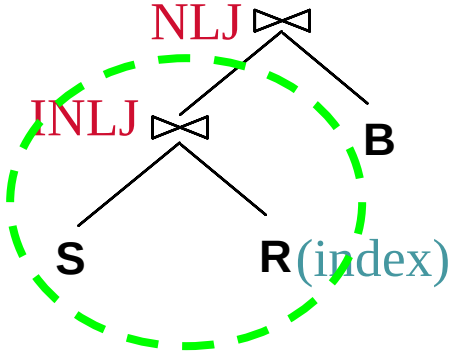
Enumerated Plans (just the S-R-B ones)

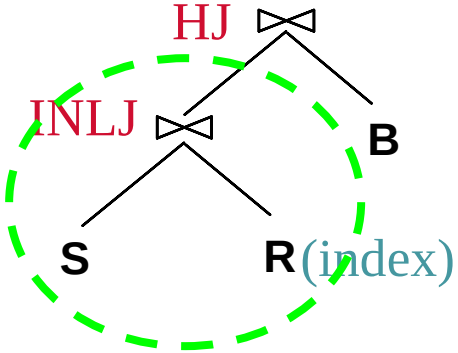












* Observe that many plans share common sub-plans (i.e., only upper part differs)

Query Optimization

- Plan enumeration and costing
- System R strategy

Readings: Chapter 15, Ramakrishnan & Gehrke, Database Systems

Improved Strategy (used in System R)

- Shared sub-plan observation suggests a better strategy:
- Enumerate plans using N passes (N = # relations joined):
  - Pass 1: Find best 1-relation plans for each relation
  - Pass 2: Find best ways to join result of each 1-relation plan as outer to another relation *(All 2-relation plans.)*
  - Pass N: Find best ways to join result of a (N-1)-relation plan as outer to the Nth relation *(All N-relation plans.)*
- For each subset of relations, retain only:
  - Cheapest subplan overall (possibly unordered), plus
  - Cheapest subplan for each *interesting order* of the tuples
- For each subplan retained, remember cost and result size estimates

A Note on ”Interesting Orders”

- An intermediate result has an “interesting order” if it is sorted by any of:
  - ORDER BY attributes
  - GROUP BY attributes
  - Join attributes of other joins

System R Plan Enumeration

- A N-1 way plan is not combined with an additional relation unless there is a join condition between them (unless all predicates in WHERE have been used up)
  - i.e., avoid Cartesian products if possible

- Always push all selections & projections as far down in the plans as possible
  - Usually a good strategy, as long as these operations are cheap

System R Plan Enumeration Example

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

- This time let's assume:
  - Two join algorithms to choose from:
    - Sort-Merge-Join
    - NL-Join (page-oriented or Index-NL-Join)
  - Clustered B+Tree on S.sid (height=3; 500 leaf pages)
  - S has 10,000 pages, 5 tuples/page
  - R has 10 pages, 10 tuples/page
  - B has 10 pages, 20 tuples/page
  - 10 R $\bowtie$ S tuples fit on a page
  - 10 R $\bowtie$ B tuples fit on a page

Pass 1 (single-relation subplans)

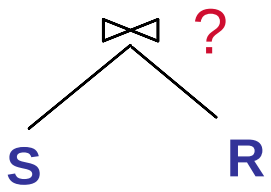
- S: (a) heap scan or (b) scan index on S.sid
  a) heap scan cost = 10,000
  b) index scan cost = 500 + 10,000 = 10,500
  Retain both, since (b) has “interesting order” by sid

- R: heap scan only option
  Cost = 10

- B: heap scan only option
  Cost = 10

Pass 2 (2-relation subplans)

Starting with S as outer



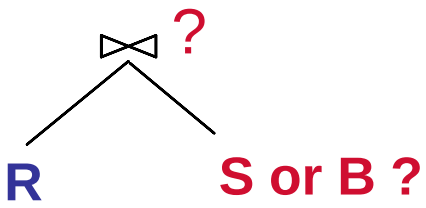
- Heap scan-S as outer:
  - a) NL-Join with R, cost = 10,000 + 10,000(10) = 110,000
  - b) SM-Join with R, cost = 10,000 + 2*10,000 + 3*10 = 30,030
- Index scan-S as outer:
  - c) NL-Join with R, cost = 10,500 + 10,000(10) = 110,500
  - d) SM-Join with R, cost = 10,500 + 3*10 = 10,530

