Massively Parallel Evaluation For Relational Queries

Paris Koutris

Xiao Hu

Simons Institute 2023

TALK OUTLINE

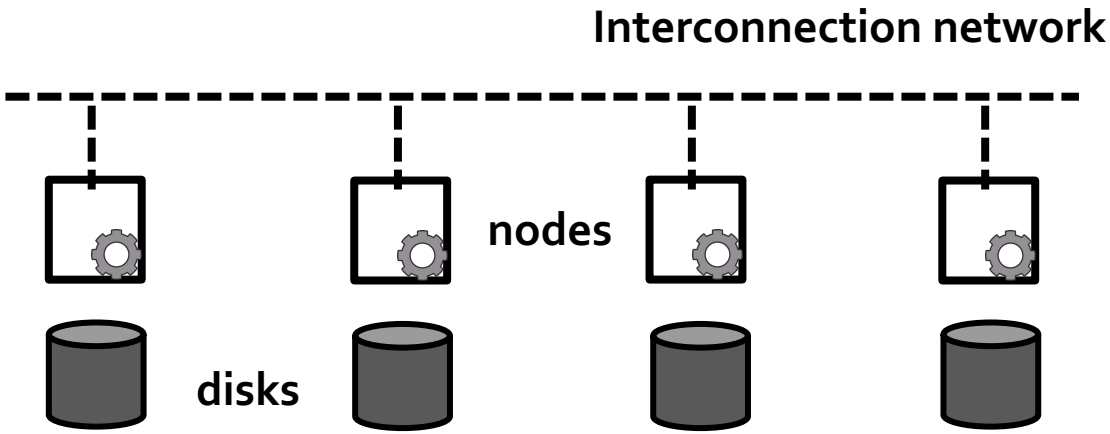
The MPC Model

- Joins: the skew-free case
- Joins: worst-case optimality
- Joins: output-sensitive algorithms
- Joins + Aggregations

SHARED-NOTHING ARCHITECTURE

Shared-nothing architectures are prevalent in large-scale systems

DeWitt and Gray - *1992*

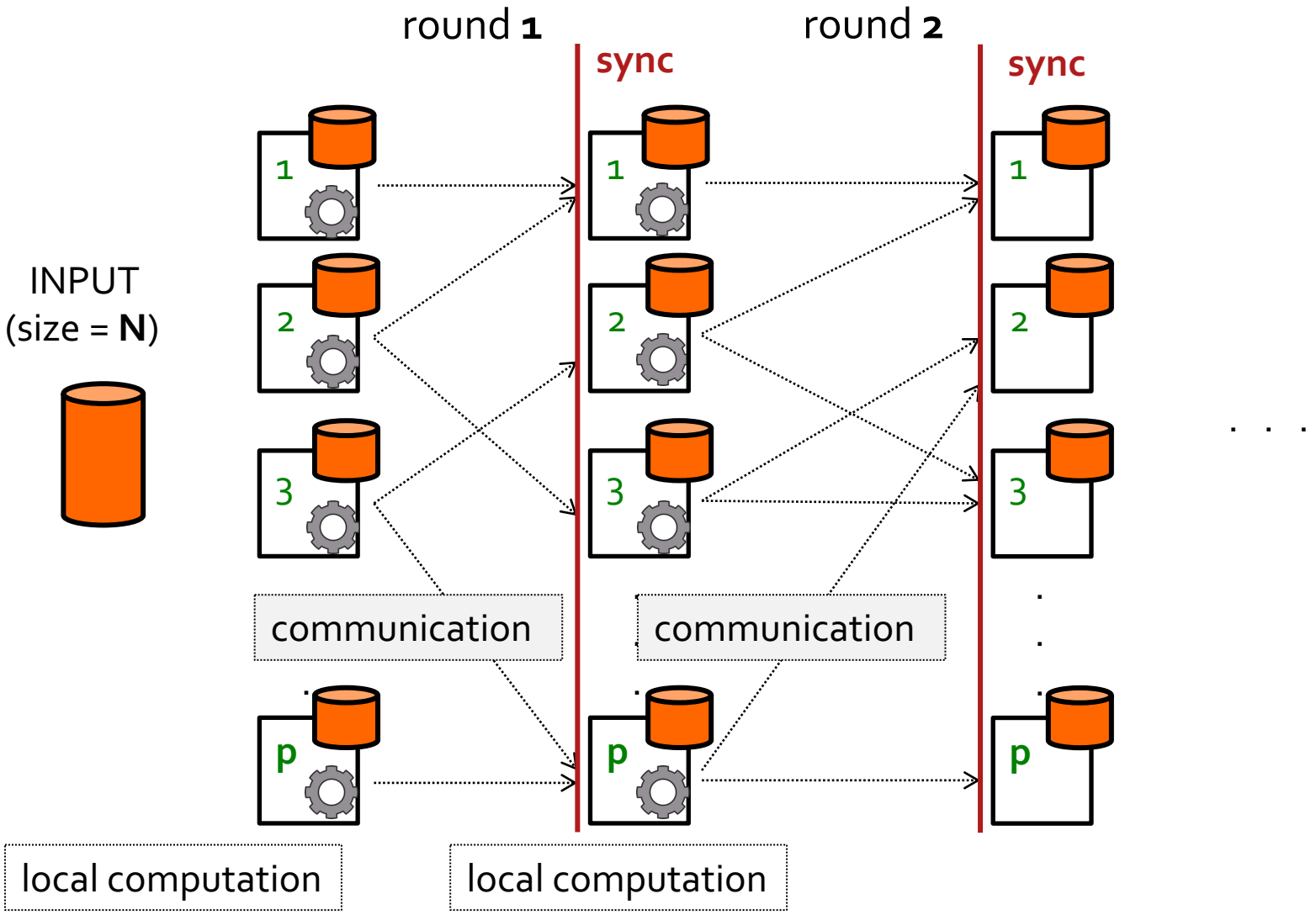


MODELING MASSIVE PARALLELISM

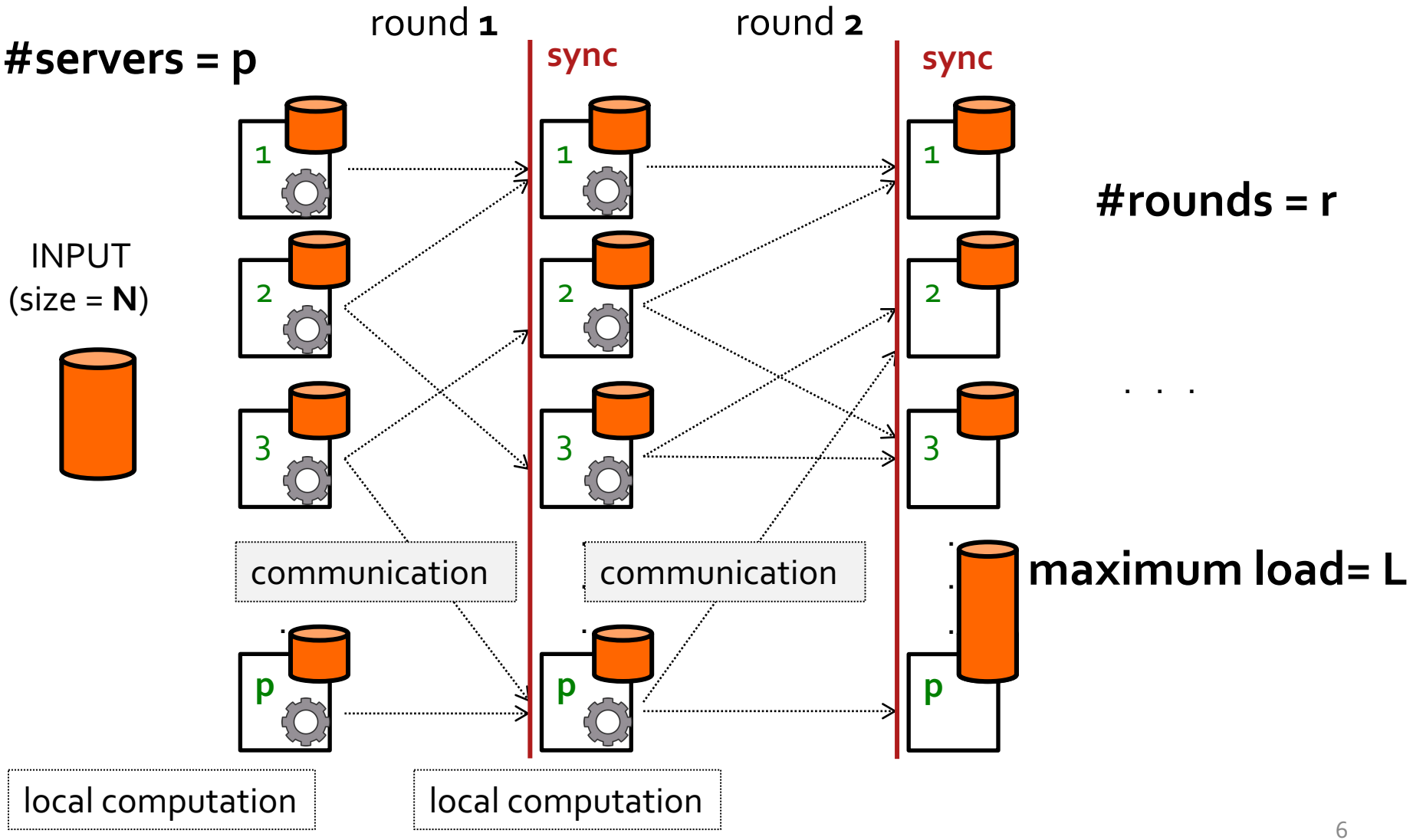
- **PRAM:** Parallel Random-Access Machine
- **BSP:** Bulk Synchronous Parallel Valiant - *1991*
- **LogP** Culler et al. - *1993*

- **MUD:** Massive, Unordered, Distributed Feldman et al. - *2010*
- **MRC** Model Karloff et al. – *2010*
- Map-Reduce model Afrati et al. - *2012*

MPC MODEL: COMPUTATION



MPC MODEL: PARAMETERS



MPC PARAMETERS

*#servers = **p***
*#rounds = **r***
*maximum load = **L***

$$\frac{N}{p} \leq L \leq N$$

The data is evenly distributed

All data to one location
no use of parallelism

What is the best load **L** for a given number of rounds **r** for processing a relational query?

