

# CSE 444: Database Internals

## Lecture 11

### Query Optimization (part 2)

# Reminders

- Homework 2 due tonight by 11pm in drop box
- Lab 2 is due on Monday by 11pm

# Query Optimizer Overview

- **Input:** A logical query plan
- **Output:** A good physical query plan
- **Basic query optimization algorithm**
  - Enumerate alternative plans (logical and physical)
  - Compute estimated cost of each plan
    - Compute number of I/Os
    - Optionally take into account other resources
  - Choose plan with lowest cost
  - This is called cost-based optimization

# The Three Parts of an Optimizer

- Cost estimation
  - Based on cardinality estimation
- Search space
  - Algebraic laws, restricted types of join trees
- Search algorithm
  - Will discuss next

# Search Algorithm

- Dynamic programming (in class)
  - Based on System R (aka Selinger) style optimizer[1979]
  - Limited to joins: *join reordering algorithm*
  - Bottom-up
- Rule-based algorithm (will not discuss)
  - Database of rules (=algebraic laws)
  - Usually: dynamic programming
  - Usually: top-down

# Dynamic Programming

Originally proposed in System R [1979]

- Only handles single block queries:

```
SELECT list  
FROM   R1, ..., Rn  
WHERE  cond1 AND cond2 AND . . . AND condk
```

- Some heuristics for search space enumeration:
  - Selections down
  - Projections up
  - Avoid cartesian products

```
SELECT list  
FROM R1, ..., Rn  
WHERE cond1 AND cond2 AND . . . AND condk
```

# Dynamic Programming

- For each subquery  $Q \subseteq \{R1, \dots, Rn\}$  compute the following:
  - $T(Q)$  = the estimated size of  $Q$
  - $\text{Plan}(Q)$  = a best plan for  $Q$
  - $\text{Cost}(Q)$  = the estimated cost of that plan

```
SELECT list  
FROM R1, ..., Rn  
WHERE cond1 AND cond2 AND . . . AND condk
```

# Dynamic Programming

- **Step 1:** For each  $\{R_i\}$  do:
  - $T(\{R_i\}) = T(R_i)$
  - $\text{Plan}(\{R_i\}) = \text{access method for } R_i$
  - $\text{Cost}(\{R_i\}) = \text{cost of access method for } R_i$



```
SELECT list  
FROM R1, ..., Rn  
WHERE cond1 AND cond2 AND . . . AND condk
```

# Dynamic Programming

- **Step 2:** For each  $Q \subseteq \{R_1, \dots, R_n\}$  of size  $k$  do:
  - $T(Q)$  = use estimator
  - Consider all partitions  $Q = Q' \cup Q''$   
compute  $\text{cost}(\text{Plan}(Q') \bowtie \text{Plan}(Q''))$
  - $\text{Cost}(Q)$  = the smallest such cost
  - $\text{Plan}(Q)$  = the corresponding plan
- Note
  - If we restrict to left-linear trees:  $Q''$  = single relation
  - May want to avoid cartesian products

```
SELECT list  
FROM R1, ..., Rn  
WHERE cond1 AND cond2 AND . . . AND condk
```

# Dynamic Programming

- **Step 3:** Return Plan( $\{R_1, \dots, R_n\}$ )

# Example

```
SELECT *  
FROM   R, S, T, U  
WHERE  cond1 AND cond2 AND . . .
```

- $R \bowtie S \bowtie T \bowtie U$
- Assumptions:

```
T(R) = 2000  
T(S) = 5000  
T(T) = 3000  
T(U) = 1000
```

