Accessing Answers to Unions of Conjunctive Queries with Ideal Time Guarantees

Nofar Carmeli

Plan

- Enumeration
  - Join queries
    - Self-joins
  - Conjunctive queries
  - Unions of conjunctive queries
- Other Evaluation Tasks
  - The tasks
  - Known complexity results

Plan

- **Enumeration**
  - **Join queries**
    - Self-joins
  - Conjunctive queries
  - Unions of conjunctive queries
- Other Evaluation Tasks
  - The tasks
  - Known complexity results

Example: Join Query

Problem

Description	Room
Moisture	5/129
Broken ceiling	Cafeteria
Missing board	5/127

Office

Room	Person
5/127	Nofar
5/128	Florent
5/128	Guillaume
5/129	David
5/129	Akira

Contact

Person	Email
Nofar	nc@lirmm.fr
Florent	ft@lirmm.fr
Guillaume	gpk@lirmm.fr
David	dc@lirmm.fr

$$Q(E, P, R, D) \leftarrow \text{Problem}(D, R), \text{Office}(R, P), \text{Contact}(P, E)$$

$$\{(E, P, R, D) | (D, R) \in \text{Problem}, (R, P) \in \text{Office}, (P, E) \in \text{Contact}\}$$

Email	Person	Room	Description
nc@lirmm.fr	Nofar	5/127	Missing board
dc@lirmm.fr	David	5/129	Moisture

Challenges

- Many answers
- Many intermediate answers

R

x	y
a1	b1
a2	b1
a3	b1

S

y	z
b1	c1
b1	c2

T

x	z
a2	c1
a4	c2

dangling tuples

$$Q_1(x, y, z) \leftarrow \mathrm{R}(x, y), \mathrm{S}(y, z)$$

x	y	z
a1	b1	c1
a1	b1	c2
a2	b1	c1
a2	b1	c2
a3	b1	c1
a3	b1	c2

$$Q_2(x, y, z) \leftarrow \mathrm{R}(x, y), \mathrm{S}(y, z), \mathrm{T}(x, z)$$

x	y	z
a2	b1	c1

Example: Algorithm

Problem

Description	Room
Moisture	5/129
~~Broken ceiling~~	~~Cafeteria~~
Missing board	5/127

Office

Room	Person
5/127	Nofar
~~5/128~~	~~Florent~~
~~5/128~~	~~Guillaume~~
5/129	David
~~5/129~~	~~Akira~~

Contact

Person	Email
Nofar	nc@lirmm.fr
~~Florent~~	~~ft@lirmm.fr~~
~~Guillaume~~	~~gpk@lirmm.fr~~
David	dc@lirmm.fr

$$Q(E, P, R, D) \leftarrow \mathrm{Problem}(D, R), \mathrm{Office}(R, P), \mathrm{Contact}(P, E)$$

Email	Person	Room	Description
nc@lirmm.fr	Nofar	5/127	Missing board
dc@lirmm.fr	David	5/129	Moisture

Example: Algorithm Fails

Registration

Student	Exam
Anna	algorithms
Thomas	databases

Staff

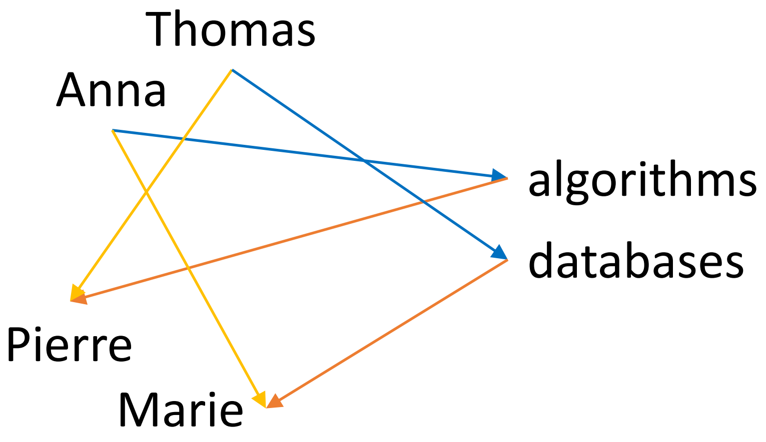
Exam	Professor
algorithms	Pierre
databases	Marie

COI

Student	Professor
Thomas	Pierre
Anne	Marie

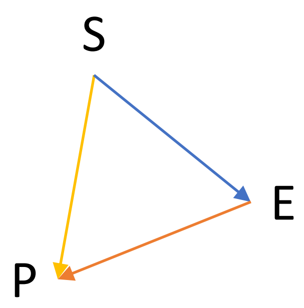
$$Q(S, E, P) \leftarrow \text{Registration}(S, E), \text{Staff}(E, P), \text{COI}(S, P)$$

Database



No query answers

Query



Complexity Guarantees

- Data complexity
  - input = database
  - query size = constant

- Possibly: output $\gg$ input
  (Polynomial number of answers)

- Minimal requirements:
  - Linear time (to read input)
  - Constant time per answer (to print output)

- RAM model
- We allow log factors

Complexity Guarantees

- Worst-case-optimal total time [Atserias, Grohe, Marx; FOCS 08]
  - Linear in input + worst-case output

- Instance-optimal total time (also relevant)
  - Linear in input + output

- Enumeration ("ideal"; our focus)
  - Preprocessing: linear in input
  - Delay: constant

Research Question

- Goal: Given a query, what is the most efficient algorithm?
- Type of results:
  Can we solve a task for a given query in a given time complexity?