EXAMPLE: SET INTERSECTION

*#servers = **p***
*#rounds = **r***
*maximum load = **L***

Compute the intersection of two relations with size N

q(x) = **R**(x), **S**(x)

Algorithm:

use a hash function $h : Dom \rightarrow \{1, 2, \ldots, p\}$

- Communication:
  - **R**(a) -> h(a)
  - **S**(a) -> h(a)
- Computation**:** compute the local intersection

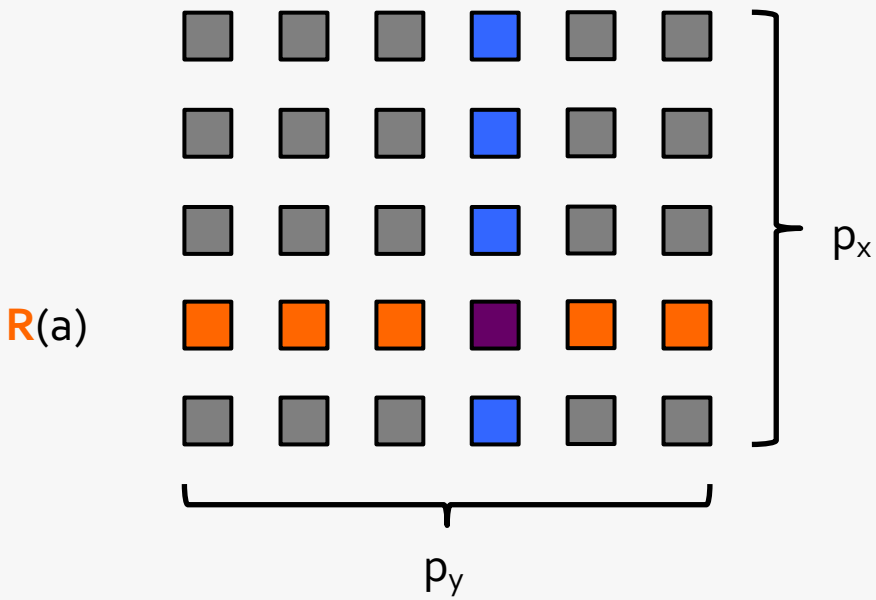
$$L = O(N/p)$$
$$r = 1$$

EXAMPLE: CARTESIAN PRODUCT

Compute the Cartesian product of two relations of size N

q(x, y) = **R**(x), **S**(y)

Algorithm:



$$p_x = p_y = \sqrt{p}$$

$$L = \frac{N}{p_x} + \frac{N}{p_y} = \frac{2N}{\sqrt{p}}$$

TALK OUTLINE

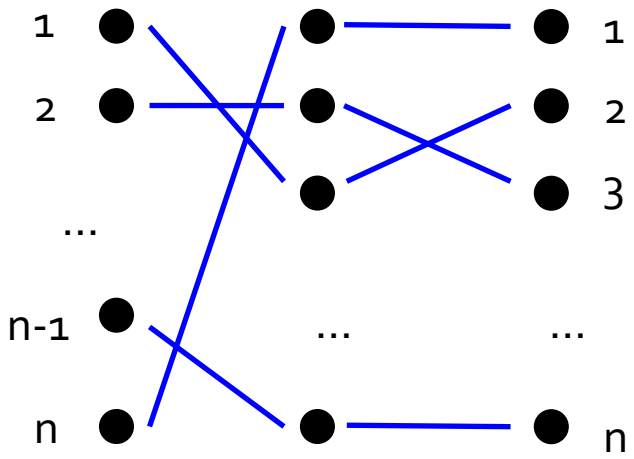
- The MPC Model

Joins: the skew-free case

- Joins: worst-case optimality
- Joins: output-sensitive algorithms
- Joins + Aggregations

MATCHING DATABASES

- Every relation $R(A_1, \ldots, A_k)$ contains exactly n tuples
- Every attribute A_i contains each value in {1,2, …, n} once
- A matching database has no skew



Relation R(X, Y, Z)

X	Y	Z
1	3	2
2	2	3
n-1	n	n
…	…	…

COMPUTING TRIANGLES

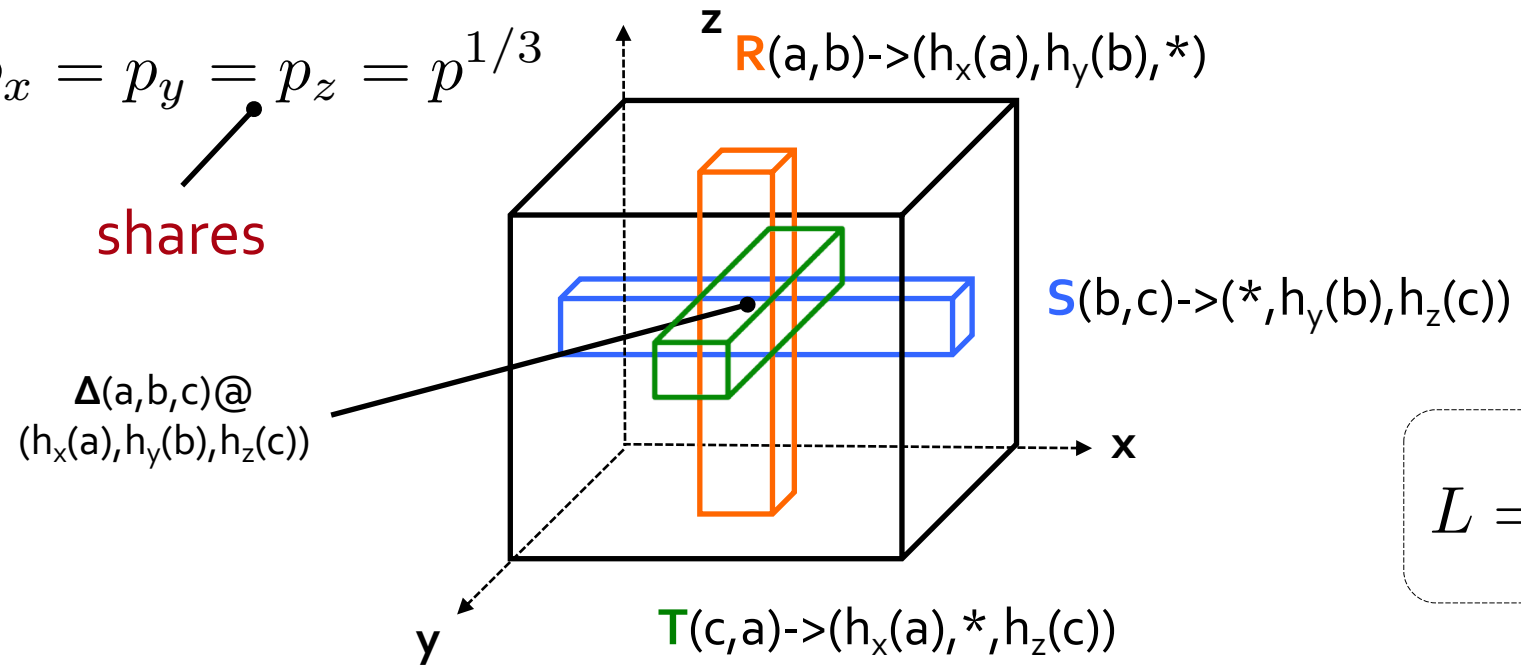
*#servers = **p***
*maximum load = **L***

Afrati and Ullman - *2010*

Δ(x, y, z) = **R**(x, y), **S**(y, z), **T**(z, x)

$$p_x = p_y = p_z = p^{1/3}$$

shares



$$L = \frac{3N}{p^{2/3}}$$

THE HYPERCUBE ALGORITHM

$\mathbf{q}(x_1, x_2, \ldots, x_k) = S_1(\ldots), S_2(\ldots), \ldots, S_t(\ldots)$ S_j has N_j tuples

Algorithm:

- organize the **p** servers in a hypercube:

  $[p] = [p_1] \times \cdots \times [p_k]$ ← shares

- send tuple $S_j(x_{i1}, x_{i2}, \ldots)$ to all servers whose coordinates i1, i2,… are $h_{i1}(x_{i1})$, $h_{i2}(x_{i2})$,… and broadcast along the missing dimensions
- compute **q** locally at each server

CHOOSING OPTIMAL SHARES (1)

q(x_1, x_2, …, x_k) = S_1(…), S_2(…), …, S_t(…)

- the shares must satisfy $\prod_i p_i \leq p$
- #tuples a server receives from S_j is $\frac{N_j}{\prod_{i:x_i \in S_j} p_i}$
- To optimize load, minimize

$$L = \max_j \frac{N_j}{\prod_{i:x_i \in S_j} p_i}$$

CHOOSING OPTIMAL SHARES (2)

q(x_1, x_2, …, x_k) = S_1(…), S_2(…), …, S_t(…)

The share $p_i = p^{e_i}$ is chosen according to the LP:

$$\begin{aligned}
&\text{minimize} \quad \lambda \\
&\text{subject to} \quad \sum_{i \in [k]} e_i \leq 1 \\
&\forall j = 1, \ldots, \ell : \sum_{i : x_i \in S_j} e_i + \lambda \geq \log_p(N_j) \\
&\forall i = 1, \ldots, k : e_i \geq 0, \quad \lambda \geq 0
\end{aligned}$$

MINIMIZING THE LOAD

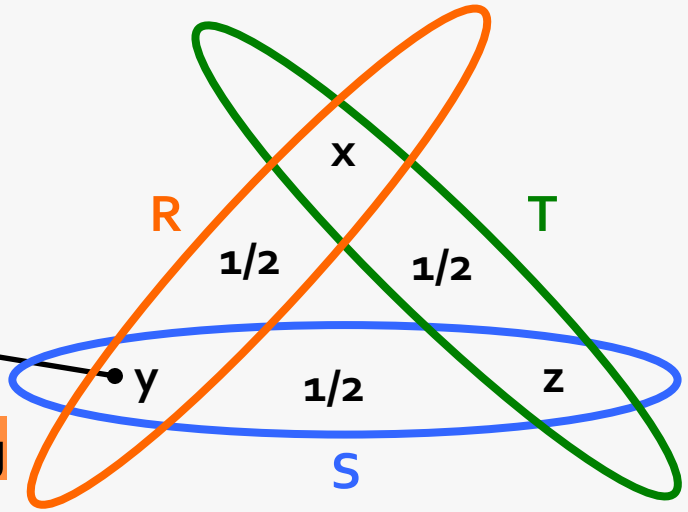
$$\text{maximize} \quad \left(\frac{\prod_{j=1}^{t} N_j^{u_j}}{p} \right)^{1/\sum_j u_j}$$

$N_1 = \ldots = N_t = N$

$$\frac{N}{p^{1/\sum_j u_j}}$$

the weights $(u_1, u_2, \ldots, u_t)$ form a fractional edge packing

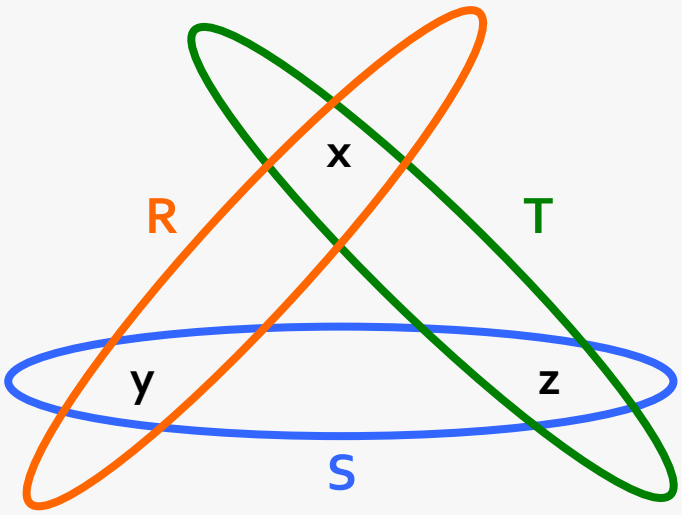
for each variable, the sum of edges is at most 1



$\tau^*(q)$ = minimum size of a fractional edge packing

PACKINGS FOR THE TRIANGLE

maximize $\left(\frac{\prod_{j=1}^{t} N_j^{u_j}}{p} \right)^{1/\sum_j u_j}$



edge packing	load
(1/2, 1/2, 1/2)	$(N_R N_S N_T)^{1/3}/p^{2/3}$
(1, 0, 0)	N_R/p
(0, 1, 0)	N_S/p
(0, 0, 1)	N_T/p

$$L = \max\left(\frac{(N_R N_S N_T)^{1/3}}{p^{2/3}}, \frac{N_R}{p}, \frac{N_S}{p}, \frac{N_T}{p} \right)$$

THEOREM #1 (UPPER BOUND)

*#servers = **p***
*maximum load = **L***

Beame, Koutris, Suciu - *2013*

The HyperCube/Shares (randomized) algorithm can compute any Conjunctive Query **Q** with one round and load

$$L = O\left(\frac{N}{p^{1/\tau^*(Q)}}\right)$$

- input a skew-free* database
- exponentially small probability of failure

*skew-free ~ the frequency of each value is at most N/p

THEOREM #2 (LOWER BOUND)

*#servers = **p***
*maximum load = **L***

Beame, Koutris, Suciu - *2013*

Any one-round randomized algorithm that computes a Conjunctive Query **Q** needs load

$$L = \Omega\left(\frac{N}{p^{1/\tau^*(Q)}}\right)$$

The lower bound uses as input matching databases.