- Every join selectivity is 0.001

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T    | Plan | Cost |
|----------|------|------|------|
| R        | 2000 |      |      |
| S        | 5000 |      |      |
| T        | 3000 |      |      |
| U        | 1000 |      |      |
| RS       |      |      |      |
| RT       |      |      |      |
| RU       |      |      |      |
| ST       |      |      |      |
| SU       |      |      |      |
| TU       |      |      |      |
| RST      |      |      |      |
| RSU      |      |      |      |
| RTU      |      |      |      |
| STU      |      |      |      |
| RSTU     |      |      |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan | Cost |
|----------|-------|------|------|
| R        | 2000  |      |      |
| S        | 5000  |      |      |
| T        | 3000  |      |      |
| U        | 1000  |      |      |
| RS       | 10000 |      |      |
| RT       | 6000  |      |      |
| RU       | 2000  |      |      |
| ST       | 15000 |      |      |
| SU       | 5000  |      |      |
| TU       | 3000  |      |      |
| RST      | 30000 |      |      |
| RSU      | 10000 |      |      |
| RTU      | 6000  |      |      |
| STU      | 15000 |      |      |
| RSTU     | 30000 |      |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                     | Cost |
|----------|-------|--------------------------|------|
| R        | 2000  | Clustered index scan R.A | 200  |
| S        | 5000  |                          |      |
| T        | 3000  |                          |      |
| U        | 1000  |                          |      |
| RS       | 10000 |                          |      |
| RT       | 6000  |                          |      |
| RU       | 2000  |                          |      |
| ST       | 15000 |                          |      |
| SU       | 5000  |                          |      |
| TU       | 3000  |                          |      |
| RST      | 30000 |                          |      |
| RSU      | 10000 |                          |      |
| RTU      | 6000  |                          |      |
| STU      | 15000 |                          |      |
| RSTU     | 30000 |                          |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                     | Cost |
|----------|-------|--------------------------|------|
| R        | 2000  | Clustered index scan R.A | 200  |
| S        | 5000  | Table scan               | 500  |
| T        | 3000  |                          |      |
| U        | 1000  |                          |      |
| RS       | 10000 |                          |      |
| RT       | 6000  |                          |      |
| RU       | 2000  |                          |      |
| ST       | 15000 |                          |      |
| SU       | 5000  |                          |      |
| TU       | 3000  |                          |      |
| RST      | 30000 |                          |      |
| RSU      | 10000 |                          |      |
| RTU      | 6000  |                          |      |
| STU      | 15000 |                          |      |
| RSTU     | 30000 |                          |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                       | Cost |
|----------|-------|----------------------------|------|
| R        | 2000  | Clustered index scan R.A   | 200  |
| S        | 5000  | Table scan                 | 500  |
| T        | 3000  | Table scan                 | 300  |
| U        | 1000  | Unclustered index scan U.F | 1000 |
| RS       | 10000 |                            |      |
| RT       | 6000  |                            |      |
| RU       | 2000  |                            |      |
| ST       | 15000 |                            |      |
| SU       | 5000  |                            |      |
| TU       | 3000  |                            |      |
| RST      | 30000 |                            |      |
| RSU      | 10000 |                            |      |
| RTU      | 6000  |                            |      |
| STU      | 15000 |                            |      |
| RSTU     | 30000 |                            |      |



$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                           | Cost |
|----------|-------|--------------------------------|------|
| R        | 2000  | Clustered index scan R.A       | 200  |
| S        | 5000  | Table scan                     | 500  |
| T        | 3000  | Table scan                     | 300  |
| U        | 1000  | Unclustered index scan U.F     | 1000 |
| RS       | 10000 | $R \bowtie S$ nested loop join | ...  |
| RT       | 6000  |                                |      |
| RU       | 2000  |                                |      |
| ST       | 15000 |                                |      |
| SU       | 5000  |                                |      |
| TU       | 3000  |                                |      |
| RST      | 30000 |                                |      |
| RSU      | 10000 |                                |      |
| RTU      | 6000  |                                |      |
| STU      | 15000 |                                |      |
| RSTU     | 30000 |                                |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                           | Cost |
|----------|-------|--------------------------------|------|
| R        | 2000  | Clustered index scan R.A       | 200  |
| S        | 5000  | Table scan                     | 500  |
| T        | 3000  | Table scan                     | 300  |
| U        | 1000  | Unclustered index scan U.F     | 1000 |
| RS       | 10000 | $R \bowtie S$ nested loop join | ...  |
| RT       | 6000  | $R \bowtie T$ index join       | ...  |
| RU       | 2000  |                                |      |
| ST       | 15000 |                                |      |
| SU       | 5000  |                                |      |
| TU       | 3000  |                                |      |
| RST      | 30000 |                                |      |
| RSU      | 10000 |                                |      |
| RTU      | 6000  |                                |      |
| STU      | 15000 |                                |      |
| RSTU     | 30000 |                                |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                           | Cost |
|----------|-------|--------------------------------|------|
| R        | 2000  | Clustered index scan R.A       | 200  |
| S        | 5000  | Table scan                     | 500  |
| T        | 3000  | Table scan                     | 300  |
| U        | 1000  | Unclustered index scan U.F     | 1000 |
| RS       | 10000 | $R \bowtie S$ nested loop join | ...  |
| RT       | 6000  | $R \bowtie T$ index join       | ...  |
| RU       | 2000  | $R \bowtie U$ index join       |      |
| ST       | 15000 | $S \bowtie T$ hash join        |      |
| SU       | 5000  | ...                            |      |
| TU       | 3000  | ...                            |      |
| RST      | 30000 |                                |      |
| RSU      | 10000 |                                |      |
| RTU      | 6000  |                                |      |
| STU      | 15000 |                                |      |
| RSTU     | 30000 |                                |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                           | Cost |
|----------|-------|--------------------------------|------|
| R        | 2000  | Clustered index scan R.A       | 200  |
| S        | 5000  | Table scan                     | 500  |
| T        | 3000  | Table scan                     | 300  |
| U        | 1000  | Unclustered index scan U.F     | 1000 |
| RS       | 10000 | $R \bowtie S$ nested loop join | ...  |
| RT       | 6000  | $R \bowtie T$ index join       | ...  |
| RU       | 2000  | $R \bowtie U$ index join       |      |
| ST       | 15000 | $S \bowtie T$ hash join        |      |
| SU       | 5000  | ...                            |      |
| TU       | 3000  | ...                            |      |
| RST      | 30000 | $(RT) \bowtie S$ hash join     | ...  |
| RSU      | 10000 | $(SU) \bowtie R$ merge join    |      |
| RTU      | 6000  | ...                            |      |
| STU      | 15000 |                                |      |
| RSTU     | 30000 |                                |      |