Yes / No

Acyclicity

- A query that has a join tree is called <u>acyclic</u>

2. tree

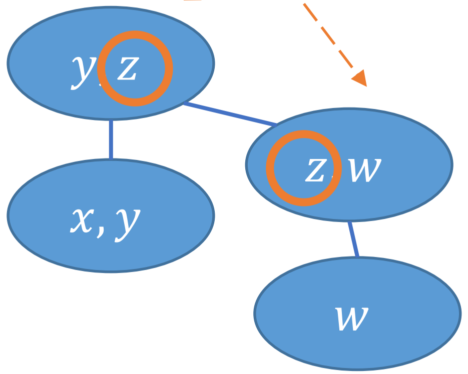
3. For every variable: the nodes containing it form a subtree

1. a node for every atom

Query: $Q_1(x, y, z, w) \leftarrow R(x, y), S(y, z), T(z, w), U(w)$

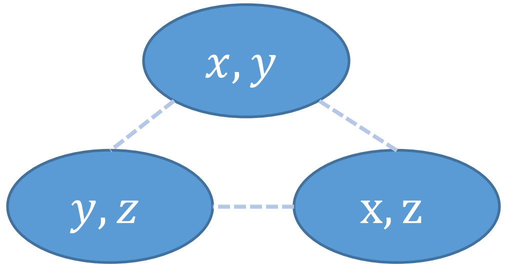
acyclic

Join Tree:



cyclic

Query: $Q_2(x, y, z) \leftarrow R(x, y), S(y, z), T(x, z)$



Dichotomy

[BaganDurandGrandjean CSL'2007]
[Brault-Baron 2013]
[Bringmann, **C**, Mengel 2022]

- Given a join query Q,

If Q is acyclic, Q ∈ Enum<lin,const>

If Q is cyclic, Q ∉ Enum<lin,const>*

* no self-joins, assuming sHyperclique or Zero-Clique

Acyclic Joins

[Yannakakis 81]

- An efficient algorithm for acyclic joins
  1. Find a join tree and set a root
  2. Remove dangling tuples
  3. Join

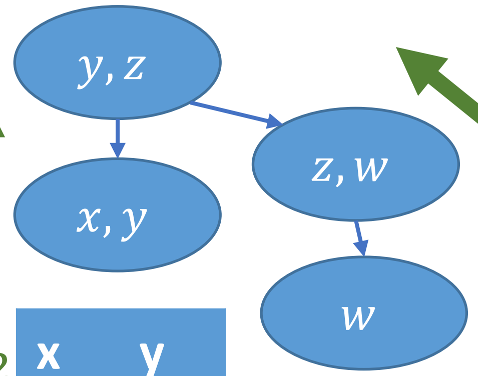
1. Leaf-to-root:
   $r_{parent} \leftarrow r_{parent} \ltimes r_{child}$
2. Root-to-leaf:
   $r_{child} \leftarrow r_{child} \ltimes r_{parent}$

R_1

y	z
y1	z1
y1	z2
y2	z2
~~y3~~	~~z1~~
~~y4~~	~~z3~~

R_2

x	y
x1	y1
x1	y2
x2	y2
~~x2~~	~~y4~~



R_3

z	w
z1	w1
z2	w2
~~z3~~	~~w3~~

R_4

w
w1
w2

No dangling tuples!

Acyclic Joins

- An efficient algorithm for acyclic joins
  1. Find a join tree and set a root
  2. Remove dangling tuples
  3. Join

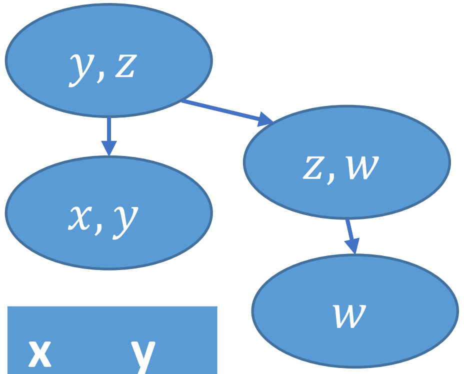
```
for t1 in R1:
    for t2 in R2 matching t1:
        for t3 in R3 matching t1,t2:
            for t4 in R4 matching t1,t2,t3:
                output t1,t2,t3,t4
```

R_1

y	z
y1	z1
y1	z2
y2	z2
~~y3~~	~~z1~~
~~y4~~	~~z3~~

R_2

x	y
x1	y1
x1	y2
x2	y2
~~x2~~	~~y4~~



R_3

z	w
z1	w1
z2	w2
~~z3~~	~~w3~~

R_4

w
w1
w2

Dichotomy

[BaganDurandGrandjean CSL'2007]
[Brault-Baron 2013]
[Bringmann, **C**, Mengel 2022]

- Given a join query Q,

If Q is acyclic, $Q \in \mathrm{Enum}\langle\mathrm{lin},\mathrm{const}\rangle$

If Q is cyclic, $Q \notin \mathrm{Enum}\langle\mathrm{lin},\mathrm{const}\rangle$*

* no self-joins, assuming sHyperclique or Zero-Clique

Example: Algorithm Fails

Registration

Student	Exam
Anna	algorithms
Thomas	databases

Staff

Exam	Professor
algorithms	Pierre
databases	Marie

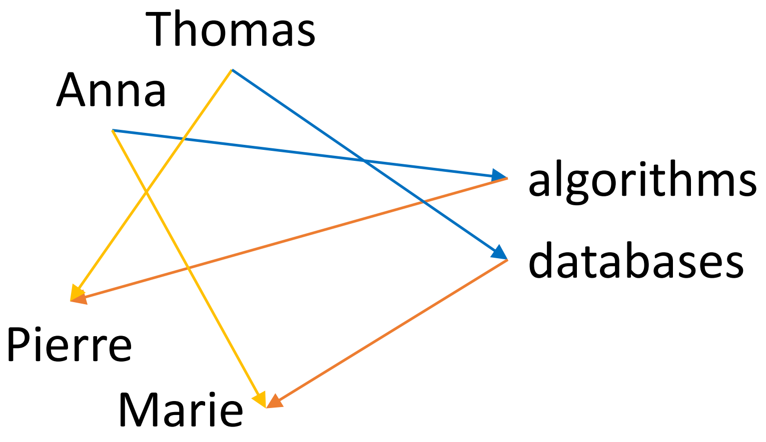
COI

Student	Professor
Thomas	Pierre
Anne	Marie

$$Q_\Delta(x,y,z) \leftarrow R_1(x,y), R_2(y,z), R_3(x,z)$$

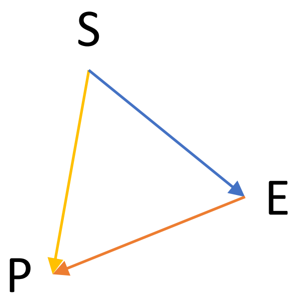
$$Q(S,E,P) \leftarrow \text{Registration}(S,E), \text{Staff}(E,P), \text{COI}(S,P)$$

