Worst Case Optimal Joins

CompSci 590.04

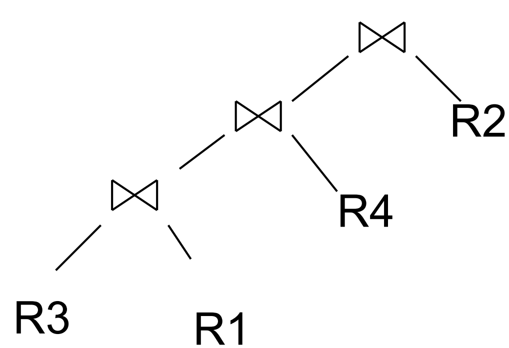
Instructor: Ashwin Machanavajjhala

Duke UNIVERSITY

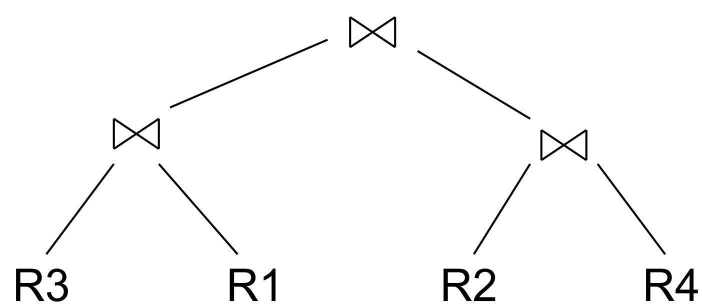
Multi-way Joins

J(a,b,c) :- R(a,b) S(b,c) T(a,c)

- Historically databases designers decided that the best way to handle multi-way joins is to do them one pair at a time.
  - For efficiency reasons.



Left-deep plan



Bushy plan

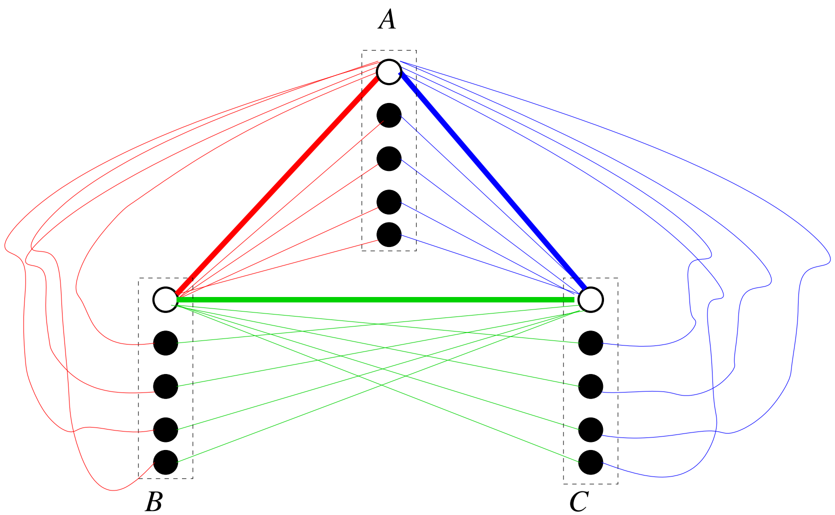
Duke UNIVERSITY

How fast is this approach?

$$R = \{a_0\} \times \{b_0, \ldots, b_m\} \cup \{a_0, \ldots, a_m\} \times \{b_0\}$$

$$S = \{b_0\} \times \{c_0, \ldots, c_m\} \cup \{b_0, \ldots, b_m\} \times \{c_0\}$$

$$T = \{a_0\} \times \{c_0, \ldots, c_m\} \cup \{a_0, \ldots, a_m\} \times \{c_0\}$$



Duke
UNIVERSITY

How fast is this approach?

$$R = \{a_0\} \times \{b_0, \ldots, b_m\} \cup \{a_0, \ldots, a_m\} \times \{b_0\}$$

$$S = \{b_0\} \times \{c_0, \ldots, c_m\} \cup \{b_0, \ldots, b_m\} \times \{c_0\}$$

$$T = \{a_0\} \times \{c_0, \ldots, c_m\} \cup \{a_0, \ldots, a_m\} \times \{c_0\}$$

- Each instance has 2m+1 rows.
- J(a, b, c) has 3m+1 rows
- Any pairwise join (e.g., J1(a,b,c) = R(a,b), S(b,c)) has size $m^2 + m$

Duke UNIVERSITY

What does this mean for triangle counting?

- Every database system necessarily takes $O(N^2)$
  - *Ignoring log terms*

- Find all pairs (b,c) are connected with a
- Check if (b,c) is an edge.