$T(R) = 2000$   
 $T(S) = 5000$   
 $T(T) = 3000$   
 $T(U) = 1000$

Assume  
 $B(..) = T(..)/10$

| Subquery | T     | Plan                           | Cost |
|----------|-------|--------------------------------|------|
| R        | 2000  | Clustered index scan R.A       | 200  |
| S        | 5000  | Table scan                     | 500  |
| T        | 3000  | Table scan                     | 300  |
| U        | 1000  | Unclustered index scan U.F     | 1000 |
| RS       | 10000 | $R \bowtie S$ nested loop join | ...  |
| RT       | 6000  | $R \bowtie T$ index join       | ...  |
| RU       | 2000  | $R \bowtie U$ index join       |      |
| ST       | 15000 | $S \bowtie T$ hash join        |      |
| SU       | 5000  | ...                            |      |
| TU       | 3000  | ...                            |      |
| RST      | 30000 | $(RT) \bowtie S$ hash join     | ...  |
| RSU      | 10000 | $(SU) \bowtie R$ merge join    |      |
| RTU      | 6000  | ...                            |      |
| STU      | 15000 |                                |      |
| RSTU     | 30000 | $(RT) \bowtie (SU)$ hash join  | ...  |

# Discussion

- Need to consider both  $R \bowtie S$  and  $S \bowtie R$ 
  - Because the cost may be different!
- When computing the cheapest plan for  $(Q) \bowtie R$ , we may consider new access methods for  $R$ , e.g. an index look-up that makes sense only in the context of the join

```
SELECT list  
FROM R1, ..., Rn  
WHERE cond1 AND cond2 AND . . . AND condk
```

# Discussion

Given a query with  $n$  relations  $R_1, \dots, R_n$

- How many entries do we have in the dynamic programming table?
- For each entry, how many alternative plans do we need to inspect?

```
SELECT list  
FROM R1, ..., Rn  
WHERE cond1 AND cond2 AND . . . AND condk
```

# Discussion

Given a query with  $n$  relations  $R_1, \dots, R_n$

- How many entries do we have in the dynamic programming table?
  - A:  $2^n - 1$
- For each entry, how many alternative plans do we need to inspect?
  - A: for each entry with  $k$  tables, examine  $2^k - 2$  plans

