CS848 Fall 2025: Algorithmic Aspects of Query Processing

Worst-case Optimal Joins

Xiao Hu

Sep 17, 2025

Agenda

- Last class: Traditional query processing
- This class: Worst-case optimal join algorithms
  - Limitations of Pairwise Framework
  - AGM bound
  - Worst-case Optimal Join Algorithms
  - Applications

Related Pointers

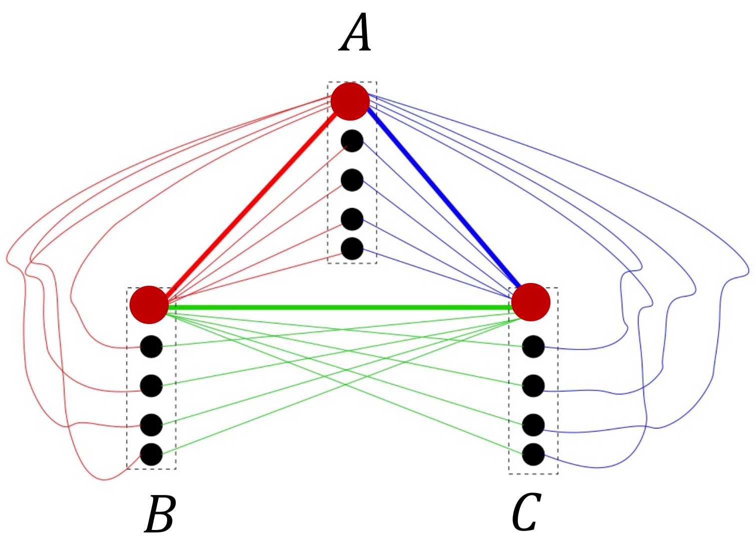
- Skew strikes back: New Developments in the Theory of Join Algorithms. SIGMOD Record 2013.
- A. ATSERIAS, M. GROHE and D. MARX, "Size bounds and query plans for relational joins," FOCS 2008.
- S. ABITEBOUL, R. HULL and V. VIANU, "Foundations of Databases."
- M. YANNAKAKIS, "Algorithms for acyclic database schemes," VLDB 1981.
- G. GOTTLOB, N. LEONE and F. SCARCELLO, "Hypertree Decompositions and Tractable Queries," Journal of Computer and System Sciences 64 (2002) .
- M. GROHE, T. SCHWENTICK and L. SEGOUFIN, "When is the evaluation of conjunctive queries tractable ?," STOC 2001 .
- G. GOTTLOB, G. GRECO and F. SCARCELLO, "Treewidth and Hypertree Width".

Algorithm 1: The Power of Two Choices

- Consider each value $a \in (\pi_A R) \cap (\pi_A T)$:

$$R(a, B) \bowtie S(B, C) \bowtie T(C, a)$$
$$\Leftrightarrow \left((\pi_B \sigma_{A=a} R) \times (\pi_C \sigma_{A=a} T)\right) \cap S$$

  - Choose the better choice of:
  - Choice 1: for each "neighbor" b, and for each "neighbor" c, check if $(b, c) \in S$
  - Choice 2: for each $(b, c) \in S$, check if b is "neighbor" of a and c is "neighbor" of a



$$Q_\Delta := R(A, B) \bowtie S(B, C) \bowtie T(C, A)$$

Algorithm 2: The delay of Computation

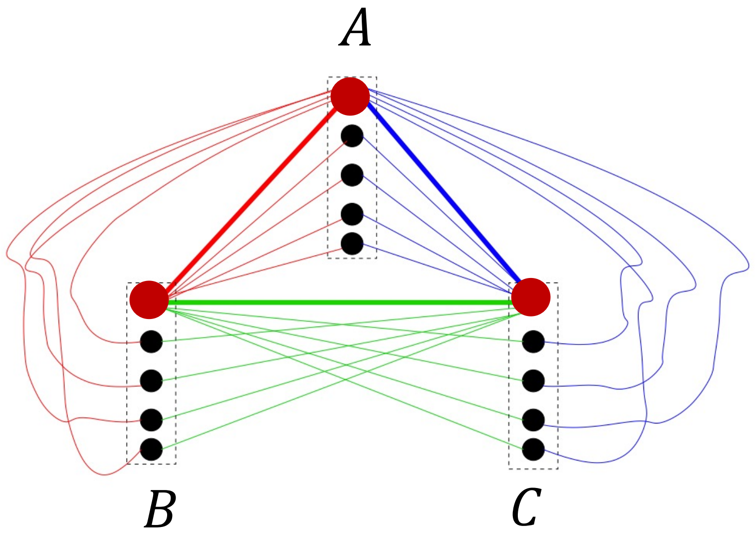
- Consider each value $a \in (\pi_A R) \cap (\pi_A T)$:

  $R(a, B) \bowtie S(B, C) \bowtie T(C, a)$

  - Consider each value $b \in (\pi_B \sigma_{A=a} R) \cap (\pi_B S)$

    $R(a, b) \bowtie S(b, C) \bowtie T(C, a)$

    - Consider each value $c \in (\pi_C \sigma_{B=b} S) \cap (\pi_C \sigma_{A=a} T)$, and output (a, b, c)

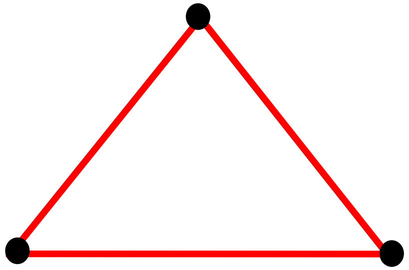


$Q_\Delta := R(A, B) \bowtie S(B, C) \bowtie T(C, A)$

AGM bound

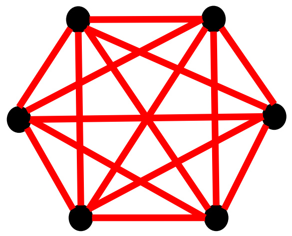
- For a join query Q, any database of input size N can produce at most $O(N^{\rho})$ join results
  - ρ: fractional edge covering number of join query

triangle



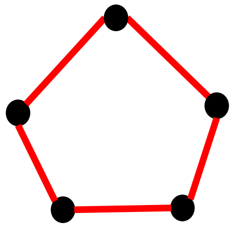
$\rho = 3/2$

k-clique



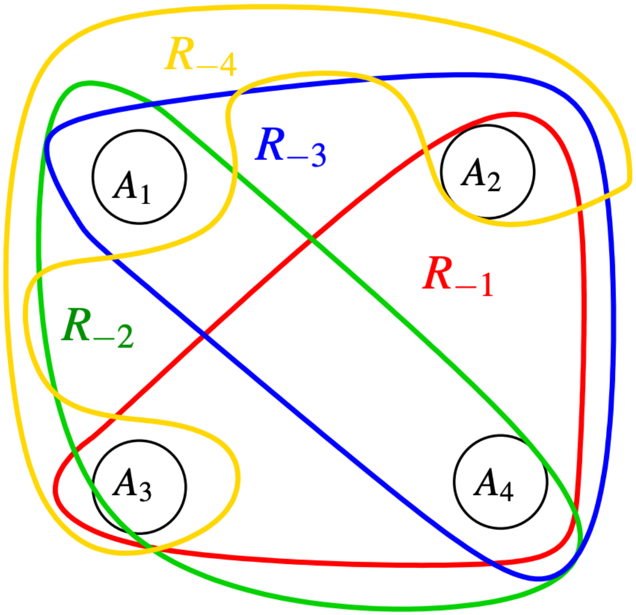
$\rho = k/2$

k-cycle



$\rho = k/2$

k- Loomis-Whitney



$\rho = k/(k-1)$

AGM bound

- For a join query $Q = (V, E)$, any database of input size $\{|R_e| : e \in E\}$

$$|Q| \leq \prod_{e \in E} |R_e|^{\rho_e}$$

where $\{\rho_e : e \in E\}$ is any fractional edge covering of Q.