MULTIPLE ROUNDS

Our results apply to a weaker model (tuple-based MPC):

- only join tuples can be sent in rounds > 1
- routing of each tuple **t** depends only on **t**

Beame, Koutris, Suciu - *2013*

For every tree-like query **Q**, any tuple-based MPC algorithm with r rounds has load

$$L = \Omega\left(\frac{N}{p^{\frac{1}{\lceil d(Q)^{1/r}/2 \rceil}}}\right)$$

This bound agrees with the upper bound within 1 round

d(Q) : *diameter of the query*

tree-like queries : *#variables + #atoms - Σ(arities) = 1*

PATH QUERY

$L_k(x_0, x_1, \ldots, x_k) = \mathbf{S_1}(x_0, x_1), \ldots, \mathbf{S_k}(x_{k-1}, x_k)$

can be computed in **r** rounds with load

$$L = \frac{N}{p^{\frac{1}{\lceil k^{1/r}/2 \rceil}}}$$

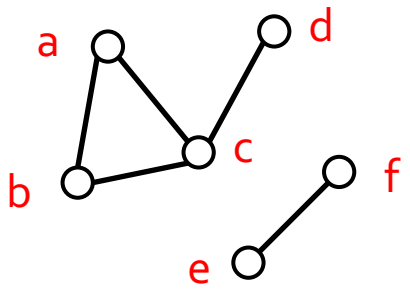
The algorithm is (almost) optimal

k=8

#rounds	load
1	$N/p^{1/4}$
2	$N/p^{1/2}$
3	N/p
4	N/p

CONNECTED COMPONENTS

Given a graph with N edges, compute all pairs of connected nodes (transitive closure)



$$(a,b),(a,c),(a,d),$$
$$(b,c),(b,d),(c,d),(e,f)$$

Any algorithm that computes the transitive closure with load L < N requires non-constant #rounds.

TALK OUTLINE

- The MPC Model
- Joins: the skew-free case

Joins: worst-case optimality

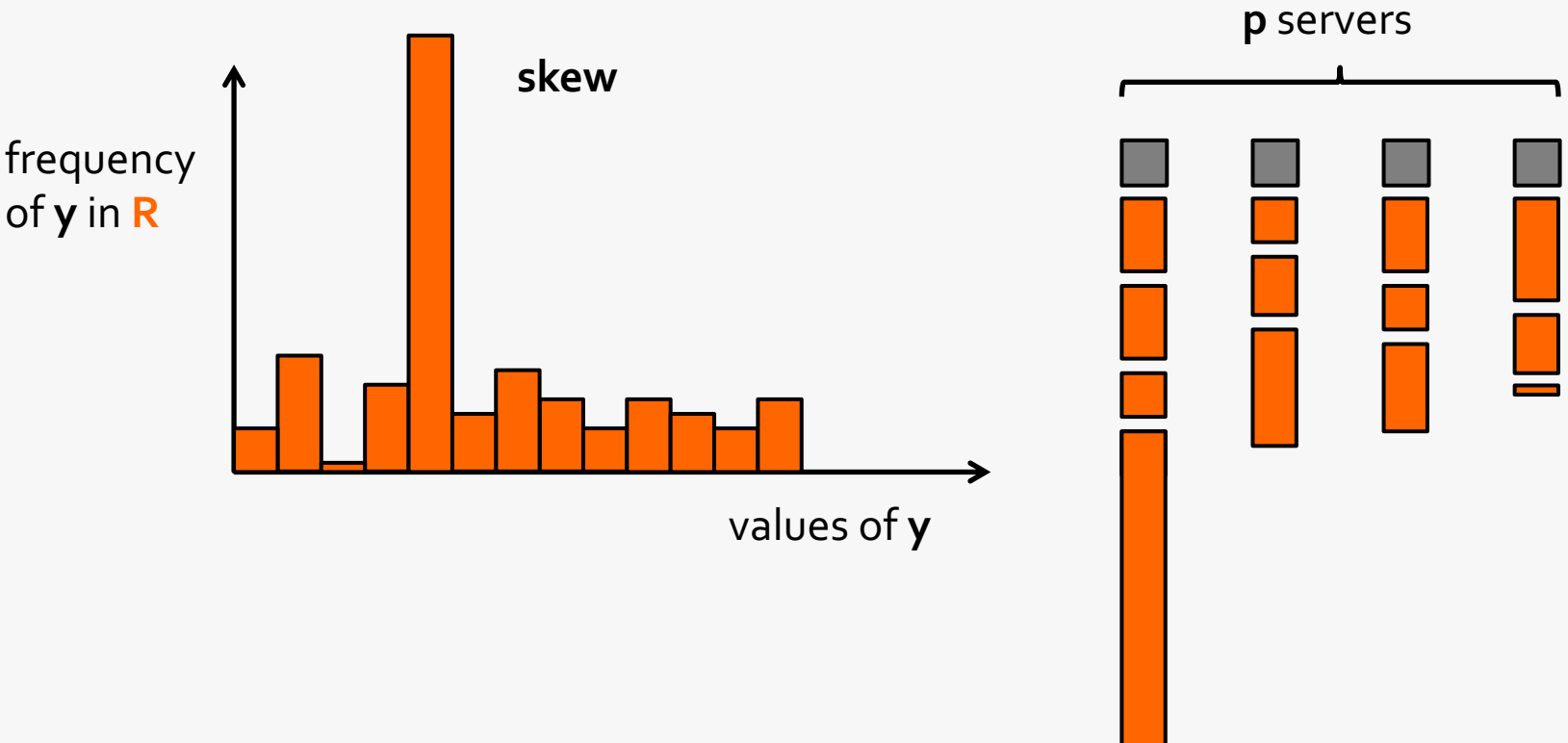
- Joins: output-sensitive algorithms
- Joins + Aggregations

DATA SKEW

A join between two relations of size N

q(x, y, z) = **R**(x,y), **S**(y,z)

HyperCube chooses shares $p_y = p$ (hash on **y**)



WORST-CASE OPTIMALITY (WCO)

the optimality of an algorithm is defined w.r.t. the worst-case input that satisfies the given statistics

1-ROUND WCO ALGORITHM

Heavy Hitter: a value *h* is a heavy hitter in *S* if the frequency of the value in *S* is at least N_S /p

Heavy tuple: a tuple t is heavy at variable x if the value t[x] is a heavy hitter in some relation (otherwise light)

Algorithm WCO-1

- For every subset **v** of variables, in parallel compute the join of the tuples that are heavy exactly at **v**
- To achieve this run HyperCube by assigning shares only to the the variables not in **v** (any variable in **v** has share 1)

EXAMPLE: TRIANGLE REVISITED

$$\Delta(x, y, z) = R(x, y), S(y, z), T(z, x)$$

v = {}

- all values are light
- residual query: $R(x, y), S(y, z), T(z, x)$
- optimal fractional edge packing: (½, ½, ½)
- the HyperCube algorithm achieves load $O(N/p^{2/3})$

EXAMPLE: TRIANGLE REVISITED

$$\Delta(x, y, z) = R(x, y), S(y, z), T(z, x)$$

$\mathbf{v} = \{x\}$

- the values x can take are heavy
- residual query: $R(y), S(y, z), T(z)$
- optimal fractional edge packing: (1,1)
- the HyperCube algorithm achieves load $O(N/p^{1/2})$

EXAMPLE: TRIANGLE REVISITED

$$\Delta(x, y, z) = R(x, y), S(y, z), T(z, x)$$

v = {x,y}

- the values x, y can take are heavy
- residual query: S(z), T(z)
- optimal fractional edge packing: (1,0)
- the HyperCube algorithm achieves load O(N/p)

FRACTIONAL EDGE QUASI-PACKING

fractional edge quasi-packing: a fractional edge packing for any residual query of Q

variables	residual query	quasi-packing	size
{}	R(x, y),S(y, z),T(z, x)	(½, ½, ½)	3/2
{x}	R(y), S(y, z), T(z)	(1,1)	2
{x,y}	S(z), T(z)	(1,0)	1
			ψ* = 2

ψ*(Q) = maximum fractional edge quasi-packing

1-ROUND WCO THEOREM

The WCO-1 algorithm is worst-case optimal for one round and achieves load

$$L = O\left(\frac{N}{p^{1/\psi^*(Q)}}\right)$$

when all relations have size equal to N

EDGE COVERS & PACKINGS

$$\psi^*(Q) \geq \max\{\rho^*(Q), \tau^*(Q)\}$$

fractional edge packing: assign a weight to each atom such that for each variable *x*, the sum of weights of the atoms that contain *x* is at most 1

$\tau^*(q)$ = maximum value of edge packing

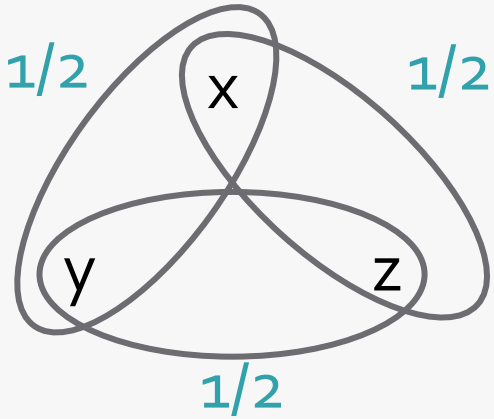
fractional edge cover: assign a weight to each atom such that for each variable *x*, the sum of weights of the atoms that contain *x* is at least 1

$\rho^*(q)$ = minimum value of edge cover

WCO In Multiple Rounds

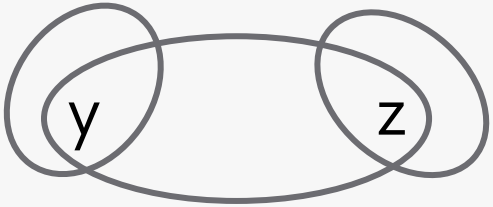
Observation for triangle join

- 1-round skew-free: $L = \frac{N}{p^{2/3}}$ **τ* = 3/2**
- 1-round skew: $L = \frac{N}{p^{1/2}}$ **ψ* = 2**
- 2-round skew: $L = \frac{N}{p^{2/3}}$ **ρ* = 3/2**



Magic of semi-joins: **R**(y), **S**(y, z), **T**(z)

- **S**(y, z) ⋉ **R**(y) ⋉ **T**(z)



WCO In Multiple Rounds

Sequential WCO Join

$O(N^{\rho^*})$
[NPRR12][V14]

$\Omega(N^{\rho^*})$
[AGM08]

MPC WCO $O(1)$-round Join

$$L = O\left(\frac{N}{p^{1/\rho^*}}\right)$$

Path, LW, Clique joins [KS16]
Graph joins [KS17][T20][KST22]
Acyclic joins [H21][T22]

$L = \Omega\left(\frac{N}{p^{1/\rho^*}}\right)$
for all joins

$L = \Omega\left(\frac{N}{p^{1/\tau^*}}\right)$ [H21]
for some cyclic joins

Multi-Round Primitives

Assume $N \geq p^{1+\epsilon}$ for some constant $\epsilon > 0$. All the following primitives can be computed in $O(1)$ rounds within $O\left(\frac{N}{p}\right)$ load.