Database



No query answers

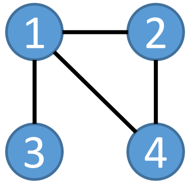
Query



Example: Conditional Lower Bound

[Brault-Baron 13]

Assumption: cannot detect triangles in a graph in linear time



edges (a,b) with a<b

Q

x	y	z
1	2	4

$R_1 = R_2 = R_3$

1	2
1	3
1	4
2	4

$$Q_\Delta(x,y,z) \leftarrow R_1(x,y), R_2(y,z), R_3(x,z)$$

first answer in linear time $\implies$ triangle in linear time $\implies$ not possible

sHyperclique Hypothesis

- $(k, k-1)$-hyperclique: k vertices, each $k-1$ of them form an edge.

- sHyperclique Hypothesis:
  $\forall k \geq 3$, deciding the existence of a $(k, k-1)$-hyperclique in a hypergraph with m edges cannot be done in time $O(m)$.

- Lemma:
  A cyclic hypergraph contains an induced k-cycle or an induced $(k, k-1)$-hyperclique for some $k \geq 3$.

Dichotomy

[BaganDurandGrandjean CSL’2007]
[Brault-Baron 2013]
[Bringmann, **C**, Mengel 2022]

- Given a join query Q,

If Q is acyclic, $Q \in \text{Enum}\langle\text{lin,const}\rangle$

If Q is cyclic, $Q \notin \text{Enum}\langle\text{lin,const}\rangle$*

* no self-joins, assuming sHyperclique or Zero-Clique

RAM Model Subtleties

- Constant time in the RAM model, what does it mean?
- Assumptions:
  - Length of registers: $\theta(\log n)$
  - Basic operations in $O(1)$
  - Available memory: $O(n^c)$ / $O(n)$
  - Modified memory: everything / $O(n)$
  - Modified memory during enumeration: everything / … / $O(1)$
- Implications:
  - Domain values $\leq n^c$
  - Sorting the input in $O(n)$
    - Radix Sort handles k integers, each bounded by n^c, in time $O(ck + cn)$
  - If $O(n^c)$ available memory,
    - Lookup table with k elements: construction in $O(k)$, search in $O(1)$

"saves" log factors

n = size of input database

RAM Model Subtleties

- Constant time in the RAM model, what does it mean?
- Assumptions:
  - Length of registers: $\theta(\log n)$
  - Basic operations in $O(1)$
  - Available memory: $\boldsymbol{O(n^c)}$ / $O(n)$
  - Modified memory: **everything** / $O(n)$
  - Modified memory during enumeration: **everything** / … / $O(1)$
- Implications:
  - Domain values $\leq n^c$
  - Sorting the input in $O(n)$
    - Radix Sort handles k integers, each bounded by n^c, in time $O(ck + cn)$
  - If $O(n^c)$ available memory,
    - Lookup table with k elements: construction in $O(k)$, search in $O(1)$

"saves" log factors

- **In this talk, assume the relaxed model**

n = size of input database

Plan

- Enumeration
  - Join queries
    - **Self-joins**
  - Conjunctive queries
  - Unions of conjunctive queries
- Other Evaluation Tasks
  - The tasks
  - Known complexity results

Dichotomy

[BaganDurandGrandjean CSL'2007]
[Brault-Baron 2013]
[Bringmann, **C**, Mengel 2022]

- Given a join query Q,

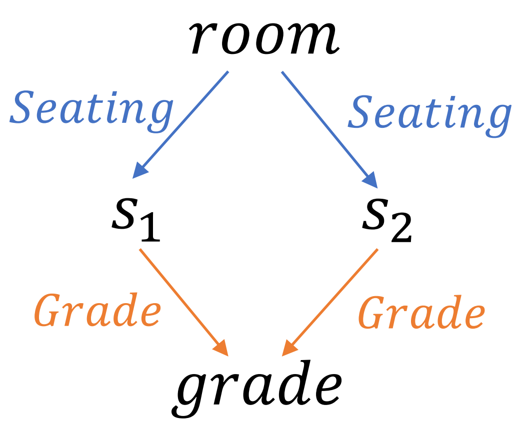
If Q is acyclic, $Q \in \mathrm{Enum}\langle lin, const\rangle$

If Q is cyclic, $Q \notin \mathrm{Enum}\langle lin, const\rangle$*

* no self-joins, assuming sHyperclique or Zero-Clique

Example 1

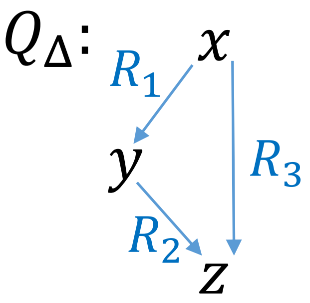
$$Q(s_1, s_2, room, grade) \leftarrow$$
$$Seating(room, s_1), Seating(room, s_2), Grade(s_1, grade), Grade(s_2, grade)$$

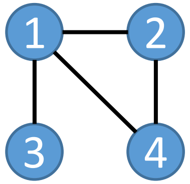


Lower Bound: Cyclic Joins

[Brault-Baron 13]

Assumption: cannot detect triangles in a graph in linear time





edges (a,b) with a<b

$\boldsymbol{Q}$

x	y	z
1	2	4

$\boldsymbol{R_1 = R_2 = R_3}$

1	2
1	3
1	4
2	4

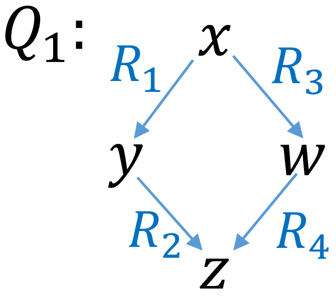
Cyclic: $Q_\Delta(x,y,z) \leftarrow R_1(x,y), R_2(y,z), R_3(x,z)$

first answer in linear time $\Longrightarrow$ triangle in linear time $\Longrightarrow$ not possible