- If all relations have the same size N,

$$|Q| \leq \prod_{e \in E} |R_e|^{\rho_e} = N^{\sum_e \rho_e}$$

where $\rho = \min \sum_e \rho_e$ is the fractional edge covering number of Q.

$$\min \sum_{e \in E} \rho_e$$
$$s.t. \sum_{e : v \in e} \rho_e \geq 1, \forall v \in V$$
$$\rho_e \geq 0, \forall e \in E$$

Prove AGM bound ---- Entropy

- Recall that the Shannon entropy of a random variable X that has n outcomes with probabilities $p_1, p_2, \cdots, p_n$ is defined as

$$H(X) = -\sum_{i=1}^{n} p_i \cdot \log p_i$$

- Let $t_1, t_2, \cdots, t_n$ be the join result of Q on any instance of input size N. For each t_i, we define a random variable X_i such that $X = (X_1, X_2, \cdots, X_n)$ has a uniform distribution over the join results of Q

$$H(X) = \log |Q|$$

Prove AGM bound ---- Entropy

- For each relation R_e, let Y_e be the marginal distribution of X onto e

$$H(Y_e) \leq \log |R_e|$$

- Shearer's lemma: for every fractional edge covering $\{\rho_e : e \in E\}$

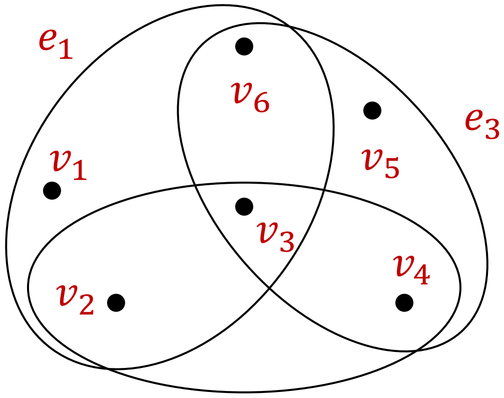
$$H(X) \leq \sum_e \rho(e) \cdot H(Y_e)$$

- Putting everything together:

$$\log |Q| = H(X) \leq \sum_e \rho(e) \cdot H(Y_e) \leq \sum_e \rho(e) \cdot \log |R_e|$$

Notations for join query (V, E)

- $V = \{v_1, v_2, v_3, v_4, v_5, v_6\}$
- $E = \{e_1, e_2, e_3\}$
  - $e_1 = \{v_1, v_2, v_3, v_6\}$
  - $e_2 = \{v_2, v_3, v_4\}$
  - $e_3 = \{v_3, v_4, v_5, v_6\}$



- For a subset of attributes $I \subseteq V$

$$E_I = \{e: e \cap I \neq \emptyset\}$$

i.e., the set of relations that have non-empty intersection of I

$$E_{\{v_1,v_2\}} = \{e_1, e_2\}$$
$$E_{\{v_3\}} = \{e_1, e_2, e_3\}$$
$$E_{\{v_5\}} = \{e_3\}$$

Worst-case Optimal Joins

GenericJoin ($\bowtie_{e \in E} R_e$):

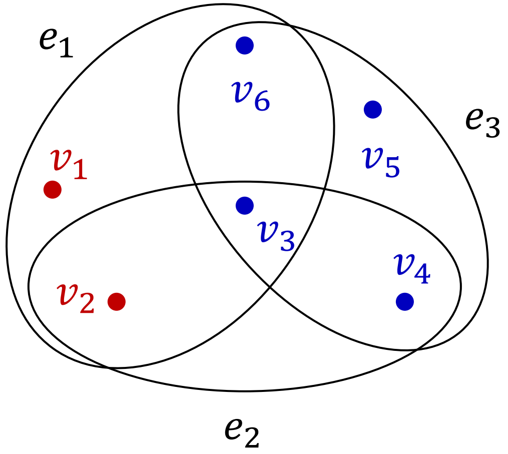
// suppose V is the set of all attributes, E is the set of all relations

$I = \{v_1, v_2\}$

$J = \{v_3, v_4, v_5, v_6\}$

- If $|V| = 1$: Compute the intersection $\cap_{e \in E} R_e$
- Let I and J be the partition of V
- $Q_I \leftarrow$ GenericJoin ($\bowtie_{e \in E_I} \pi_I R_e$)