- Is this the best we can do?

Detour: Can Sampling Help Joins?

- Sample(Join(R,S)) ≠ Join(Sample(R), Sample(S))

$$R = \{(a, x_0)\} \cup \{b\} \times \{x_1, \ldots, x_n\}$$
$$S = \{(b, y_0)\} \cup \{a\} \times \{y_1, \ldots, y_n\}$$

- In R x S: Half the records have ‘a’ and half the records have ‘b’
- In Sample(R): probability ‘a’ appears is very small.

Back to triangle counting?

- Every database system necessarily takes $O(N^2)$
  - *Ignoring log terms*

- Find all pairs (b,c) are connected with a
- Check if (b,c) is an edge.

- Is this the best we can do?

We can do better!

- *… not only for triangle counting, but it seems database systems have been doing multi-way joins suboptimally for 40 years!!!*

- Triangle counting can be solved in $O(N^{1.5})$, and so can any join of the form R(a,b) S(b,c) T(a,c).

How?

- Is there an O(N) algorithm for the following join problem:

$$R = \{a_0\} \times \{b_0, \ldots, b_m\} \cup \{a_0, \ldots, a_m\} \times \{b_0\}$$

$$S = \{b_0\} \times \{c_0, \ldots, c_m\} \cup \{b_0, \ldots, b_m\} \times \{c_0\}$$

$$T = \{a_0\} \times \{c_0, \ldots, c_m\} \cup \{a_0, \ldots, a_m\} \times \{c_0\}$$

Duke UNIVERSITY

Power of Two Choices: Heavy vs Light

- Consider attribute A

- For all ai not equal to a0, there is exactly one tuple in R (ai, b0) and one tuple in T (ai, c0)

Compute $\sigma_{A=a_i}(R) \bowtie \sigma_{A=a_i}(T)$ and filter the results by probing against S

- The above strategy is bad for a0
  - Joining tables R and T on a0 results in an intermediate of N^2.

Duke UNIVERSITY

Power of Two Choices: Heavy vs Light

- Consider attribute A

- For all ai not equal to a0,
  and one tuple in T (ai, c0)

There are O(N) values ai, each resulting in a single join record (ai, b0, c0). Checking whether (b0, c0) is in S is O(1) … *assuming an index*

Compute $\sigma_{A=a_i}(R) \bowtie \sigma_{A=a_i}(T)$ and filter the results by probing against S

- For ai = a0:

Consider each tuple in $(b, c) \in S$ and check if $(a_i, b) \in R$ and $(a_i, c) \in T$.

There are N rows in S. Again, checking (ai, b) is in R and (ai, c) is in T takes O(1) …
assuming an index

Power of Two Choices: Heavy vs Light

- Consider attribute A

Such ai's are called *light* nodes. Traditional join processing works here.

- For all ai not equal to a0,
  and one tuple in T (ai, c0)

Compute $\sigma_{A=a_i}(R) \bowtie \sigma_{A=a_i}(T)$ and filter the results by probing against S

- For ai = a0:

Consider each tuple in $(b, c) \in S$ and check if $(a_i, b) \in R$ and $(a_i, c) \in T$.

Such ai's are called *heavy* nodes. Need to compute the join jointly.

Power of Two Choices Algorithm

Algorithm 1 Computing $Q_{\triangle}$ with power of two choices.

Input: $R(A,B), S(B,C), T(A,C)$ in sorted order

$Q_{\triangle} \leftarrow \emptyset$
$L \leftarrow \pi_A(R) \cap \pi_A(T)$
For each $a \in L$ **do**
    If $|\sigma_{A=a}R| \cdot |\sigma_{A=a}T| \geqslant |S|$ **then**
        For each $(b,c) \in S$ **do**
            If $(a,b) \in R$ and $(a,c) \in T$ **then**
                Add (a,b,c) to $Q_{\triangle}$
    else
        For each $b \in \pi_B(\sigma_{A=a}R) \wedge c \in \pi_C(\sigma_{A=a}T)$
do
            If $(b,c) \in S$ **then**
                Add (a,b,c) to $Q_{\triangle}$
Return Q

Heavy value

Light value

Runtime Analysis

- Computing L takes:

$$\min\left(|\sigma_{A=a}R| \cdot |\sigma_{A=a}T|, |S|\right)$$

- Rest of the algorithm takes:

$$\sum_{a \in L} \min\left(|\sigma_{A=a}R| \cdot |\sigma_{A=a}T|, |S|\right) \leqslant \sqrt{|S|} \cdot \sqrt{|R|} \cdot \sqrt{|T|}$$

Can we do better?

- NO!