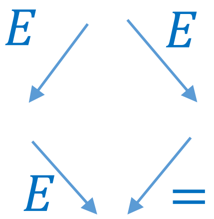
Lower Bound: Cyclic Joins

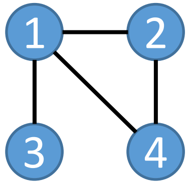
[Brault-Baron 13]

Assumption: cannot detect triangles in a graph in linear time



Construction:





edges (a,b) with a<b

Q

x	y	z	w
1	2	4	4

$R_1 = R_2 = R_3$

1	2
1	3
1	4
2	4

R_4

1	1
2	2
3	3
4	4

with self-joins,
cannot assign different relations
to different atoms

Cyclic: $Q_1(x,y,z,w) \leftarrow R_1(x,y), R_2(y,z), R_3(x,w),$ ~~R_4~~ R_3 (w,z)

first answer in linear time $\Longrightarrow$ triangle in linear time $\Longrightarrow$ not possible

Algorithm

[C Segoufin PODS'2023]

Algorithm

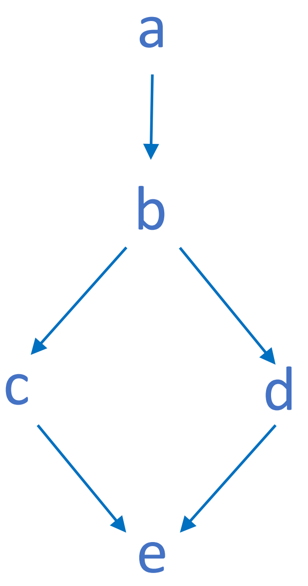
α = empty dictionary
for answer (x, u, y) to I :
  output (x, u, u, y)
  for v in $\alpha(x, y)$:
    output (x, u, v, y)
    output (x, v, u, y)
  $\alpha(x, y)$.insert(u)

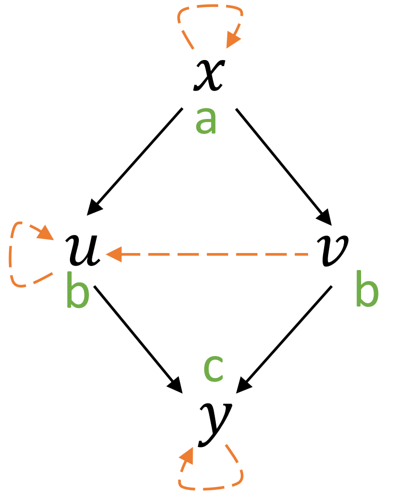
Query

$$Q(x, u, v, y) \leftarrow R(x, u), R(u, y), R(x, v), R(v, y)$$

Database

R	
a	b
b	c
b	d
c	e
d	e





endomorphism

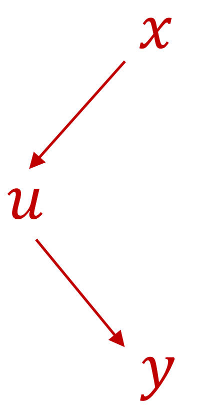


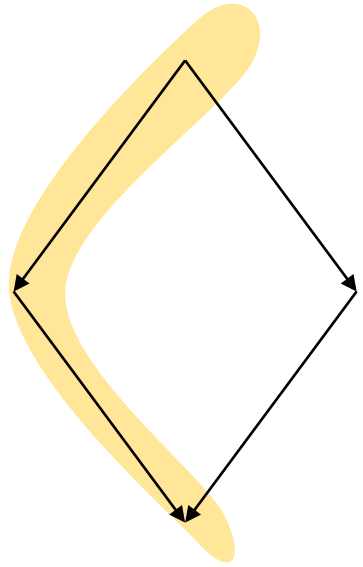
Image I

Answers

I answers		
a	b	c
a	b	d
b	c	e
b	d	e

Q answers			
b	c	d	e
b	d	c	e
a	b	b	c
a	b	b	d
b	c	c	e
b	d	d	e

Examples: Full CQs

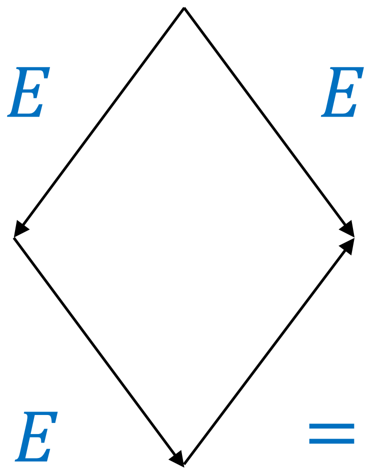


∈ Enum<lin,const>

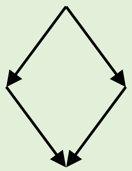
∉ Enum<lin,const> *

* assuming sTriangle

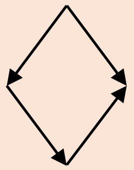
Examples: Full CQs



∈ Enum<lin,const>

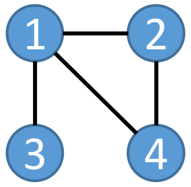


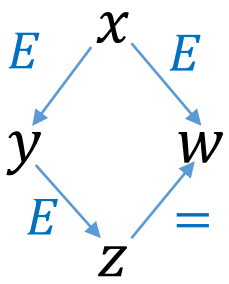
∉ Enum<lin,const> *



* assuming sTriangle

Hardness Proof





$R(x, y) \leftarrow E$

1	2
1	3
1	4
2	4

$R(y, z) \leftarrow E$

1	2
1	3
1	4
2	4

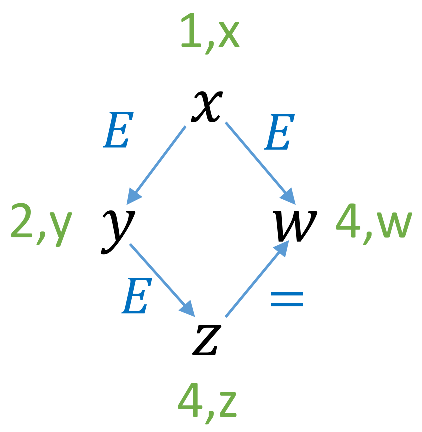
$R(x, w) \leftarrow E$

1	2
1	3
1	4
2	4

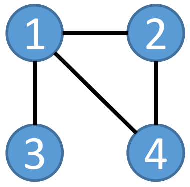
$R(w, z) \leftarrow =$

1	1
2	2
3	3
4	4

Hardness Proof



Works because Q is a core!