  // Recursively compute the sub-join induced by attributes I



$Q_I \leftarrow (\pi_{v_1,v_2} R_1) \bowtie (\pi_{v_2} R_2)$

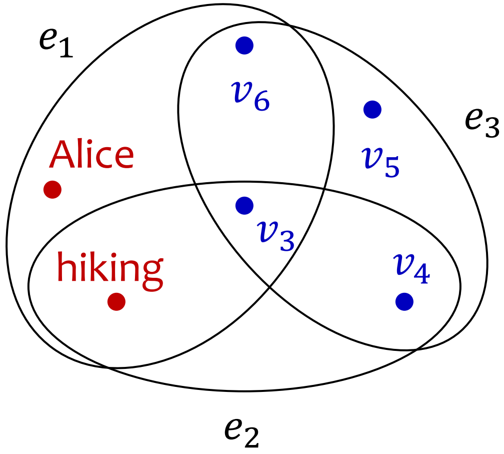
Worst-case Optimal Joins

GenericJoin $(\bowtie_{e\in E} R_e)$:

// suppose V is the set of all attributes, E is the set of all relations

$I = \{v_1, v_2\}$

$J = \{v_3, v_4, v_5, v_6\}$



- If $|V| = 1$: Compute the intersection $\cap_{e\in E} R_e$
- Let I and J be the partition of V
- $Q_I \leftarrow$ GenericJoin $(\bowtie_{e\in E_I} \pi_I R_e)$

  // Recursively compute the sub-join induced by attributes I
- For each tuple $t \in Q_I$:
  - $Q_t \leftarrow$ GenericJoin $(\bowtie_{e\in E_J} \pi_J(R_e \ltimes \boldsymbol{t}))$

    // Recursively compute all the join results participated by t
  - Output $\{t\} \times Q_t$

$Q_I \leftarrow (\pi_{v_1,v_2} R_1) \bowtie (\pi_{v_2} R_2)$

$Q_t \leftarrow (\pi_{v_3,v_6} \sigma_{v_1=\text{Alice} \cap v_2=hiking} R_1)$

$\bowtie R_2 \bowtie (\pi_{v_3,v_4} \sigma_{v_2=hiking} R_2)$

Worst-case Optimal Joins

GenericJoin ($\bowtie_{e \in E} R_e$):

// suppose V is the set of all attributes, E is the set of all relations

- **If $|V| = 1$: Compute the intersection $\cap_{e \in E} R_e$**
- Let I and J be the partition of V
- $Q_I \leftarrow$ GenericJoin ($\bowtie_{e \in E_I} \pi_I R_e$)

  // Recursively compute the sub-join induced by attr
- For each tuple $t \in Q_I$:
  - $Q_t \leftarrow$ GenericJoin ($\bowtie_{e \in E_J} \pi_J (R_e \ltimes \boldsymbol{t})$)

  // Recursively compute all the join results participated by t
  - Output $\{t\} \times Q_t$

If there exists $e^* \in E$ such that $e^* = V$, pick one of the following choices:

Choice 1:

- $Q' \leftarrow$ GenericJoin ($\bowtie_{e \in E - \{e^*\}} R_e$)
- For each tuple $t \in Q'$, check whether $t \in R_{e^*}$

Choice 2:

- For each tuple $t \in R_{e^*}$, check whether $\pi_e t \in R_e$ for all $e \in E$

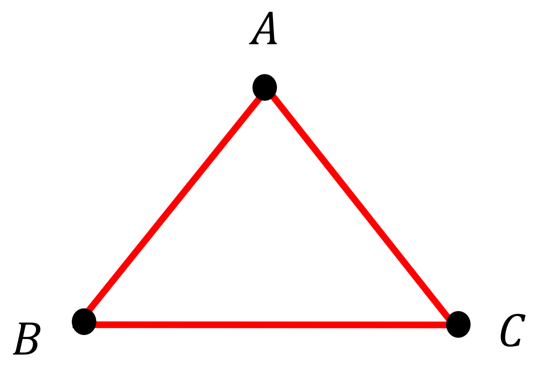
Worst-case Optimal Joins – Triangle Join Revisited

GenericJoin for $R(A,B) \bowtie S(B,C) \bowtie T(A,C)$:

- ~~If $|V| = 1$: Compute the intersection $\cap_{e \in E} R_e$~~
- Let $I = \{A\}$ and $J = \{B, C\}$ be the partition
- $Q_I \leftarrow$ GenericJoin $\left((\pi_A R) \bowtie (\pi_A T)\right)$

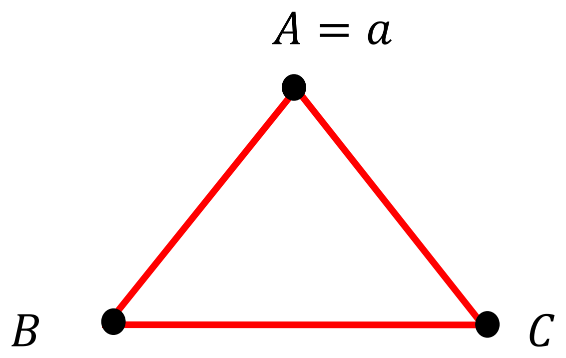
  ~~// Recursively compute the sub-join induced by attributes I~~
- For each value $a \in (\pi_A R) \bowtie (\pi_A T)$:
  - $Q_a \leftarrow$ GenericJoin $\left((\pi_B \sigma_{A=a} R) \bowtie (\pi_C \sigma_{A=a} T) \bowtie S\right)$

    // Recursively compute all the join results participated by a
  - Output $\{a\} \times Q_a$



Worst-case Optimal Joins – Triangle Join Revisited

GenericJoin for $(\pi_B \sigma_{A=a} R) \bowtie (\pi_C \sigma_{A=a} T) \bowtie S$:

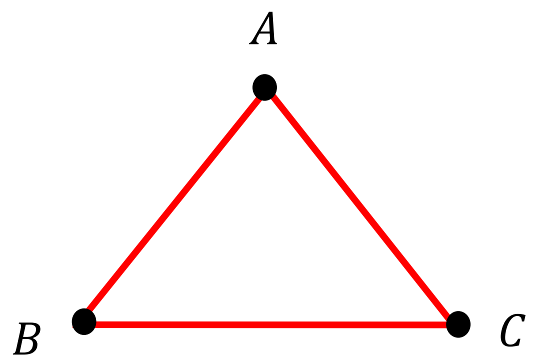


- $S \leftarrow$ the relation containing B, C
- Pick one of the two choices:
- Choice 1:
  - $Q_I \leftarrow$ GenericJoin $\left((\pi_B \sigma_{A=a} R) \bowtie (\pi_C \sigma_{A=a} T)\right)$
  - For each tuple $(b, c) \in Q_I$, whether $(b, c) \in S$
- Choice 2 :
  - For each tuple $(b, c) \in S$, check if $b \in \pi_B \sigma_{A=a} R$ and $c \in \pi_C \sigma_{A=a} T$

Algorithm 1: The power of two choices
For each $a \in (\pi_A R) \cap (\pi_A T)$:
  If $|\pi_B \sigma_{A=a} R| \cdot |\pi_C \sigma_{A=a} T| \leq |S|$: choice 1
  Else: choice 2

Worst-case Optimal Joins – Triangle Join Revisited

GenericJoin for $R(A,B) \bowtie S(B,C) \bowtie T(A,C)$:



- ~~If $|V| = 1$: Compute the intersection $\cap_{e \in E} R_e$~~
- Let $I = \{A,B\}$ and $J = \{C\}$ be the partition
- $Q_I \leftarrow$ GenericJoin $\left(R \bowtie (\pi_B S) \bowtie (\pi_A T)\right)$

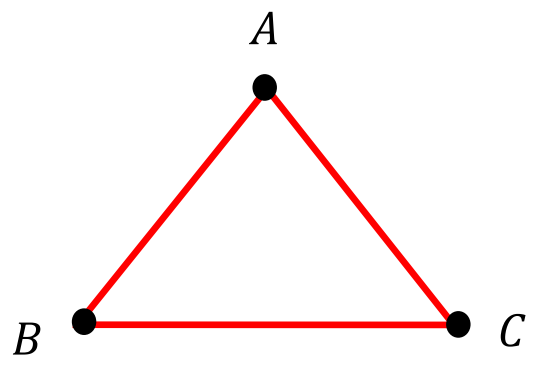
  ~~// Recursively compute the sub-join induced by attributes I~~
- For each tuple $(a,b) \in Q_I$:
  - $Q_{ab} \leftarrow$ GenericJoin $\left((\pi_C \sigma_{A=a} T) \bowtie (\pi_C \sigma_{B=b} S)\right)$