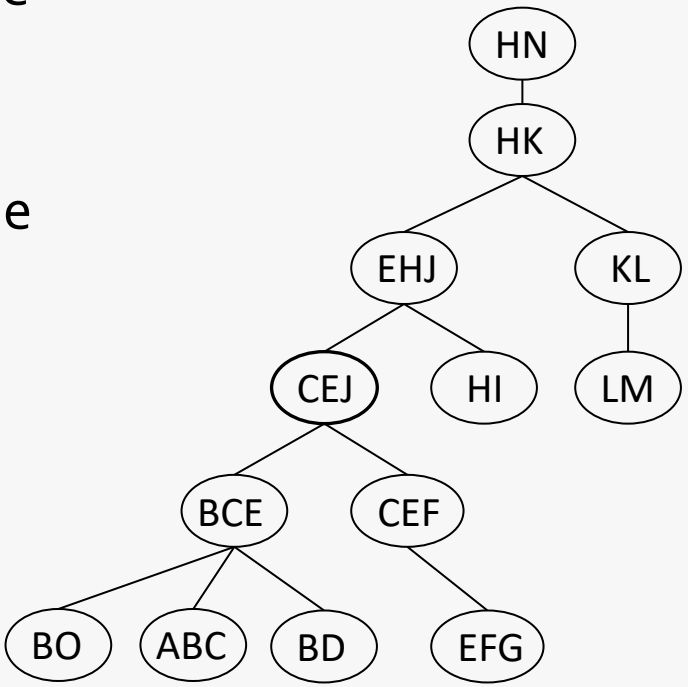
- Sorting [G99]
- Prefix-Sum [GSZ11]
- Multi-Search [GSZ11]
- Multi-numbering, Parallel packing, Reduce-by-key, Semi-joins, Server allocation etc. [HY17]

degrees

Reduce join: there exists no pair of relations e and e' such that e⊆e'

Acyclic Joins

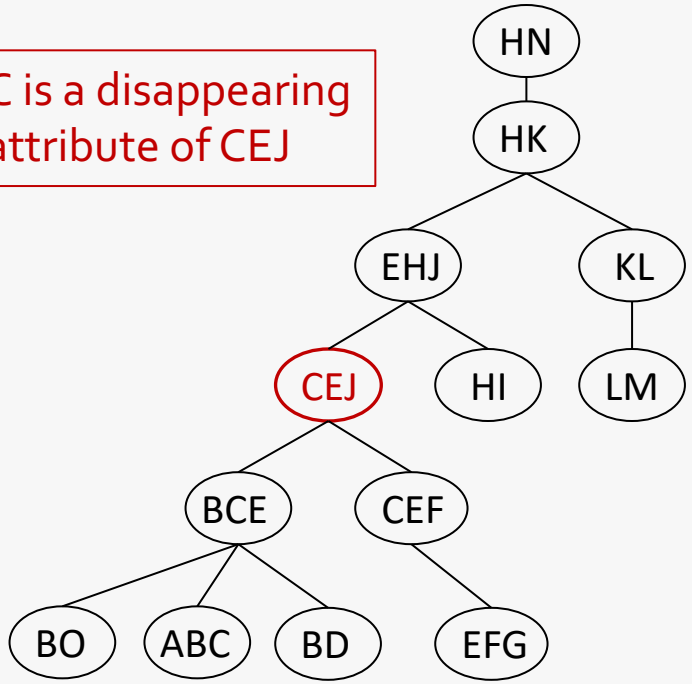
- Join Tree [F83]
  - There is an one-to-one correspondence between relations and nodes
  - For every attribute x, the set of nodes containing x forms a connected subtree
- Yannakakis algorithm [Y81]
  - Semi-joins
  - # intermediate join results $\leq$ OUT
- Challenges in MPC
  - Shuffling intermediate join results incurs a load of $O\left(\frac{OUT}{p}\right)$!



WCO Acyclic Joins

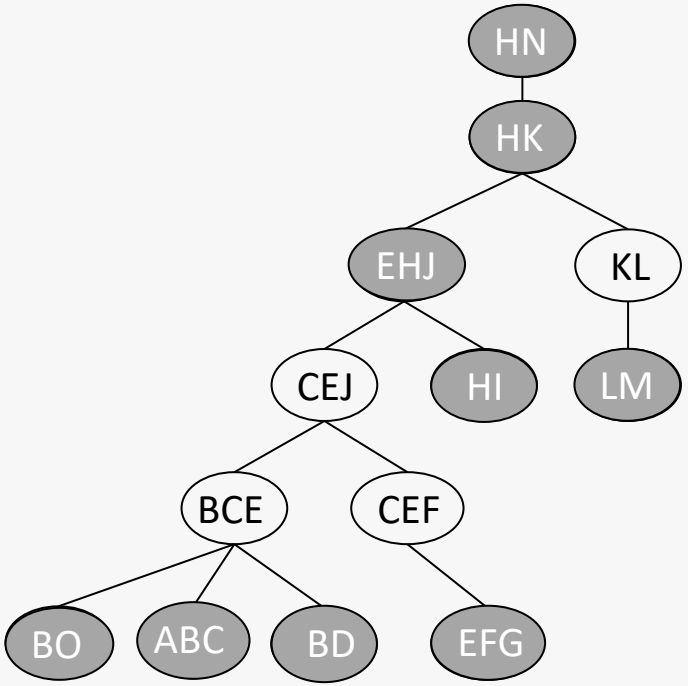
- Canonical edge cover
  - Aways peel off a leaf node
  - If there is a disappearing attribute not covered yet, add this node to the edge cover
  - Repeat until the join tree is Ø

C is a disappearing attribute of CEJ



WCO Acyclic Joins

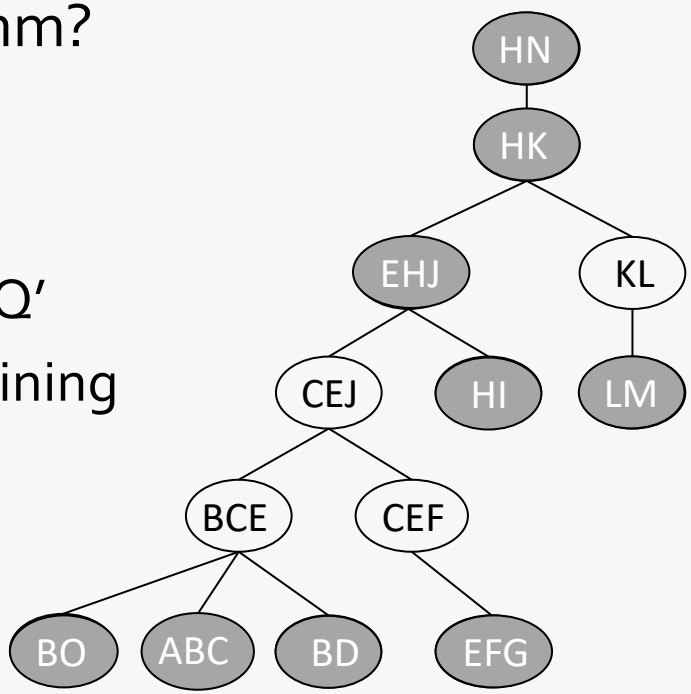
- Canonical edge cover (**CEC**)
  - Aways peel off a leaf node
  - If there is a disappearing attribute not covered yet, add this node to the edge cover
  - Repeat until the join tree is Ø
- **CEC** is optimal



WCO Acyclic Joins

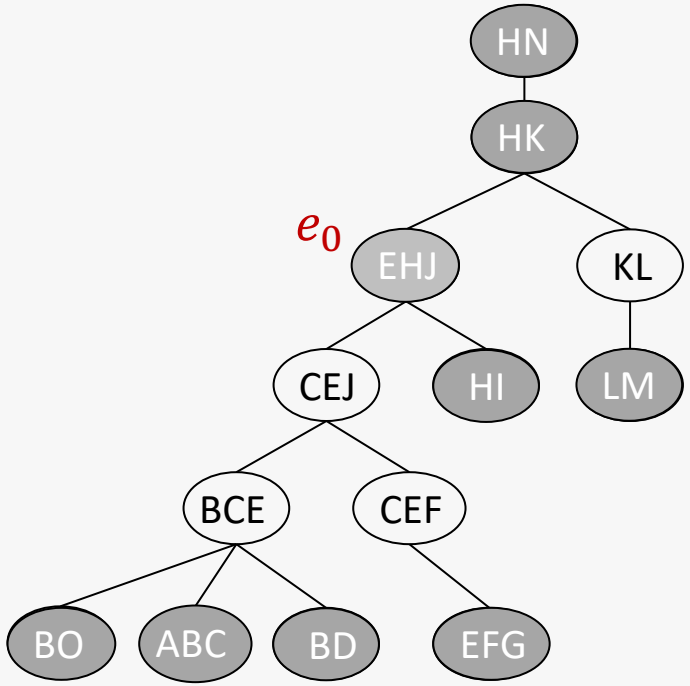
How does **CEC** lead to a WCO join algorithm?

- Sequential algorithm:
  - Compute the CP of relations in **CEC** as Q'
  - Apply semi-joins between Q' and remaining relations
- Shuffling Q' needs $O\left(\frac{N^{\rho^*}}{p}\right)$ load!
- $O\left(\frac{N}{p^{1/\rho^*}}\right)$? only the CP of ρ^* relations!



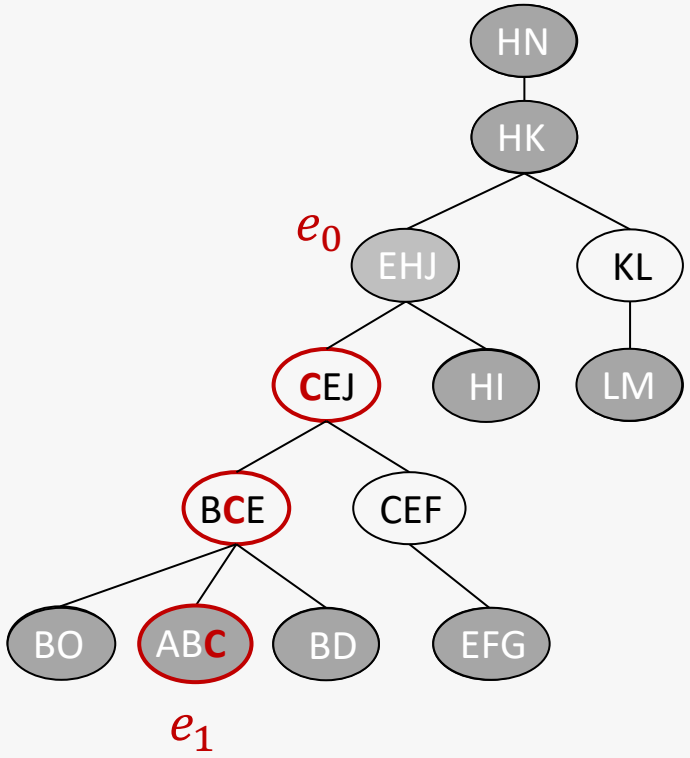
WCO Acyclic Joins

- Find an internal node e_0 in **CEC** whose descendants in **CEC** are all leaves of the join tree
- Find an attribute x and a descendant leaf node e_1 in **CEC** such that x appears in every node on the path $e_1 \rightarrow e_0$ (except e_0)