$R(x,y) \leftarrow E$

1,x	2,y
1,x	3,y
1,x	4,y
2,x	4,y

$R(y,z) \leftarrow E$

1,y	2,z
1,y	3,z
1,y	4,z
2,y	4,z

$R(x,w) \leftarrow E$

1,x	2,w
1,x	3,w
1,x	4,w
2,x	4,w

$R(w,z) \leftarrow =$

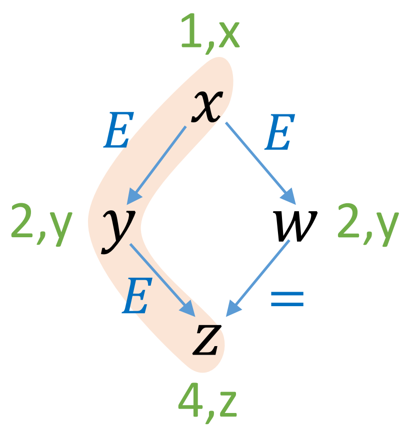
1,w	1,z
2,w	2,z
3,w	3,z
4,w	4,z

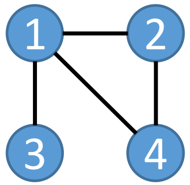
union $\Longrightarrow$

R

1,x	2,y
1,x	3,y
1,x	4,y
2,x	4,y
1,y	2,z
1,y	3,z
1,y	4,z
2,y	4,z
1,x	2,w
1,x	3,w
1,x	4,w
2,x	4,w
1,w	1,z
…	…

Hardness Proof Fails





$R(x, y) \leftarrow E$

1,x	2,y
1,x	3,y
1,x	4,y
2,x	4,y

$R(y, z) \leftarrow E$

1,y	2,z
1,y	3,z
1,y	4,z
2,y	4,z

$R(x, w) \leftarrow E$

1,x	2,w
1,x	3,w
1,x	4,w
2,x	4,w

$R(w, z) \leftarrow =$

1,w	1,z
2,w	2,z
3,w	3,z
4,w	4,z

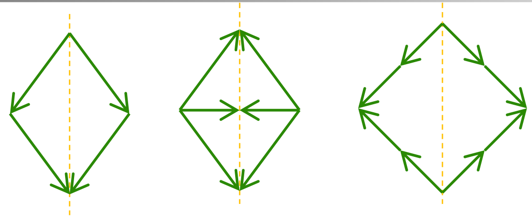
union $\Longrightarrow$

R

1,x	2,y
1,x	3,y
1,x	4,y
2,x	4,y
1,y	2,z
1,y	3,z
1,y	4,z
2,y	4,z
1,x	2,w
1,x	3,w
1,x	4,w
2,x	4,w
1,w	1,z
…	…

Sufficient and Necessary Conditions

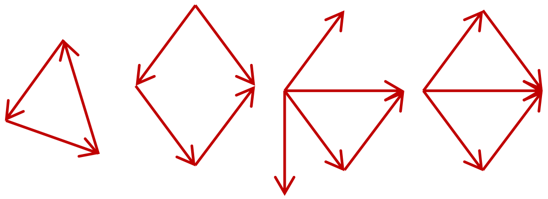
Let Q be a full CQ.



Mirror: isomorphism between two acyclic halves, identity on common variables

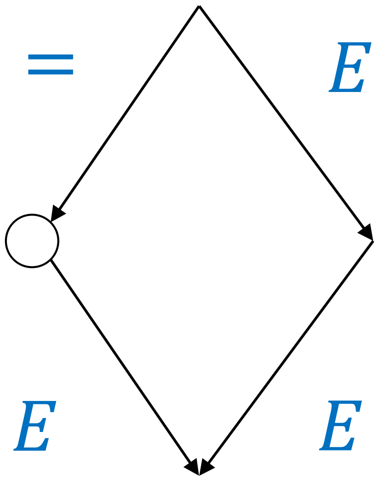
If Q is a mirror, then $Q \in$ Enum<lin,const>

If Q has a cyclic core, then $Q \notin$ Enum<lin,const> *



* assuming sHyperclique

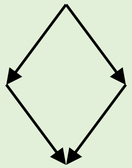
Examples: Full CQs



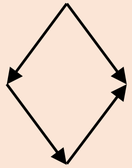
Unlike the self-join-free case,
may affect the complexity:

- reordering variables inside an atom

∈ Enum<lin,const>

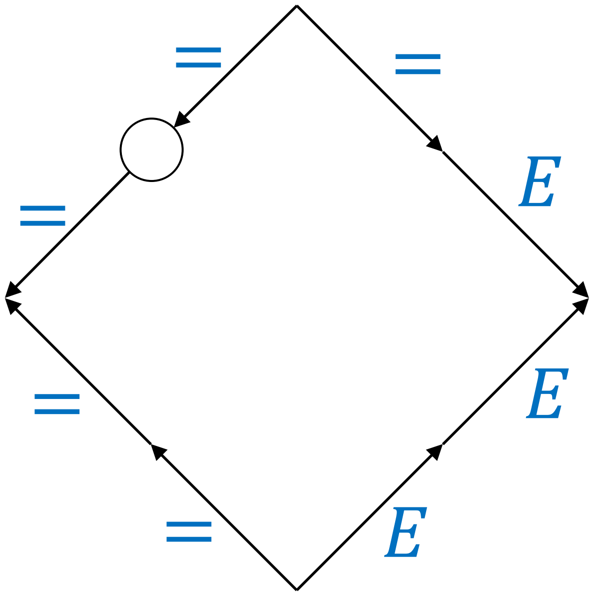


∉ Enum<lin,const> *



* assuming sTriangle

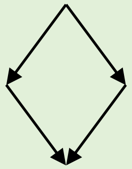
Examples: Full CQs



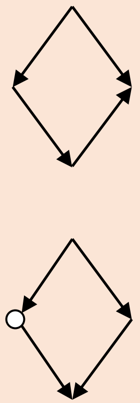
Unlike the self-join-free case,
may affect the complexity:

- reordering variables inside an atom
- introducing unary atoms

∈ Enum<lin,const>

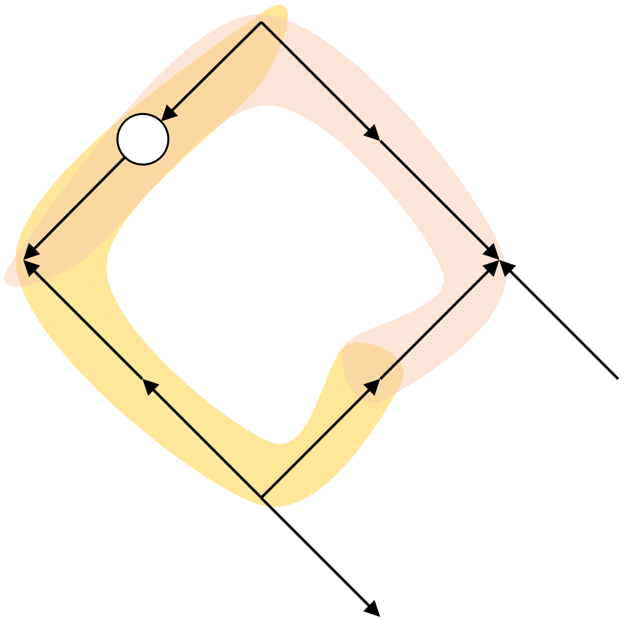


∉ Enum<lin,const> *



* assuming sTriangle

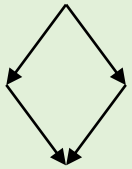
Examples: Full CQs



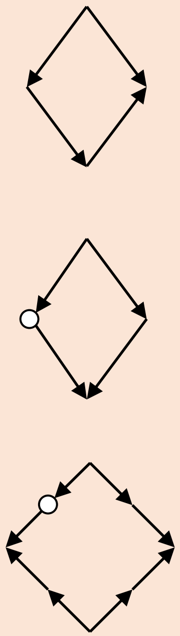
Unlike the self-join-free case,
may affect the complexity:

- reordering variables inside an atom
- introducing unary atoms

∈ Enum<lin,const>



∉ Enum<lin,const> *



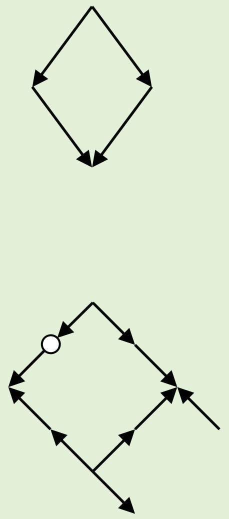
* assuming sTriangle

Examples: Full CQs

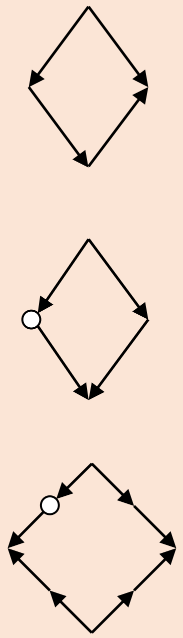
Unlike the self-join-free case,
may affect the complexity:

- reordering variables inside an atom
- introducing unary atoms
- introducing ‘spikes’