- Retain (d) only

*** Note: best S ⋈ R plan exploits "interesting order" of non-optimal subplan !**

Pass 2 (continued)

Starting with R as outer

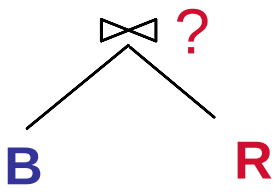


- Join with S:
  - a) NL-Join with S, cost = 10 + 10(10,000) = 100,010
  - b) Index-NL-Join with Index-S, cost = 10 + 100*4 = 410
  - c) SM-Join with S, cost = 10 + 2*10 + 3*10,000 = 30,030
  - d) SM-Join with Index-S, cost = 10 + 2*10 + 10,500 = 10,530

- Join with B:
  - a) NL-Join with B, cost = 10 + 10(10) = 110
  - b) SM-Join with B, cost = 10 + 2*10 + 3*10 = 60

Pass 2 (continued)

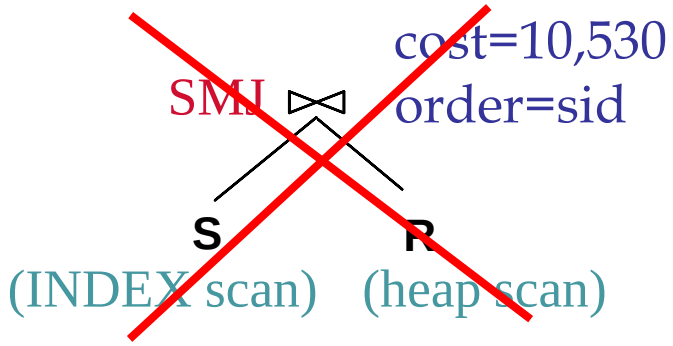
Starting with B as outer



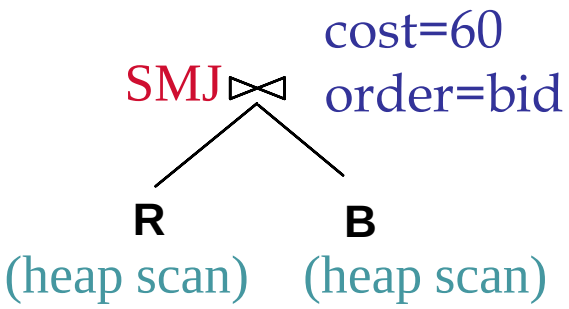
- Join with R:
  - a) NL-Join with R, cost = 10 + 10(10) = 110
  - b) SM-Join with R, cost = 10 + 2*10 + 3*10 = 60

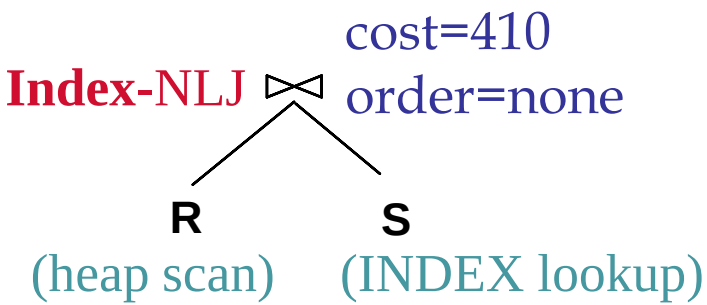
Further pruning of 2-relation subplans

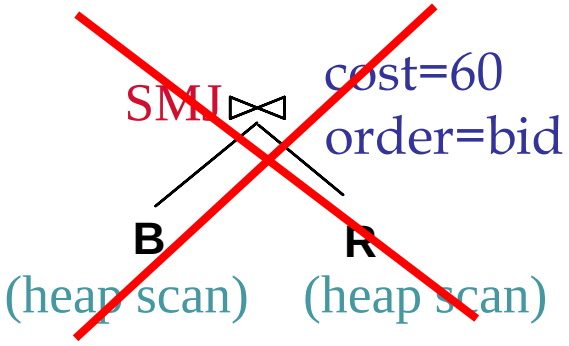
S ⋈ R:



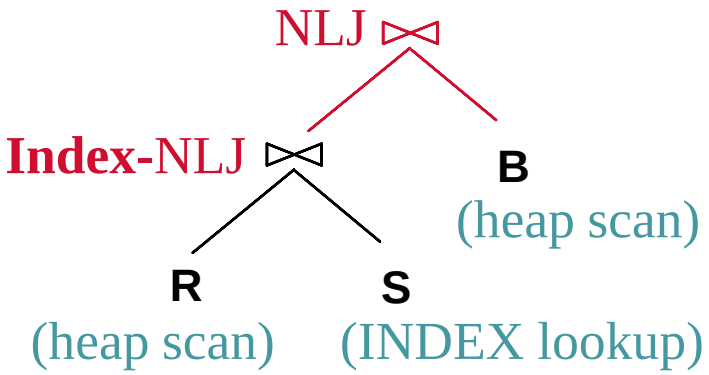
B ⋈ R:





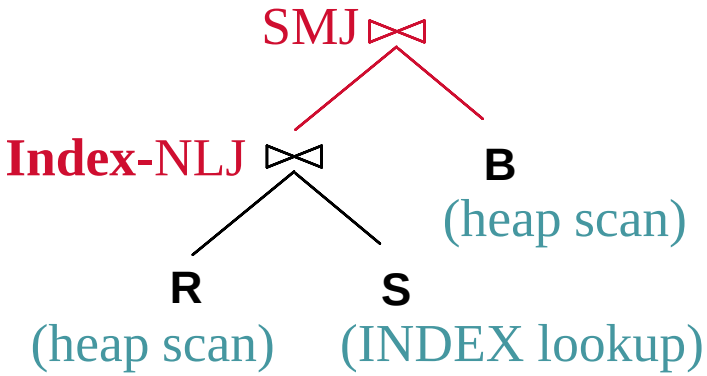


Pass 3 (3-relation subplans)



cost = 410 + 10(10) = 510

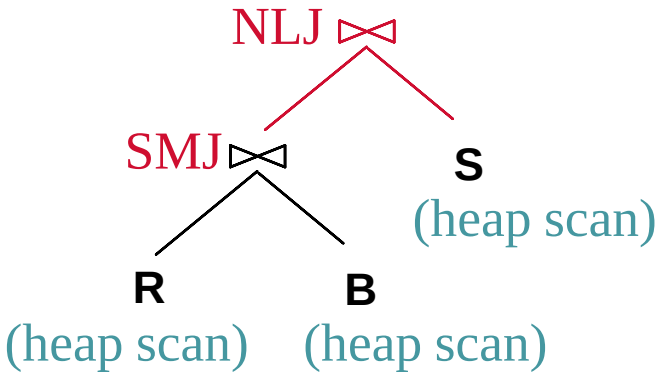
S ⋈ R subplan:
cost=410
order=none
result size = 10 pages



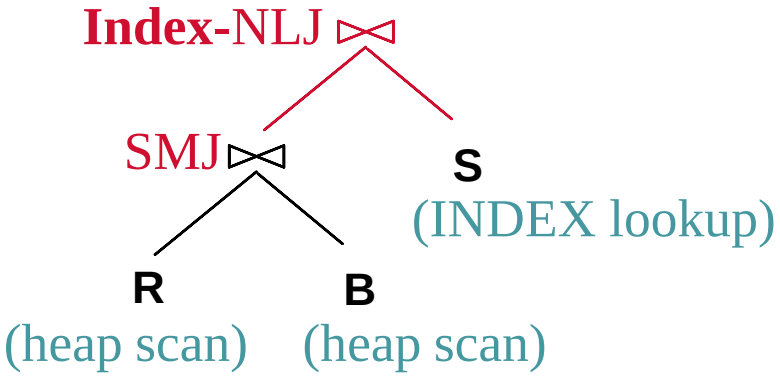
cost = 410 + 2*10 + 3*10 = 460

Pass 3 (continued)