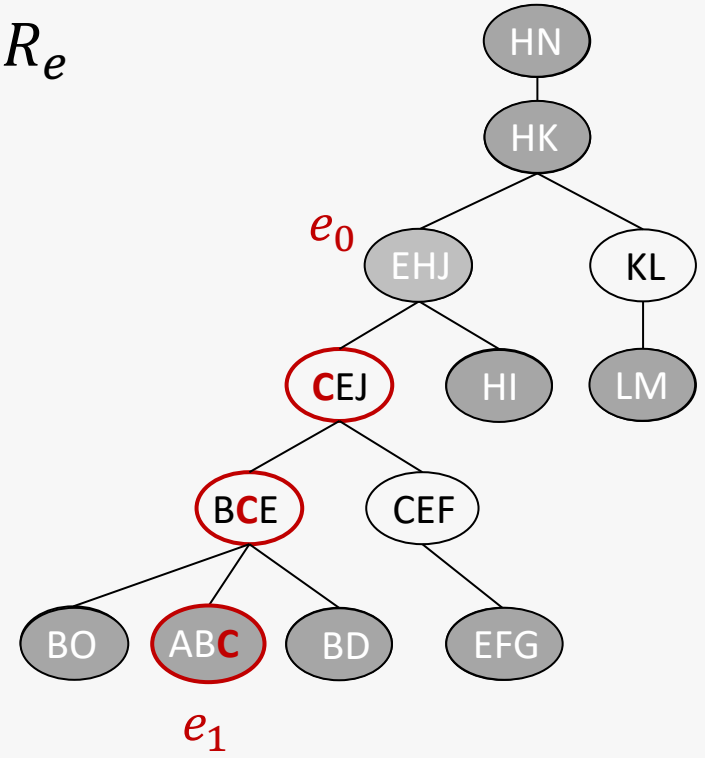
WCO Acyclic Joins

- Find an internal node e_0 in **CEC** whose descendants in **CEC** are all leaves of the join tree
- Find an attribute x and a descendant leaf node e_1 in **CEC** such that x appears in every node on the path $e_1 \rightarrow e_0$ (except e_0)



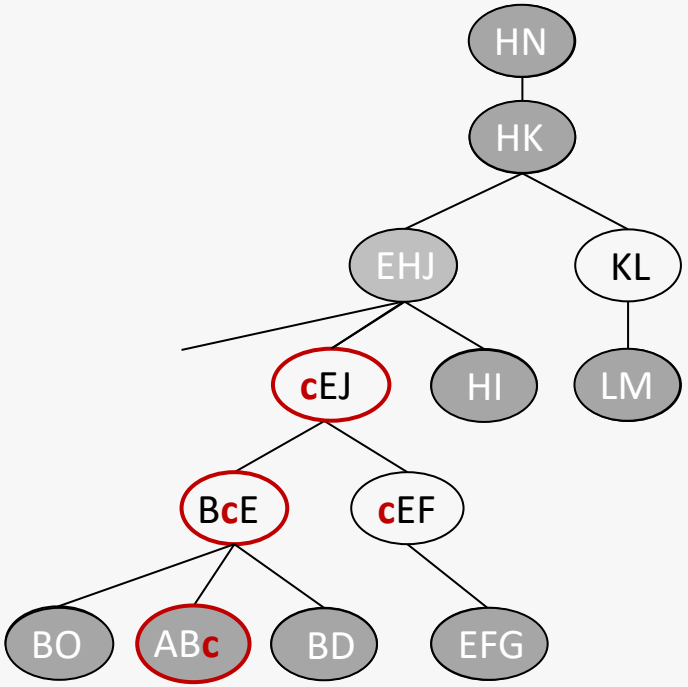
WCO Acyclic Joins

- A values c in C is heavy if its degree in R_e is larger than L for any node e on the path $e_1 \rightarrow e_0$, and light otherwise



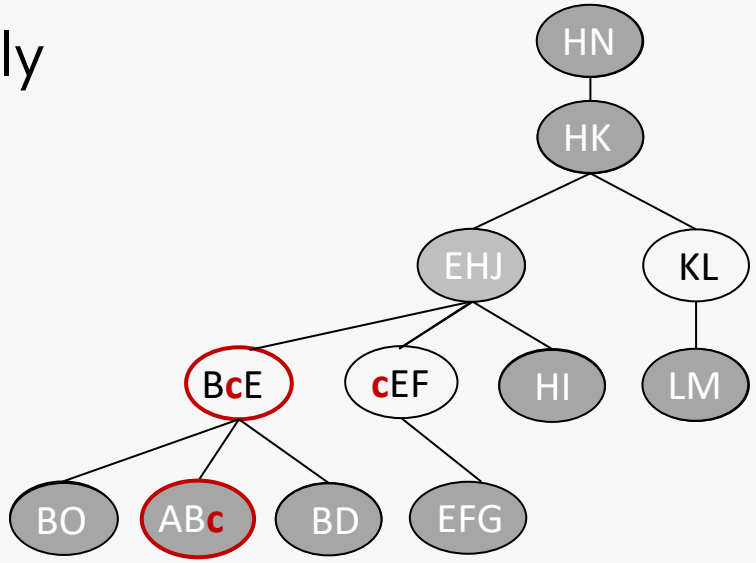
WCO Acyclic Joins

- Handle each heavy value c separately



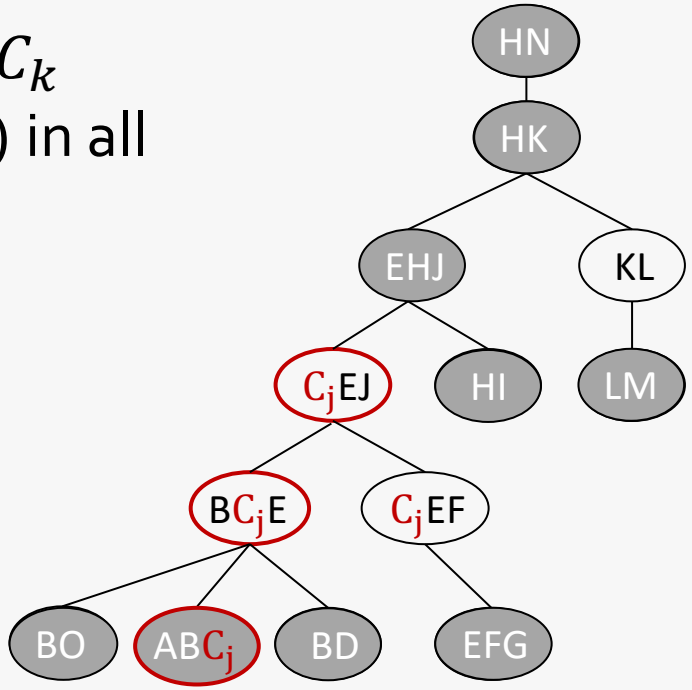
WCO Acyclic Joins

- Handle each heavy value c separately



WCO Acyclic Joins

- Put light values into groups $C_1, C_2, \cdots, C_k$ such that the total degree of C_j is $\Theta(L)$ in all R_e for any node e on the path $e_1 \rightarrow e_0$
- Handle each light group C_j separately

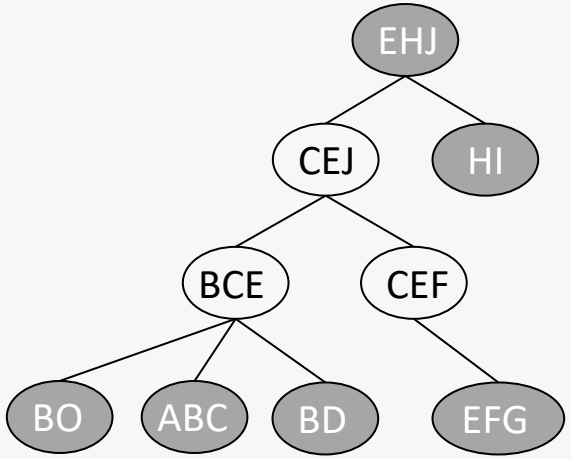


WCO Acyclic Joins

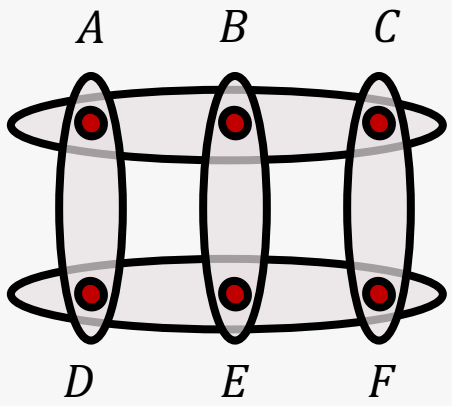
- Path Cover based on **CEC**
  {ABC–BCE–CEJ, EFG–CEF, BO, BD, HI, EHJ}
- # paths in a path cover = ρ^*
- Load Analysis:

$$\max_S \left(\frac{\prod_{e \in S}|R_e|}{p}\right)^{\frac{1}{|S|}} \leq \max_S \frac{N}{p^{1/|S|}} \leq \frac{N}{p^{1/\rho^*}}$$

  - S is a subset of relations
  - S does not contain two relations from the same path in the path cover
  - $|S| \leq \rho^*$



Lower Bound for Cyclic Joins

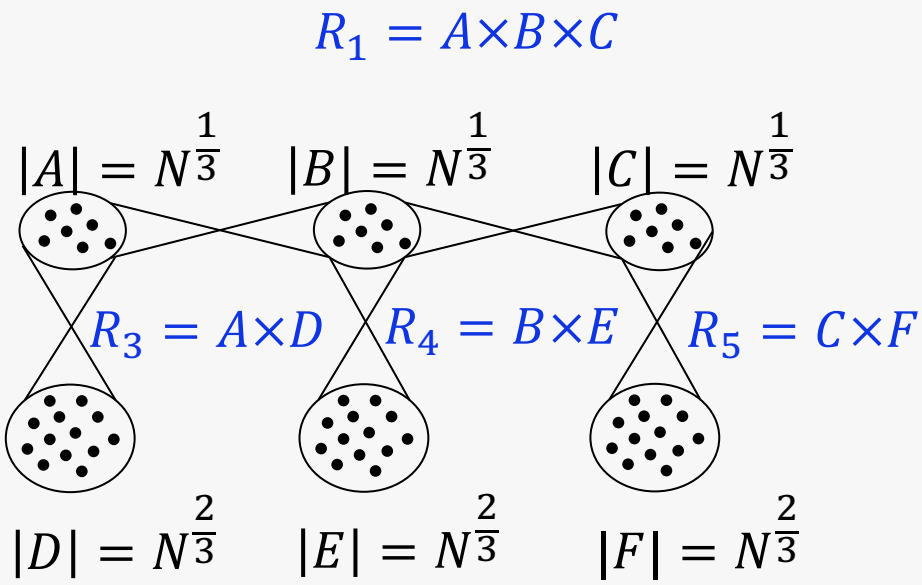


⊟ –join: $R_1(A,D) \bowtie R_2(B,E) \bowtie R_3(C,F) \bowtie R_4(A,B,C) \bowtie R_5(D,E,F)$

- ρ^*= 2: $\{A,B,C\},\{D,E,F\}$
- $\tau^* = 3$: $\{A,D\},\{B,E\},\{C,F\}$
- $\Omega\left(\frac{N}{p^{1/2}}\right)$ v. s. $\tilde{O}\left(\frac{N}{p^{1/3}}\right)$[KS16]
- $\Omega\left(\frac{N}{p^{1/3}}\right)$ [H21]

Conjecture: $L = O\left(\frac{N}{p^{1/\max\{\rho^*,\tau^*\}}}\right)$ for WCO join algorithms, where ρ^*, τ^* are the fractional edge covering and packing number of the reduced join.