∈ Enum<lin,const>

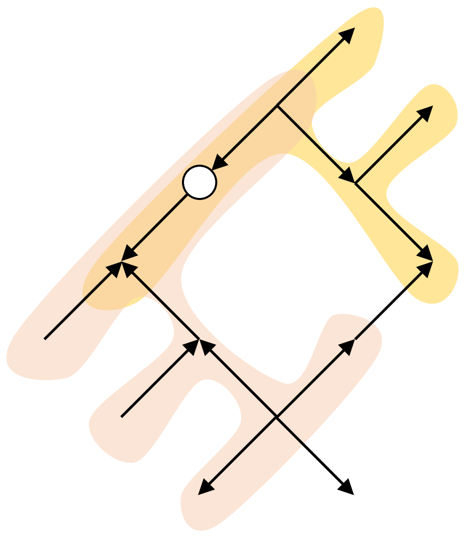


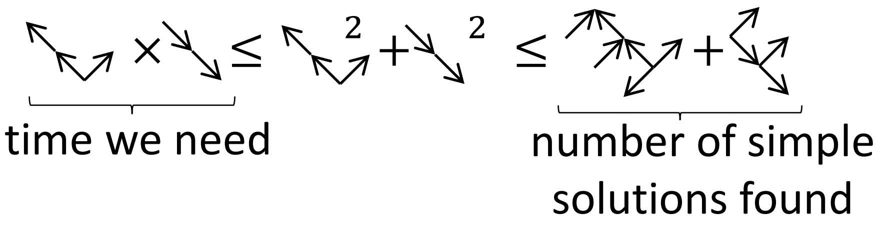
∉ Enum<lin,const> *



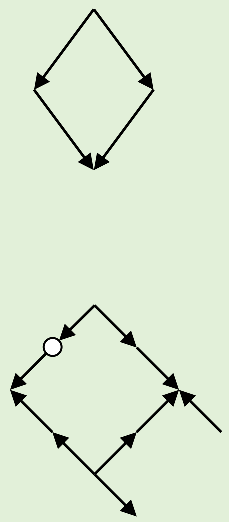
* assuming sTriangle

Examples: Full CQs

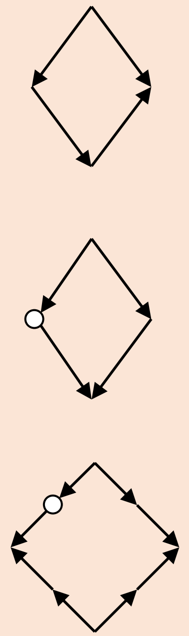




∈ Enum<lin,const>

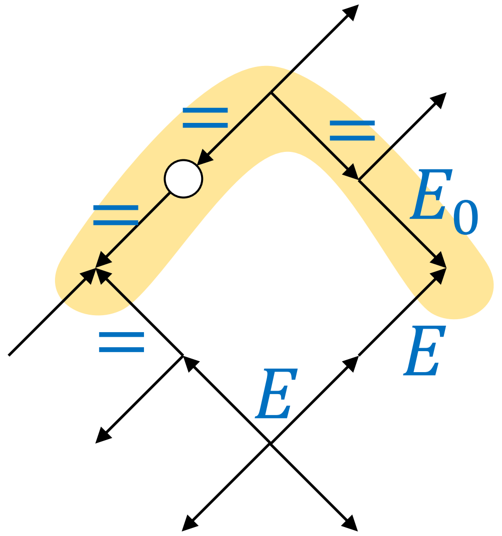


∉ Enum<lin,const> *



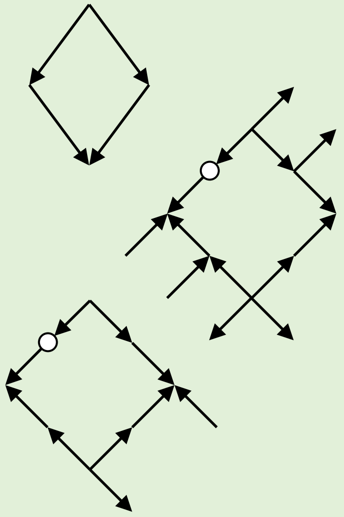
* assuming sTriangle

Examples: Full CQs

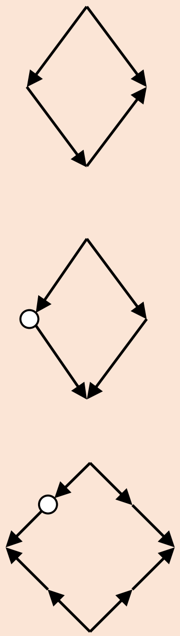


non-triangle solutions $\leq$ $|E_0|^2$

∈ Enum<lin,const>



∉ Enum<lin,const> *



* assuming sTriangle

Vertex-Unbalanced Triangle Detection

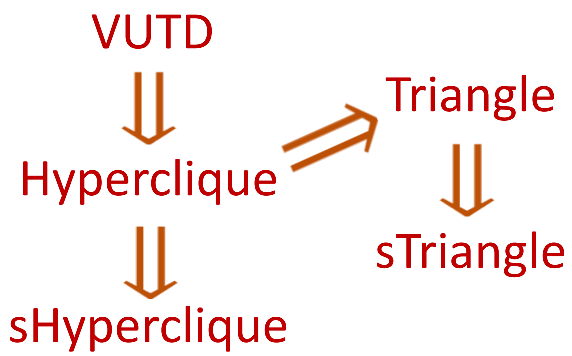
- An α-unbalanced tripartite graph has vertex sets

$$|V_1| = n \text{ and } |V_2| = |V_3| = \Theta(n^\alpha)$$

- Hypothesis: $\forall$ constant $\alpha \in (0,1]$, it is not possible to test the existence of a triangle in an α-unbalanced tripartite graph in time $O(n^{1+\alpha})$.

Remark: this hypothesis is also connected to UCQs [Bringmann, **C**; 22]

Hypotheses



sTriangle: The existence of a triangle in an undirected graph with m edges cannot be decided in time $O(m)$

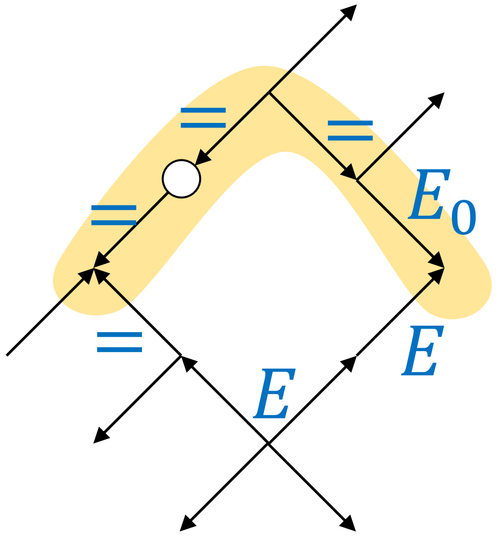
Triangle: The existence of a triangle in an undirected graph with n nodes cannot be decided in time $O(n^2)$

VUTD (Vertex-Unbalanced Triangle Detection) [Bringmann, **C**; 22] :
$\forall \alpha \in (0,1]$ the existence of a triangle in a tripartite graph
with $|V_1| = n$ and $|V_2| = |V_3| = \Theta(n^\alpha)$ cannot be decided in time $O(n^{1+\alpha})$

sHyperclique: $\forall k \geq 3$ the existence of a k-hyperclique in a $(k-1)$-uniform hypergraph with m edges cannot be decided in time $O(m)$

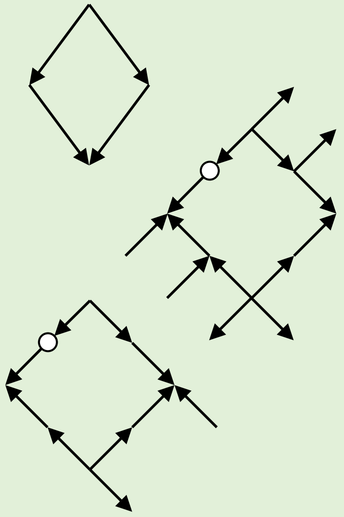
Hyperclique: $\forall k \geq 3$ the existence of a k-hyperclique in a $(k-1)$-uniform hypergraph with n nodes cannot be decided in time $O(n^{k-1})$