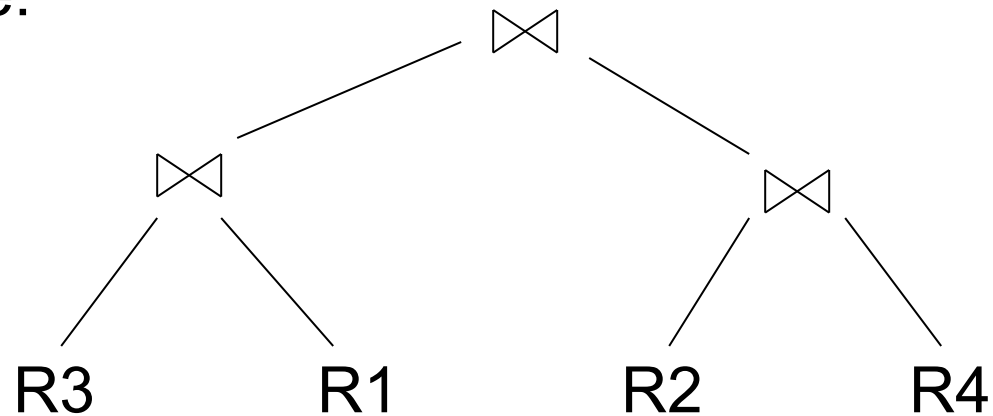


# Reducing the Search Space

- Left-linear trees
- No cartesian products

# Join Trees

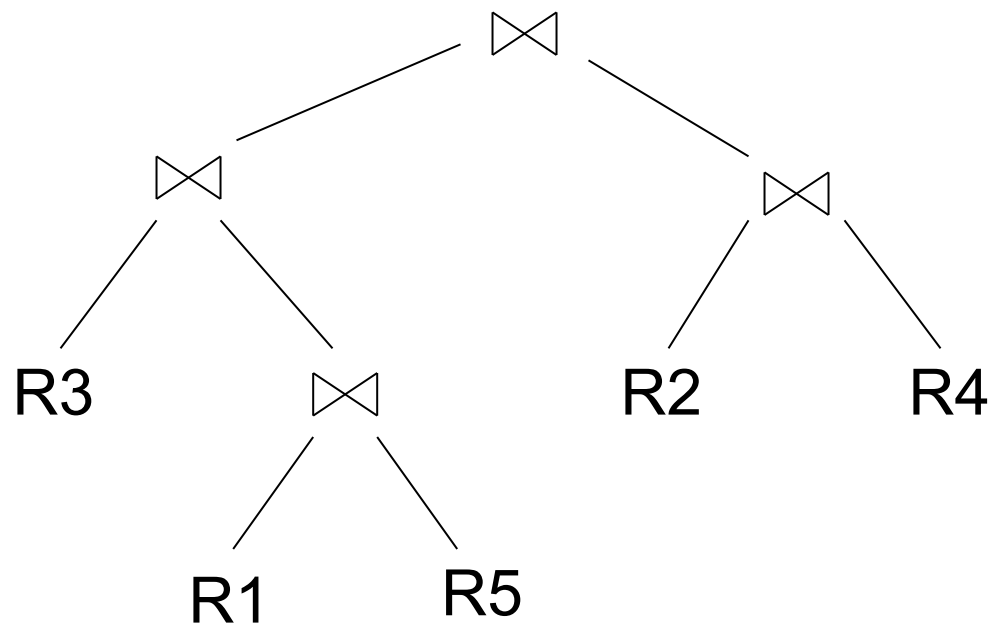
- $R1 \bowtie R2 \bowtie \dots \bowtie Rn$
- Join tree:



- A plan = a join tree
- A partial plan = a subtree of a join tree

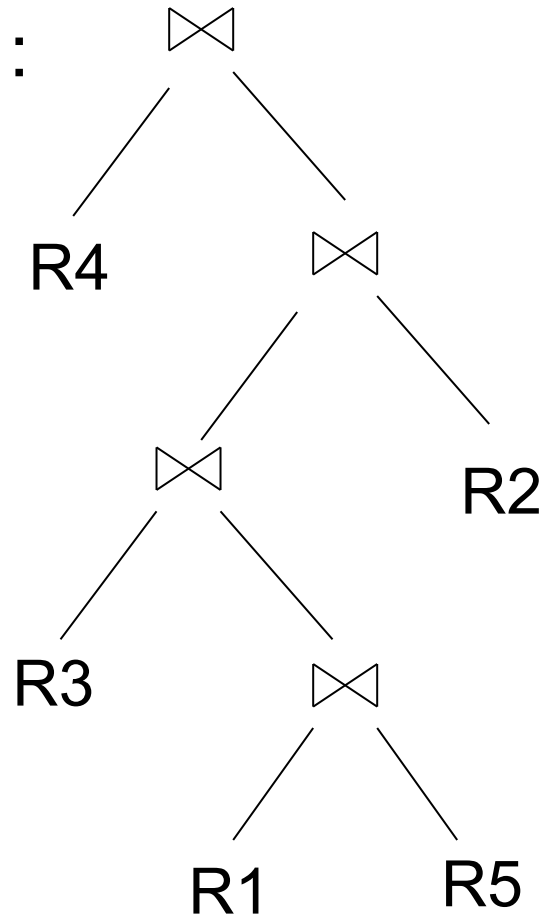
# Types of Join Trees

- Bushy:



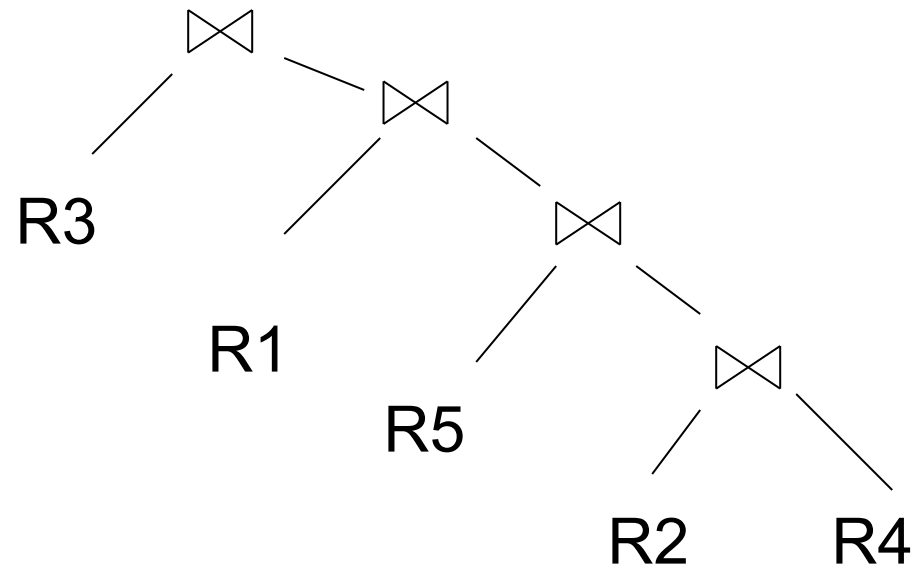
# Types of Join Trees

- Linear :



# Types of Join Trees

- Right deep:



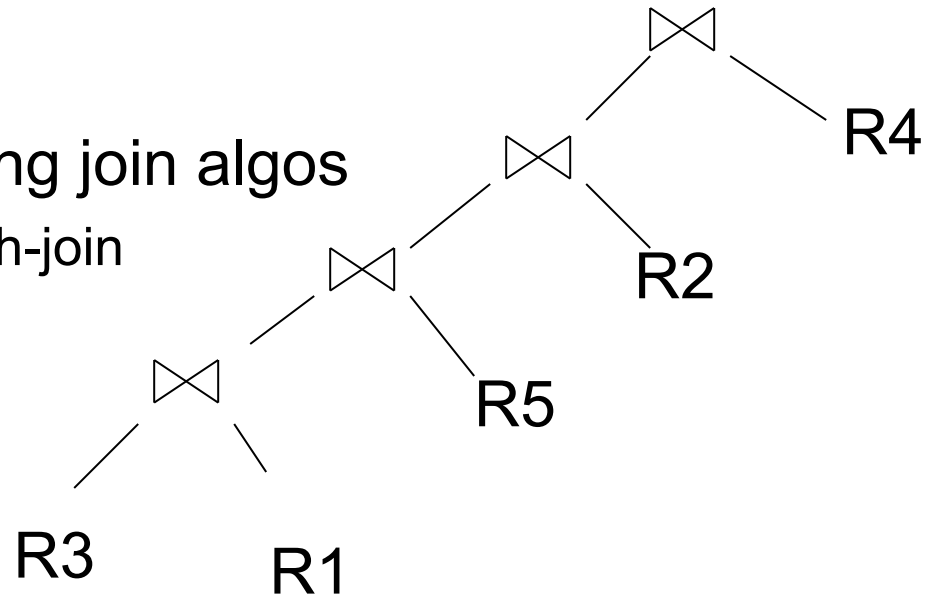
# Types of Join Trees

- Left deep:

- Work well with existing join algos

- Nested-loop and hash-join

- Facilitate pipelining



- Dynamic programming can be used with all trees

# Selinger Algorithm

Selinger enumeration algorithm considers

- Different logical and physical plans *at the same time*
- Cost of a plan is IO + CPU
- Concept of *interesting order* during plan enumeration
  - Same order as that requested by ORDER BY or GROUP BY
  - Attributes that appear in eq-join predicates

# More about the Selinger Algorithm

- Step 1: Enumerate all access paths for a single relation
  - File scan or index scan
  - Keep the cheapest for each *interesting order*
- Step 2: Consider all ways to join two relations
  - Use result from step 1 as the outer relation
  - Consider every other possible relation as inner relation
  - Estimate cost when using sort-merge or nested-loop join
  - Keep the cheapest for each *interesting order*
- Steps 3 and later: Repeat for three relations, etc.



# Example on the Board