Proof Idea



R_2: sample each $(d, e, f) \in D \times E \times F$ with prob. $\frac{1}{N}$

- An instance with $\Theta(N)$ tuples yields $\Theta(N^2)$ join results w.h.p.
- Each server produces $\leq 2 \cdot \frac{L^3}{N}$ join results in each round w.h.p
- Set $2 \cdot \frac{L^3}{N} \cdot p \geq N^2 \Rightarrow L \geq \frac{N}{p^{1/3}}$

TALK OUTLINE

- The MPC Model
- Joins: the skew-free case
- Joins: worst-case optimality

Joins: output-sensitive algorithms

- Joins + Aggregations

R1(A,B) ⋈ R2(B,C)

[HY17] [HYT19]

- Load: $L = O\left(\frac{N}{p} + \sqrt{\frac{OUT}{p}}\right)$
- This is output-optimal:
  - A server receiving L tuples can produce at most L^2 join results, thus $pL^2 \geq OUT$

- Step 1: Sort

- Step 2: Join values falling in one server
- Step 3: Join values falling in more than one server
  - Allocate p_b servers for b:

$$p_b \propto \left\lceil \frac{|R_1(b)|+|R_2(b)|}{\sum_{b'} |R_1(b')|+|R_2(b')|} \right\rceil + \left\lceil \frac{|R_1(b)|\cdot|R_2(b)|}{\sum_{b'} |R_1(b')|\cdot|R_2(b')|} \right\rceil$$

  - Use HyperCube to compute all Cartesian products in parallel

Output-Sensitive Acyclic Join

Sequential Pairwise

[Y81]

$O(N + OUT)$

Parallel Pairwise

[AJRSU17]

$$L = O\left(\frac{N}{p} + \frac{OUT}{p}\right)$$

not optimal even for two-table join

$$L = O\left(\frac{N}{p} + \sqrt{\frac{OUT}{p}}\right)$$

→ r-hierarchical joins

R-HIERARCHICAL JOINS

Reduced join

Hierarchical join

For every pair of attributes x and y, either atom(x) ⊆ atom(y), or atom(y) ⊆ atom(x), or atom(x) ∩ atom(y) = ∅

- Any r-hierarchical join can be computed with $L = O\left(\frac{N}{p} + \sqrt{\frac{OUT}{p}}\right)$.
- Any non-r-hierarchical join requires $L = \Omega\left(\frac{\sqrt{OUT \cdot N}}{p}\right)$ when $N \leq OUT \leq p \cdot N$

Output-Sensitive Acyclic Join

Sequential Pairwise
[Y81]
$O(N + OUT)$

Parallel Pairwise
[AJRSU17]
$$L = O\left(\frac{N}{p} + \frac{OUT}{p}\right)$$

not optimal even for two-table join
$$L = O\left(\frac{N}{p} + \sqrt{\frac{OUT}{p}}\right)$$

Sequential Hybrid GHD
for Cyclic Joins – Hung's talk

Parallel Hybrid Pairwise [HY19]
$$L = O\left(\frac{N}{p} + \frac{\sqrt{N \cdot OUT}}{p}\right)$$

(optimal if $OUT \leq p \cdot N$ and better than naive if $OUT \leq p^2 \cdot N$)

→ r-hierarchical joins

Three-Table Join

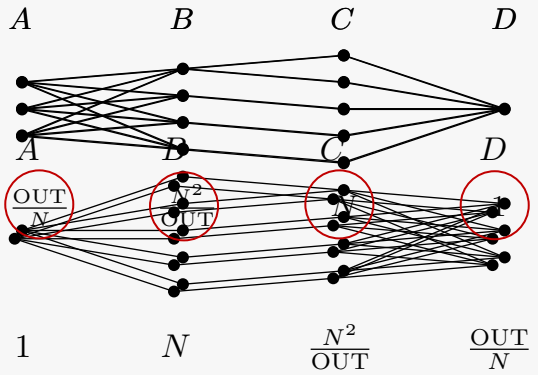
Compute $R_1(A,B) \bowtie R_2(B,C) \bowtie R_3(C,D)$

- Example 1: $(R_1 \bowtie R_2) \bowtie R_3 = O\left(\frac{N}{p} + \frac{OUT}{p}\right)$

  but $R_1 \bowtie (R_2 \bowtie R_3) = O\left(\frac{N}{p} + \sqrt{\frac{OUT}{p}}\right)$

- Example 2: $(R_1 \bowtie R_2) \bowtie R_3 = O\left(\frac{N}{p} + \sqrt{\frac{OUT}{p}}\right)$

  but $R_1 \bowtie (R_2 \bowtie R_3) = O\left(\frac{N}{p} + \frac{OUT}{p}\right)$



$|R_1 \bowtie R_2| = N$ and $|R_2 \bowtie R_3| = OUT$

- Example 3 = Example 1 + Example 2
  - Either pairwise ordering $= O\left(\frac{N}{p} + \frac{OUT}{p}\right)$

Decompose the join into multiple pieces and find a good join order for each.

Three-Table Join

- Decompose join instance

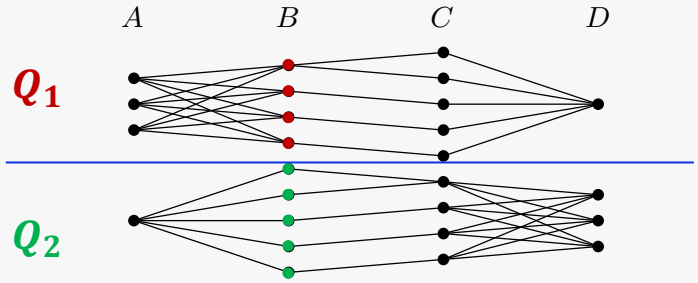
  $b \in B$ appears in $\geq \tau$ tuples in R_1

  $$Q_1 = R_1(AB) \bowtie \underbrace{R_2(BC) \bowtie R_3(CD)}_{O(OUT/\tau)}$$

  $b \in B$ appears in $< \tau$ tuples in R_1

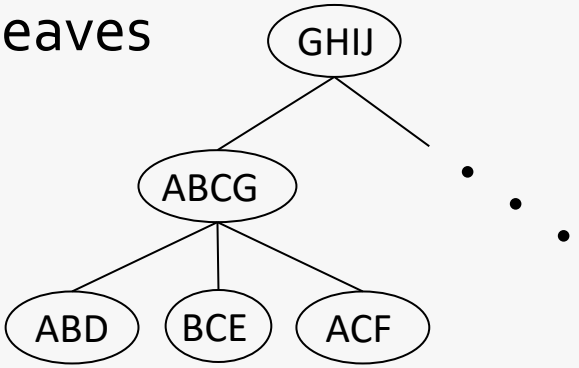
  $$Q_2 = \underbrace{R_1(AB) \bowtie R_2(BC)}_{O(N \cdot \tau)} \bowtie R_3(CD)$$

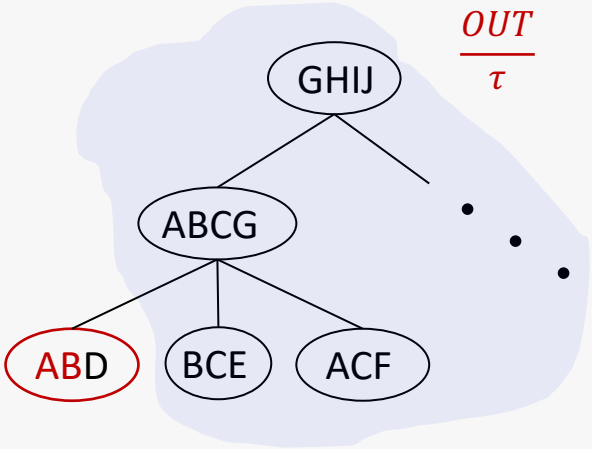
- Minimize $\frac{OUT}{\tau} + N \cdot \tau$
  - Set $\tau = \sqrt{\frac{OUT}{N}}$ to balance $\frac{OUT}{\tau} = N \cdot \tau$
  - $L = O\left(\frac{N}{p} + \frac{\sqrt{N \cdot OUT}}{p}\right)$

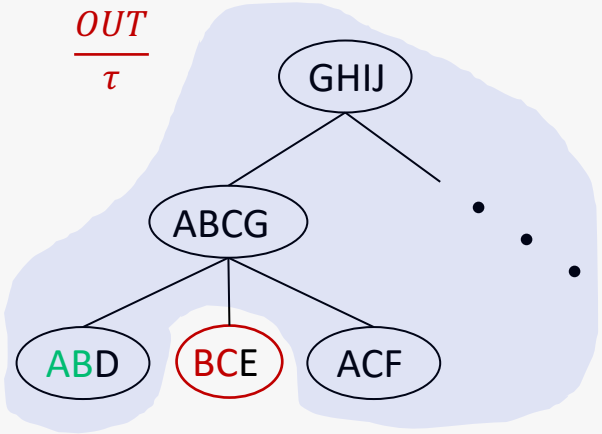


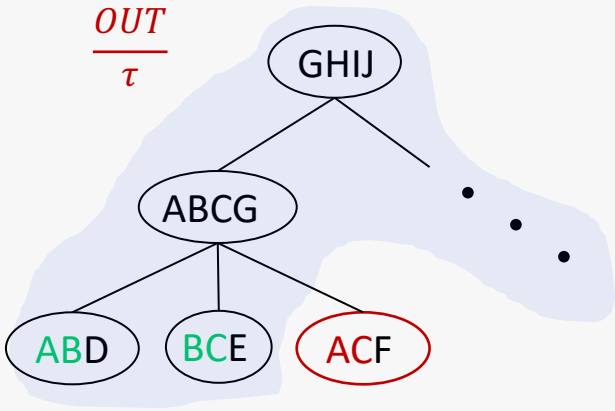
Output-Sensitive Acyclic Join

- Decompose join instance but recursion saves us efforts
- Pick an internal node whose children are all leaves
  - Handle heavy values in leaf nodes: binary join