- A matching lower bound by Atserias Grohe and Marx (or the AGM bound)

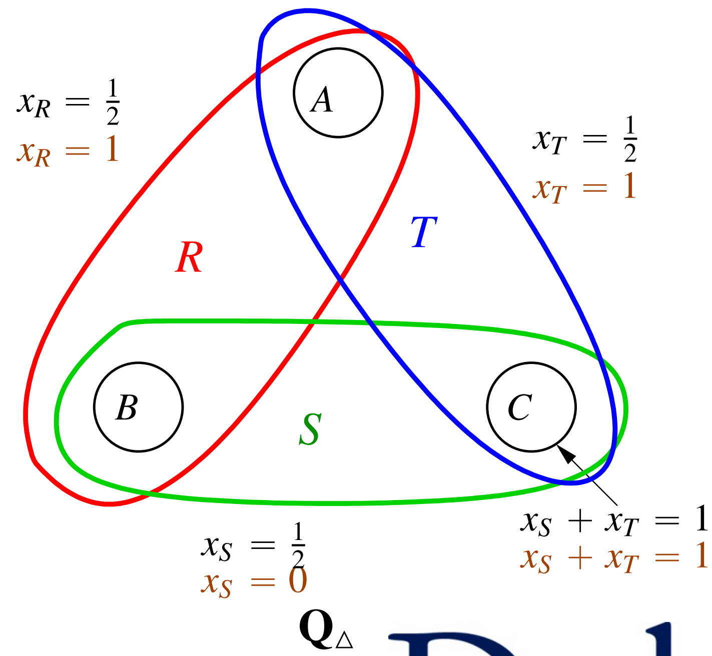
AGM Bound

- Let V denote the set of relations
- Every relation is a subset of attributes F (or a hyper edge)

- Let x be a vector of weights associated with each relation (hyperedge)

- **Fractional Edge Cover**:

$$\left\{ \mathbf{x} \mid \sum_{F:v\in F} x_F \geqslant 1, \forall v \in \mathcal{V}, \mathbf{x} \geqslant \mathbf{0} \right\}$$



Duke UNIVERSITY

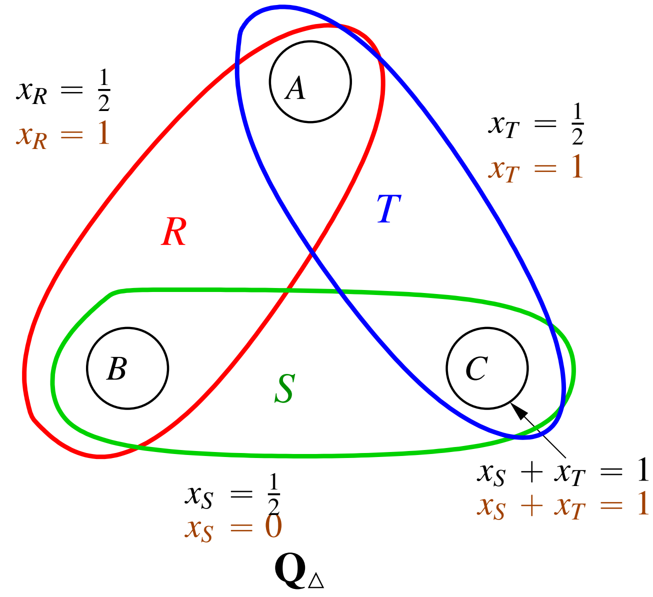
AGM Bound

$$|Q| = |\bowtie_{F \in \mathcal{E}} R_F| \leqslant \prod_{F \in \mathcal{E}} |R_F|^{x_F}$$

Duke UNIVERSITY

Examples

- Triples query
- Best fractional cover assigns weight 0.5 to each relation
- Join size is at most $(|R|.\ |S|.\ |T|)^{0.5}$
- Another fractional cover assings
  0 to relation S and 1 each to R and T
- Join size is at most $|R|.|T|$



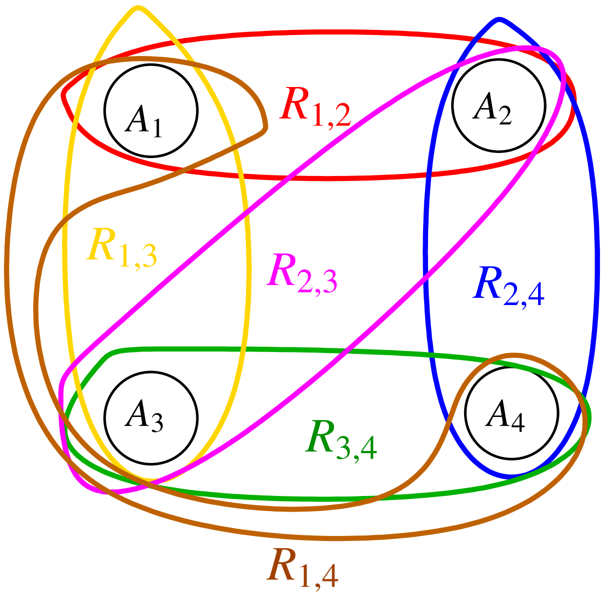
Duke UNIVERSITY

Examples

- J(a,b,c,d) :- R(a,b,) S(b,c) T(c,d) U(a,c) X(a,d) Y(b,d) Z(c,d)