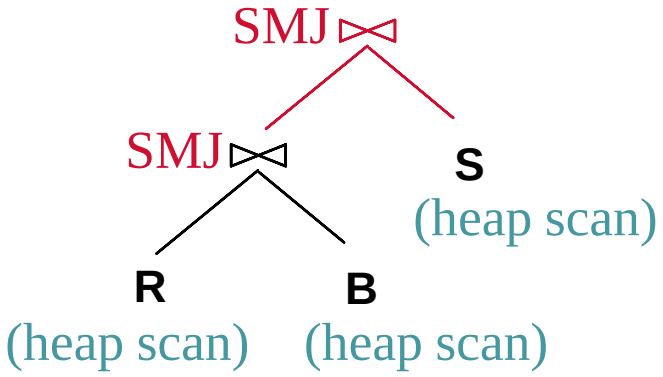
B⋈R subplan: cost=60, order=bid
result size = 100 tuples (10 pages)



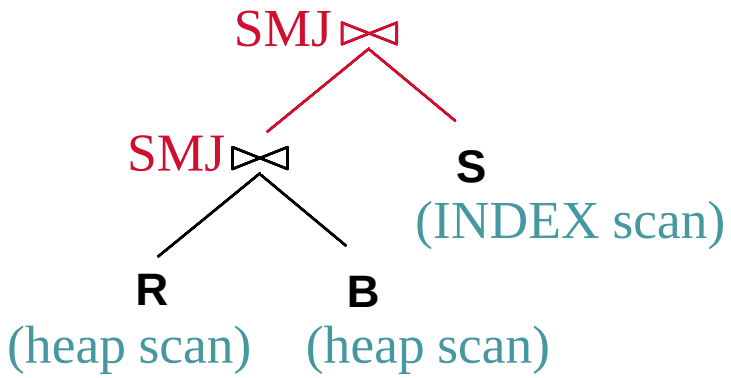
cost = 60 + 10(10,000) = 100,060



cost = 60 + 100*4 = 460

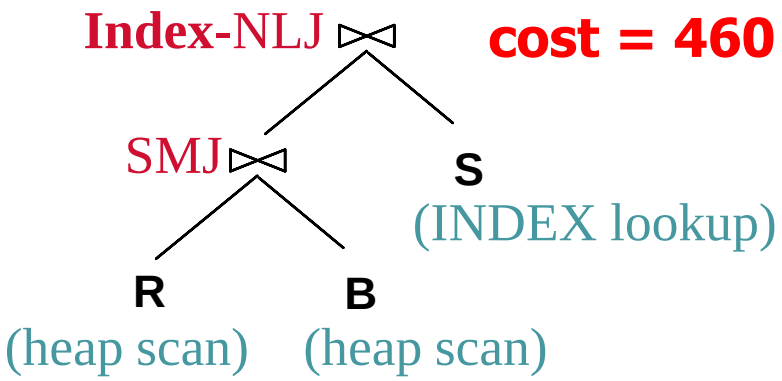


cost = 60 + 10*2 + 3*10,000 = 30,080



cost = 60 + 10*2 + 10,500 = 10,580

And the Winner is …



- Observations:
  - Best plan mixes join algorithms
  - Worst plan had cost > 100,000
    (exact cost unknown due to pruning)

*** Optimization yielded ~ 1000-fold improvement over worst plan!**

Some notes w.r.t. reality…

- In spite of pruning plan space, this approach is still exponential in the # of tables
  - Rule of thumb: works well for < 10 joins

- In real systems, COST considered is:

  #IOs + *factor* * #CPU Instructions

System R strategy: Summary

- Enumerate plans using N passes (N = # relations joined):
- For each subset of relations, retain only:
  - Cheapest subplan overall (possibly unordered), plus
  - Cheapest subplan for each *interesting order* of the tuples
- For each subplan retained, remember cost and result size estimates

What's examinable

- Understand plan enumeration and costing
- Discuss System R strategy

- Important for Assignment 3 as well

Next Lecture

- Mid-semester test
  - Good luck to all of you!
  - And bye for now ;)