    // Recursively compute all the join results participated by (a,b)
  - Output $\{a,b\} \times Q_{ab}$

Worst-case Optimal Joins – Triangle Join Revisited

GenericJoin for $R \bowtie (\pi_B S) \bowtie (\pi_A T)$:

- ~~If $|V| = 1$: Compute the intersection $\cap_{e \in E} R_e$~~
- Let $I = \{A\}$ and $J = \{B\}$ be the partition
- $Q_I \leftarrow$ GenericJoin $((\pi_A R) \bowtie (\pi_A T))$
  ~~// Recursively compute the sub-join induced by attributes I~~
- For each tuple $a \in Q_I$:
  - $Q_a \leftarrow$ GenericJoin $((\pi_B \sigma_{A=a} R) \bowtie (\pi_B S))$
    // Recursively compute all the join results participated by (a)
  - Output $\{a\} \times Q_a$



Algorithm 2: The delay of Computation
For each $a \in (\pi_A R) \cap (\pi_A T)$:
  For each value $b \in (\pi_B \sigma_{A=a} R) \cap (\pi_B S)$
    For each value $c \in (\pi_C \sigma_{B=b} S) \cap (\pi_C \sigma_{A=a} T)$
      Output (a, b, c)

Worst-case Optimal Join Algorithm

- Query Decomposition Lemma

$$\sum_{t \in Q_I} \prod_{e \in E_J} |R_e \ltimes t|^{\rho_e} \le \prod_{e \in E} |R_e|^{\rho_e}$$

  - where (I, J) is the partition of V and $Q_I \coloneqq \Join_{e \in E_I} \pi_I R_e$

Worst-case Optimal Joins – Complexity

- For any fractional edge covering ρ of Q, GenericJoin(Q) can compute Q within $O(\prod_{e\in E}|R_e|^{\rho_e})$ time.
  - Base case: if $|V| = 1$, computing $\cap_{e\in E} R_e$ takes

$$\min_e |R_e| \leq \prod_{e\in E} |R_e|^{\rho_e}$$

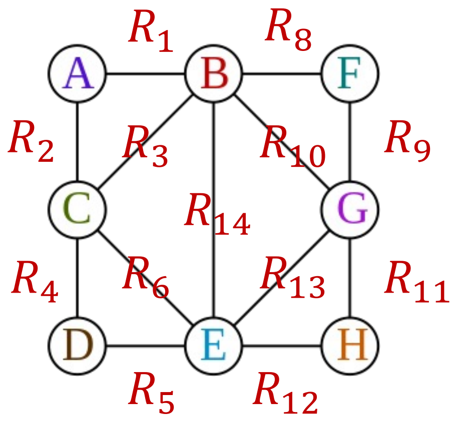
    where $\sum_e \rho_e \geq 1$ for covering the only attribute in V.
  - By hypothesis, computing Q_I takes $\prod_{e\in E_I} |\pi_I R_e|^{\rho_e} \leq \prod_{e\in E} |R_e|^{\rho_e}$ time.
  - By hypothesis, computing Q_t takes $\prod_{e\in E_J}|R_e \ltimes t|^{\rho_e}$
  - General case (implied by the query decomposition lemma):