- One cover is assigning weight of 1/(n-1) to all relations

- If all relations have size N,
  Join size is at most $N^{n/2}$



Duke UNIVERSITY

Tightest AGM Bound

- Answer to the following program

$$\begin{aligned} \min \quad & \sum_{F \in \mathcal{E}} (\log_2 |R_F|) \cdot x_F \\ \text{s.t.} \quad & \sum_{F : v \in F} x_F \geqslant 1, v \in \mathcal{V} \\ & \mathbf{x} \geqslant \mathbf{0} \end{aligned}$$

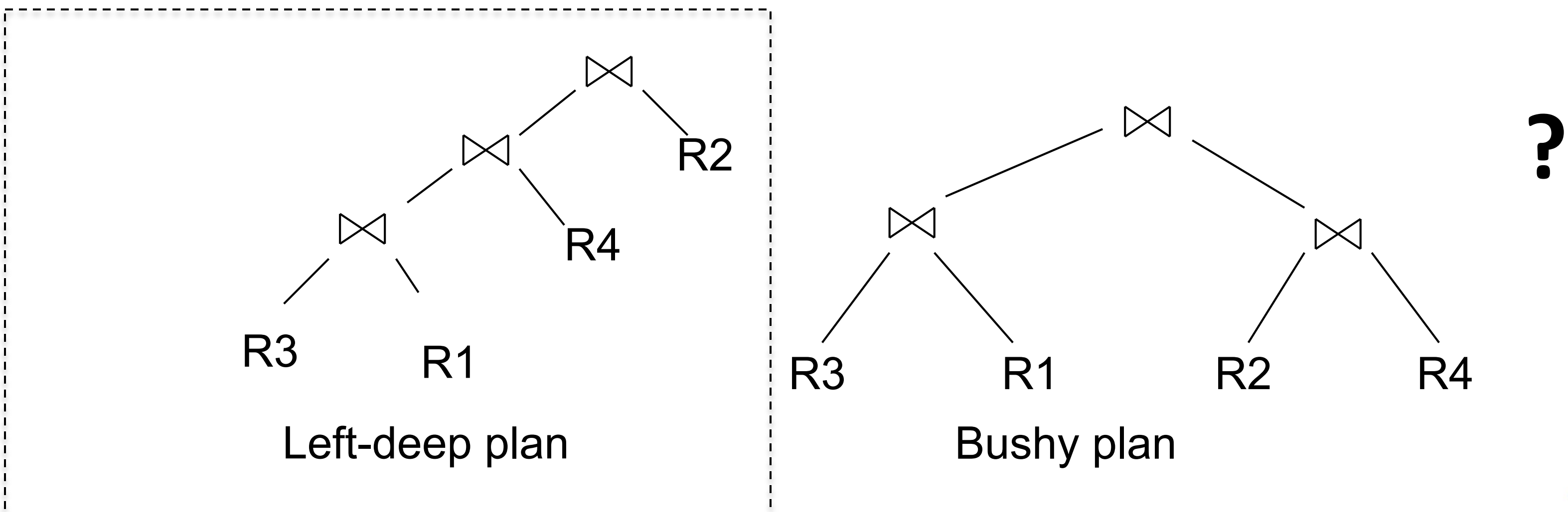
- Answer is called the *fractional edge cover number* $\rho^*(\mathcal{Q}, \mathcal{D})$

$$|\mathcal{Q}| \leqslant 2^{\rho^*(\mathcal{Q},\mathcal{D})}$$

Multi-way Joins *in Parallel Systems*

J(a,b,c) :- R(a,b) S(b,c) T(a,c)

- Historically databases designers decided that the best way to handle multi-way joins is to do them one pair at a time.
  - For efficiency reasons.



Summary

- We have been doing multiway joins wrong for 4 decades.
- Worstcase optimal joins work by carefully identifying skew in the data and using different algorithms depending on the skew of the tuple.
- Bushy multiway joins maybe useful in parallel settings.

Duke UNIVERSITY