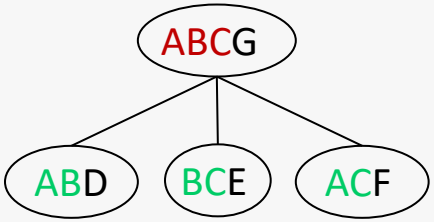


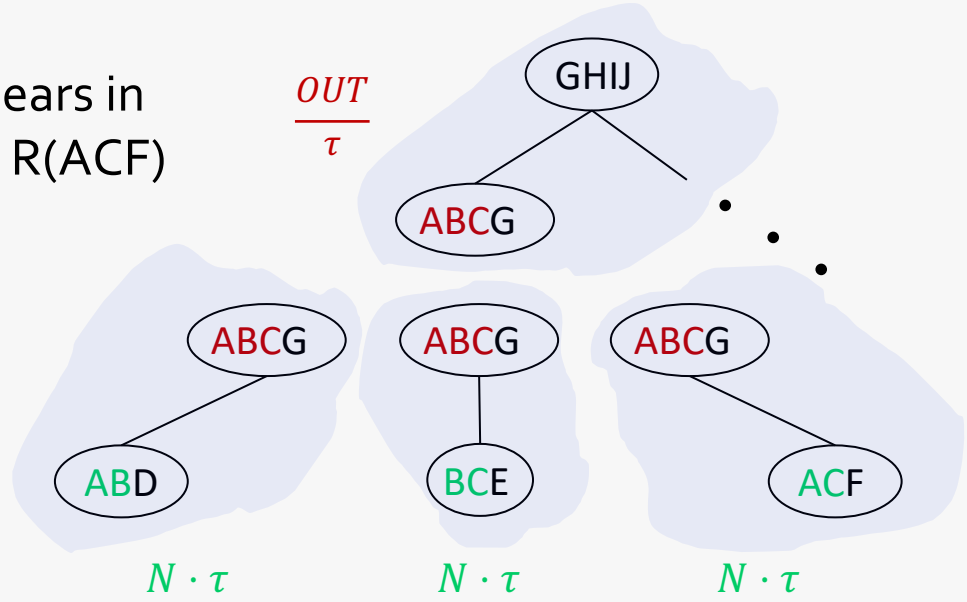
(a,b) in A×B is heavy if it appears in $\geq \tau$ tuples in R(ABD)

Output-Sensitive Acyclic Join

- Decompose join instance but recursion saves us efforts
- Pick an internal node whose children are all leaves
  - Handle heavy values in leaf nodes : binary join
  - Handle heavy values in the internal node : r-hierarchical join

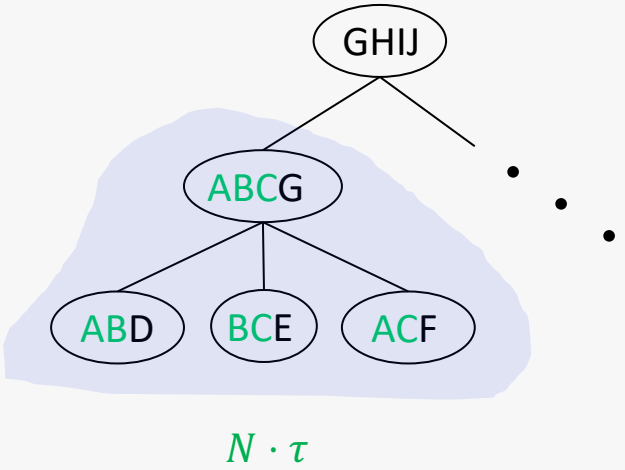
(a,b,c) in A×B×C is heavy if it appears in $\geq \tau$ tuples in R(ABD) ⋈ R(BCE) ⋈ R(ACF)





Output-Sensitive Acyclic Join

- Decompose join instance but recursion saves us efforts
- Pick an internal node whose children are all leaves
  - Handle heavy values in leaf nodes : binary join
  - Handle heavy values in the internal node : r-hierarchical join
  - Handle light values in the internal and leaf nodes : recursion



The size of intermediate join results is $O\left(\sqrt{N \cdot OUT}\right)$

Load $L = O\left(\frac{N}{p} + \frac{\sqrt{N \cdot OUT}}{p}\right)$

OUTPUT-OPTIMALITY

WCO Acyclic Joins

$$L = O\left(\frac{N}{p^{1/\rho^*}}\right)$$

+

Output-sensitive Acyclic Joins

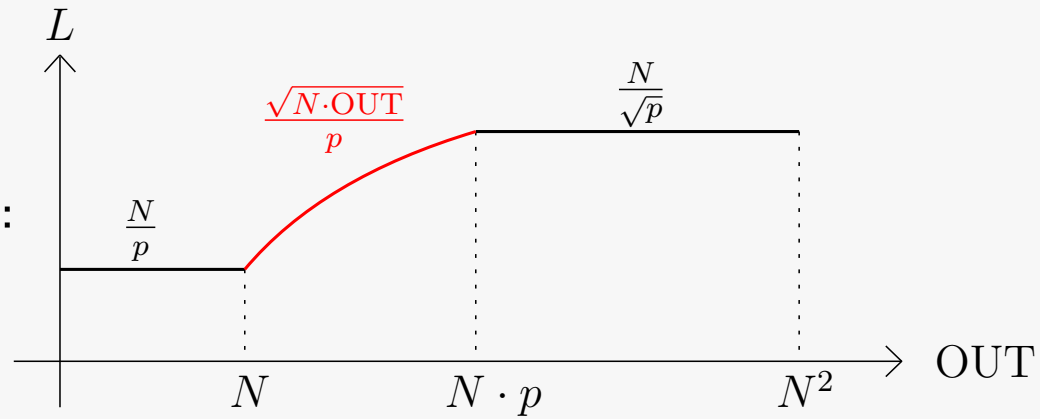
$$L = O\left(\frac{N}{p} + \frac{\sqrt{N \cdot OUT}}{p}\right)$$

(optimal if $OUT \leq p \cdot N$)

+ ?

The simplest non-r-hierarchical join:

$R_1(A,B) \bowtie R_2(B,C) \bowtie R_3(C,D)$



Yannakakis

Output-sensitive algorithm

WCO algorithm

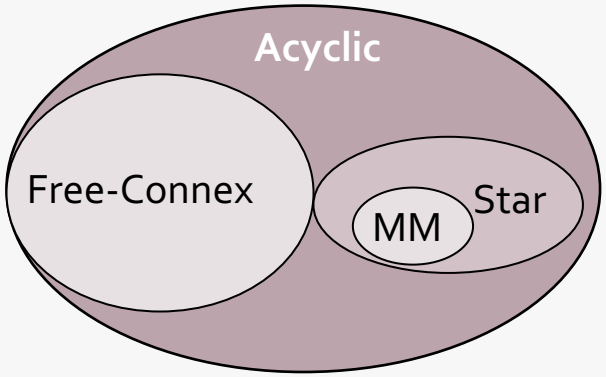
TALK OUTLINE

- The MPC Model
- Joins: the skew-free case
- Joins: worst-case optimality
- Joins: output-sensitive algorithms

Joins + Aggregations

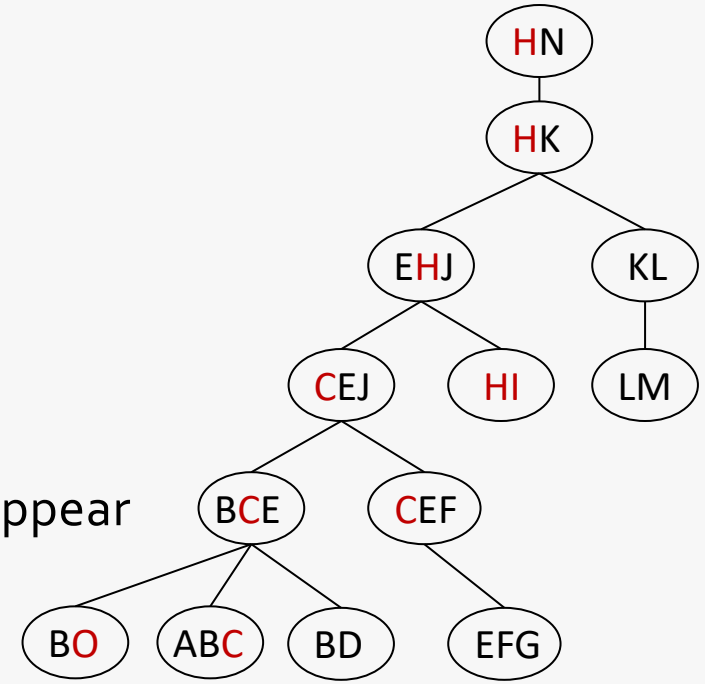
AGGREGATE YANNAKAKIS

- $L = O\left(\frac{N}{p} + \frac{J}{p}\right)$ for acyclic query
  - J is the maximum intermediate join size
- Free-Connex query: $O\left(\frac{N}{p} + \frac{OUT}{p}\right)$
- Matrix Multiplication: $O\left(\frac{N}{p} + \frac{N \cdot \sqrt{OUT}}{p}\right)$ [AP09]
- Star query: $O\left(\frac{N}{p} + \frac{N \cdot OUT^{1-\frac{1}{n}}}{p}\right)$ [AP09]
  - n is the number of relations
- General acyclic join-aggregate queries: $O\left(\frac{N}{p} + \frac{N \cdot OUT}{p}\right)$



AGGREGATE YANNAKAKIS

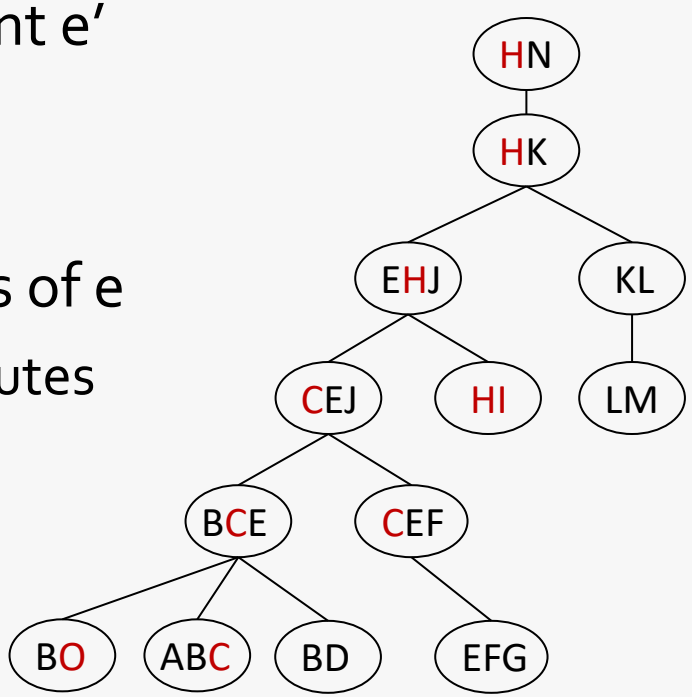
- Bottom-up Semi-joins
- Top-down Semi-joins
- Bottom-up Binary Join + Aggregate
  - Aggregate as early as possible!
  - For a non-output attribute, if it does not appear in any node above, then aggregate over it



output attributes

AGGREGATE YANNAKAKIS

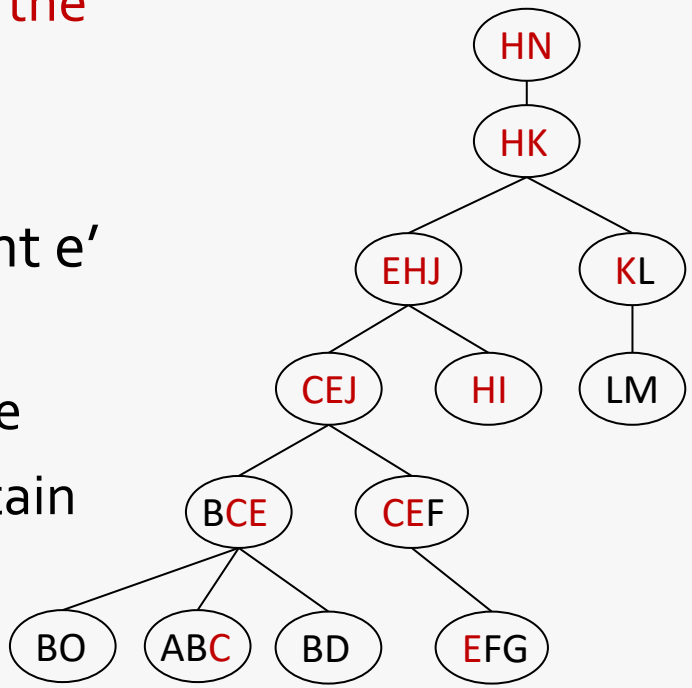
- The binary join between e and its parent e′ is bounded by N · OUT
- Let $\bar{e}$ be the non-aggregated attributes of e
  - All attributes in $\bar{e} - e'$ are output attributes