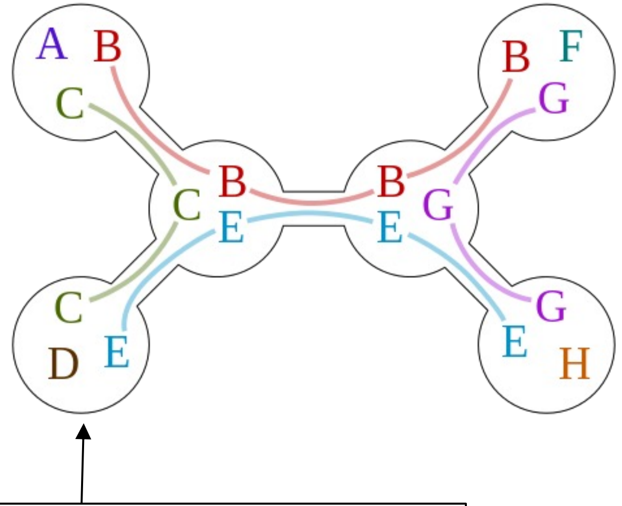
$$\sum_{t\in Q_I} \prod_{e\in E_J} |R_e \ltimes t|^{\rho_e} \leq \prod_{e\in E} |R_e|^{\rho_e}$$

ρ is also a fractional edge covering of (I, E_I) and (J, E_J)

Generalized Hypertree Decomposition (GHD)

- For a join query $Q = (V, E)$, a generalized hypertree decomposition for Q is a tree T with the set of nodes V_T and a mapping $\lambda: V_T \to 2^V$ such that
  - (coverage) for each relation $e \in E$, there exists a node $u \in T$ with e $\subseteq \lambda_u$
  - (connectness) for each attribute $x \in V$, the set of nodes containing x, i.e., $\{u \in V_T: x \in \lambda_u\}$ forms a connected subtree of T
- The sub-join query induced by node u is
  $Q_u = (\lambda_u, \{u \in V_T: e \cap u \neq \emptyset\})$.





$R_4 \bowtie R_5 \bowtie R_6 \bowtie (\pi_C R_2) \bowtie (\pi_C R_3)$
$\bowtie (\pi_E R_{12}) \bowtie (\pi_E R_{13}) \bowtie (\pi_E R_{14})$

Generalized Hypertree Decomposition (GHD)

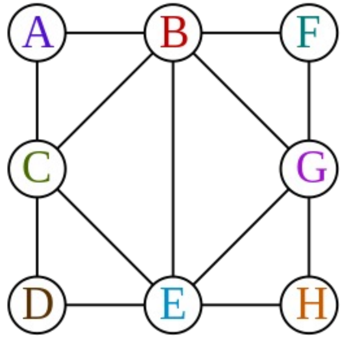
- The fractional hypertree width of Q is defined as

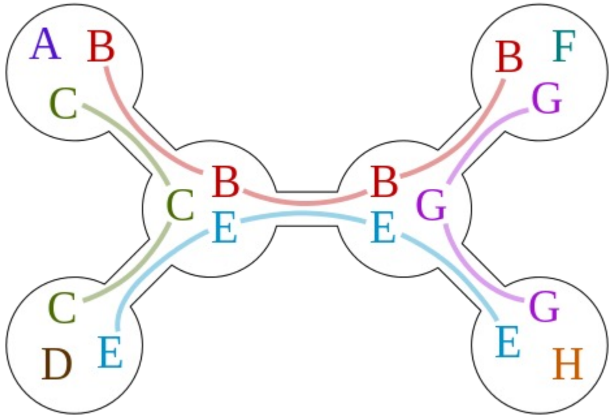
$$w(Q) = \min_{(T,\lambda) \text{ is a GHD for } Q} \max_{u \in T} \rho(Q_u)$$

  - $\rho(\cdot)$ is the optimal fractional edge covering number

- Algorithm for a GHD(T, λ):
  - Step 1: Compute the join results for each node $u \in V_T$ using WCOJ algorithm and materialize it as a table
  - Step 2: Invoke the Yannakakis algorithm on T

- Total complexity is $O(N^w + OUT)$
  - The time complexity of step 1 is $O\left(N^{\max_{u \in T} \rho(Q_u)}\right)$
  - the input size of T in step 2 is $O\left(N^{\max_{u \in T} \rho(Q_u)}\right)$





$w(Q) = 1.5$

Summary of Worst-case Optimal Joins

- For any join, the WCOJ algorithm can compute it in $O(N^{\rho})$ time
  - ρ is the fractional edge covering number

- For all joins, the WCOJ algorithm and Yannakakis algorithm together can compute it in $O(N^{w} + OUT)$ time
  - $w \leq \rho$ is the fractional hypertree width
  - $w = 1$ for acyclic joins