FREE-CONNEX QUERY

- Free-connex Join Tree
  - No non-output attribute appear above the topmost node of any output attribute

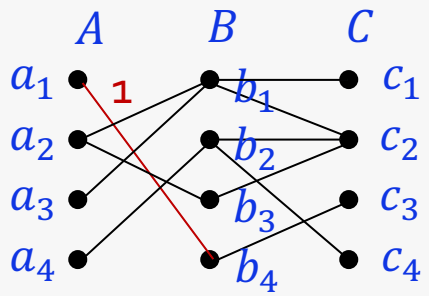
- The binary join between e and its parent e′ is bounded by N or OUT.
  - $\bar{e}$ be the non-aggregated attributes of e
  - If $\bar{e} - e' \neq \emptyset$, then e′ and $\bar{e}$ do not contain any non-output attribute



Sparse Matrix Multiplication

$$\begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} b_1 & b_2 & b_3 & b_4 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 3 & 0 & 1 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \end{array} \times \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{bmatrix} 1 & 3 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \end{bmatrix} \end{array} \!\!\begin{array}{c} b_1 \\ b_2 \\ b_3 \\ b_4 \end{array} = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{bmatrix} 0 & 0 & 2 & 0 \\ 3 & 9 & 0 & 0 \\ 2 & 6 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \end{array}$$

$$\sum_B R_1(A,B) \bowtie R_2(B,C)$$



A	B	w
a_1	b_4	1
a_2	b_1	3
a_2	b_3	1
a_3	b_1	2
a_4	b_2	1

B	C	w
b_1	c_1	1
b_1	c_2	3
b_2	c_2	1
b_2	c_4	2
b_3	c_2	1
b_4	c_3	2

A	C	w
a_1	c_3	2
a_2	c_1	3
a_2	c_2	9
a_3	c_1	2
a_3	c_2	6
a_4	c_2	1
a_4	c_4	1

Sparse Matrix Multiplication

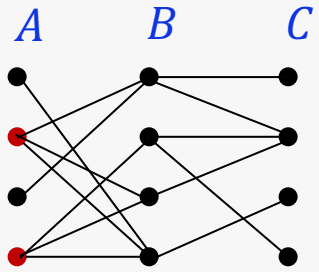
When output size is smaller than $\leq N/p$, then $L = O(N/p)$

- Sort all tuples by B

- Each server generates $\leq N/p$ triples (a, c, w)
- Reduce at most N triples by A and C

Sparse Matrix Multiplication

A worst-case optimal algorithm with $L = O\left(\frac{N}{\sqrt{p}}\right)$



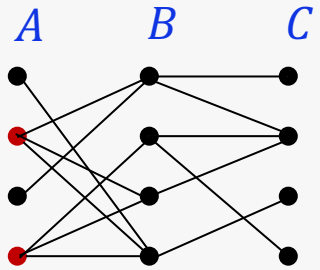
- A value a in A (c in C) is heavy if its degree $\geq L$ and light otherwise

$$\sum_{B} R_1(A^{??}, B) \bowtie R_2(B, C^{??})$$

- A^{heavy} or C^{heavy}: binary join and then aggregate
- A^{light} and C^{light} :
  - Partition A^{light} into groups $A_1, A_2, \cdots, A_k$, each with total degree $\Theta(\tau)$
  - Partition C^{light} into groups $C_1, C_2, \cdots, C_k$, each with total degree $\Theta(\tau)$
  - Send every pair (A_i, C_j) into a distinct server

Sparse Matrix Multiplication

An output-sensitive algorithm $L = \tilde{O}\left(\frac{N}{p} + \frac{N^{2/3} \cdot OUT^{1/3}}{p^{2/3}}\right)$



- Obtain w.h.p. O(1)-approximation for
  - OUT and # query results participated by each a in A

- Set $\tau = \sqrt{OUT \cdot L}$
- A value a in A is heavy if it appears in $\geq \tau$ query results and light otherwise

$$\sum_{B} R_1(A^{??}, B) \bowtie R_2(B, C)$$

Sparse Matrix Multiplication

$N \cdot \sqrt{\frac{OUT}{L}}$ intermediate results

- A^{heavy}: binary join and then aggregate
- Partition A^{light} into groups $A_1, A_2, \cdots, A_k$, each participating in $\Theta(\tau)$ query results

Consider $\sum_B R_1(A_i, B) \bowtie R_2(B, C)$

- Obtain w.h.p O(1)-approximation for #query results participated by each c in C
- A value c in C is heavy if it appears in $\geq L$ query results and light otherwise

$$\sum_B R_1(A_i, B) \bowtie R_2(B, C^{??})$$

Sparse Matrix Multiplication

- C^{heavy}: binary join and then aggregate

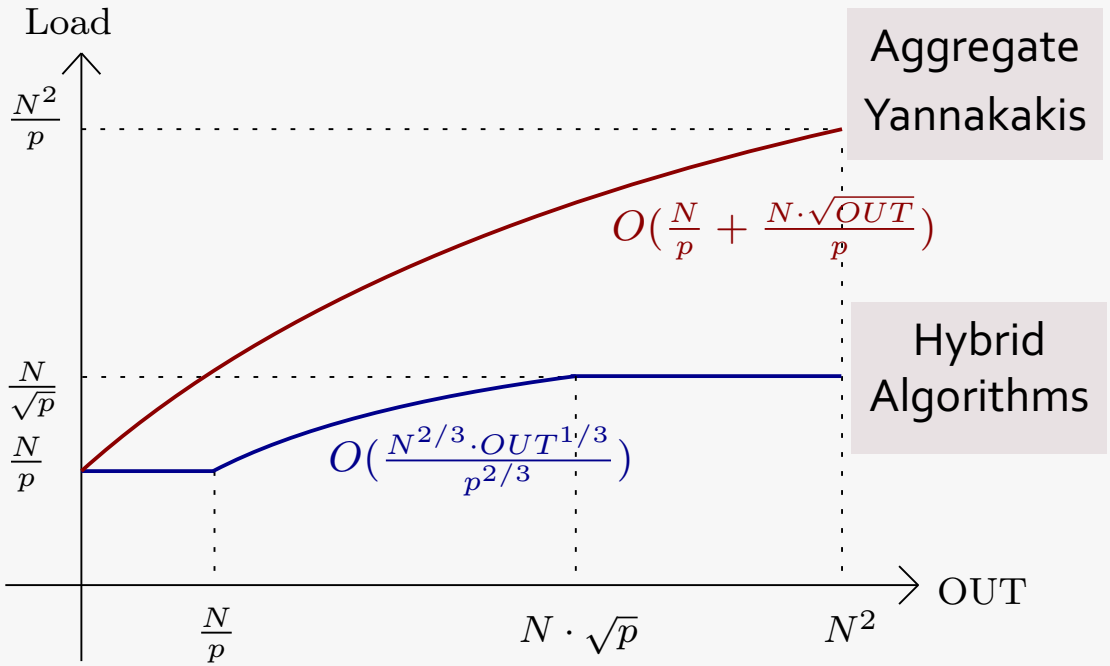
$$N \cdot \sqrt{\frac{OUT}{L}}$$
intermediate results

- Partition C^{light} into groups $C_1, C_2, \cdots, C_k$, each participating in $\Theta(\tau')$ query results

- Consider each $\sum_B R_1(A_i, B) \bowtie R_2(B, C_j)$ separately
  - Small output size $\leq L$

Sparse Matrix Multiplication

Output-optmal load $\mathrm{L} = \tilde{O}\left(\min\left\{\frac{N}{\sqrt{p}}, \frac{N}{p} + \frac{N^{2/3} \cdot OUT^{1/3}}{p^{2/3}}\right\}\right)$



Aggregate Yannakakis

Hybrid Algorithms

Aggregate Yannakakis

Output-sensitive algorithm

WCO algorithm

Star Matrix Multiplication

$$\sum_B R_1(A_1, B) \bowtie R_2(A_2, B) \bowtie \cdots \bowtie R_k(A_k, B)$$

- Find a partition of $\{1,2,\cdots,k\}$ as (I, J)
- Materialize $R_1'(A_I, B) = \bowtie_{i \in \mathrm{I}} R_i$ and $R_2'(A_J, B) = \bowtie_{i \in J} R_i$
- Compute $\sum_B R_1(A_I, B) \bowtie R_2(A_J, B)$

Suppose $d_1(b) \geq d_2(b) \geq \cdots \geq d_k(b)$ for every b in B

$$\max\left\{\prod_{i=3:\text{odd } i}^{k} d_i(b), \prod_{i=3:\text{odd } i}^{k} d_i(b)\right\} \leq \sqrt{\prod_{i=1}^{k} d_i(b)} \leq \sqrt{OUT}$$

- $|R_1'|, |R_2'| \leq N \cdot \sqrt{OUT}$ (I, J chooses odd and even indexes separately)

Chain Matrix Multiplication

$$\sum_{A_2,\cdots,A_k} R_1(A_1, A_2) \bowtie R_2(A_2, A_3) \bowtie \cdots \bowtie R_k(A_k, A_{k+1})$$

- A value a in A_2 is heavy if it can be joined with $\geq \tau$ distinct values in A_1 and light otherwise

$\frac{N \cdot OUT}{\tau}$ intermediate results

- $\sum_{A_2} R_1\left(A_1, A_2^{\text{heavy}}\right) \bowtie R\left(A_2^{\text{heavy}}, A_{k+1}\right)$
  - $R\left(A_2^{\text{heavy}}, A_{n+1}\right) = \sum_{A_3,\cdots,A_k} R_2(A_2, A_3) \bowtie \cdots \bowtie R_k(A_k, A_{k+1})$
- $\sum_{A_3,\cdots,A_k} R(A_1, A_3) \bowtie R_3(A_3, A_4) \bowtie \cdots \bowtie R_k(A_k, A_{k+1})$
  - $R(A_1, A_3) = \sum_{A_2} R_1\left(A_1, A_2^{\text{light}}\right) \bowtie R_2\left(A_2^{\text{light}}, A_3\right)$

$N \cdot \tau$ intermediate results

Other Join-Aggregate Queries

Join-Aggregate Query	Aggregated Yannakakis	New Results
Matrix Multiplication	$O\left(\frac{N}{p} + \frac{N \cdot \sqrt{OUT}}{p}\right)$	$\tilde{O}\left(\min\left\{\frac{N}{\sqrt{p}}, \frac{N}{p} + \frac{N^{2/3} \cdot OUT^{1/3}}{p^{2/3}}\right\}\right)$ Optimal for $N \geq 2$ and $N \leq OUT \leq N^2$
Star	$O\left(\frac{N}{p} + \frac{N \cdot OUT^{1-\frac{1}{n}}}{p}\right)$	$\tilde{O}\left(\left(\frac{N \cdot OUT}{p}\right)^{2/3} + \frac{N \cdot \sqrt{OUT}}{p} + \frac{N+OUT}{p}\right)$
Line	$O\left(\frac{N}{p} + \frac{N \cdot OUT}{p}\right)$	
Tree		$\tilde{O}\left(\frac{N \cdot OUT^{2/3}}{p} + \frac{N+OUT}{p}\right)$

- Elementary products for General queries?
- Other join-aggregate and join-project queries?
- Different semiring additions?

THANK YOU !