Examples: Full CQs

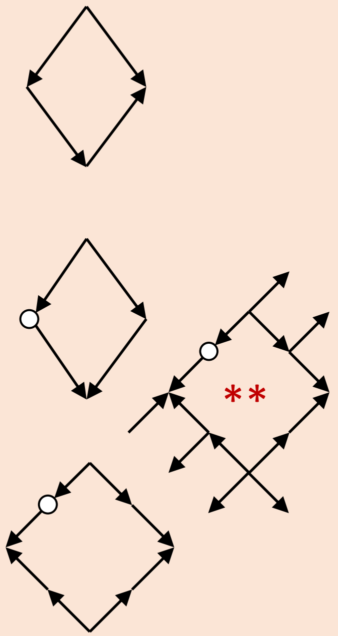


\# non-triangle solutions $\leq$ $|E_0|^2$

$\in$ Enum<lin,const>



$\notin$ Enum<lin,const> *

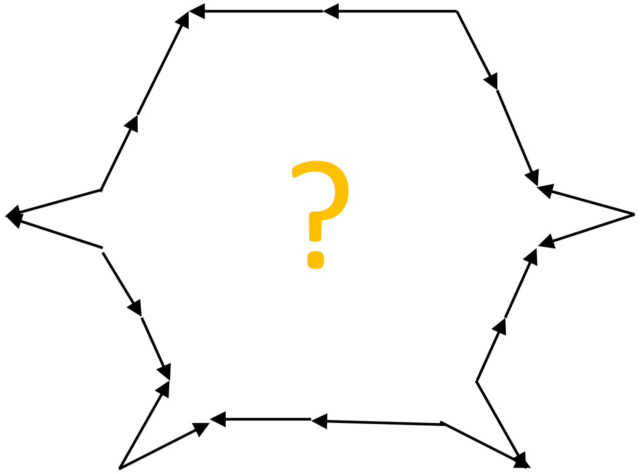


* assuming sTriangle

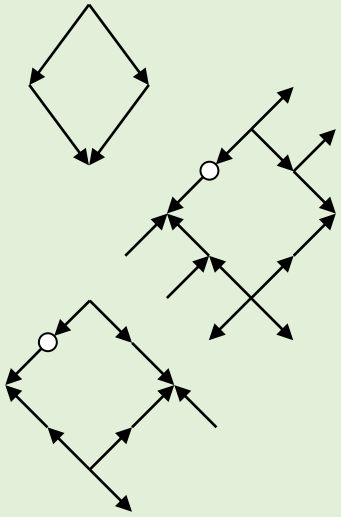
** assuming VUTD

Examples: Full CQs

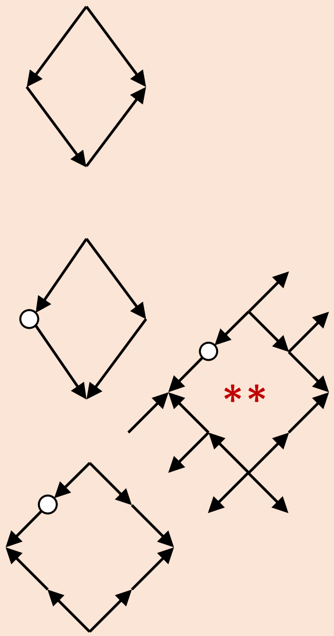
[C Segoufin PODS’2023]



∈ Enum<lin,const>



∉ Enum<lin,const> *



* assuming sTriangle
** assuming VUTD

Plan

- Enumeration
  - Join queries
    - Self-joins
  - **Conjunctive queries**
  - Unions of conjunctive queries
- Other Evaluation Tasks
  - The tasks
  - Known complexity results

Example: Query

Problem

Description	Room
Moisture	5/129
Broken ceiling	Cafeteria
Missing board	5/127

Office

Room	Person	Phone
5/127	Nofar	9590
5/127	Nofar	9591
5/128	Florent	6548
5/128	Guillaume	6548
5/129	David	7544
5/129	Akira	7544

Contact

Person	Email
Nofar	nc@lirmm.fr
Florent	ft@lirmm.fr
Guillaume	gpk@lirmm.fr
David	dc@lirmm.fr

$\exists N$

Conjunctive query $\{(E, P, R, D, \cancel{N}) \mid (D, R) \in \text{Problem}, (R, P, N) \in \text{Office}, (P, E) \in \text{Contact}\}$

~~Join query~~: $Q(E, P, R, D, \cancel{N}) \leftarrow \text{Problem}(D, R), \text{Office}(R, P, N), \text{Contact}(P, E)$

Email	Person	Room	Description	~~Phone~~
nc@lirmm.fr	Nofar	5/127	Missing board	~~9590~~
~~nc@lirmm.fr~~	~~Nofar~~	~~5/127~~	~~Missing board~~	~~9591~~
dc@lirmm.fr	David	5/129	Moisture	~~7544~~

Handling Projection

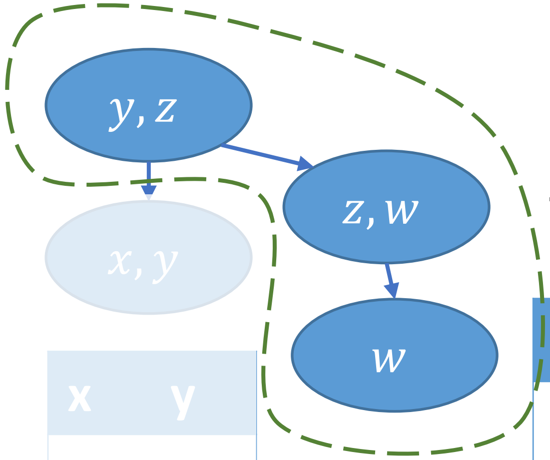
works

$$Q_1(y,z,w) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w), R_4(w)$$

Solution:

1. Find a join tree
2. Remove dangling tuples
3. **Ignore existential variables**
4. Join

y	z
y1	z1
y1	z2
y2	z2
~~y3~~	~~z1~~
~~y4~~	~~z3~~



z	w
z1	w1
z2	w2
~~z3~~	~~w3~~

x	y
x1	y1
x1	y2
x2	y2
~~x2~~	~~y4~~

w
w1
w2

~~x~~	y	z	w
~~x1~~	y1	z1	w1
~~x1~~	y1	z2	w2
~~x1~~	y2	z2	w2
~~x2~~	y2	z2	w2

y	z	w
y1	z1	w1
y1	z2	w2
y2	z2	w2

Handling Projection

works $Q_1(y,z,w) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w), R_4(w)$

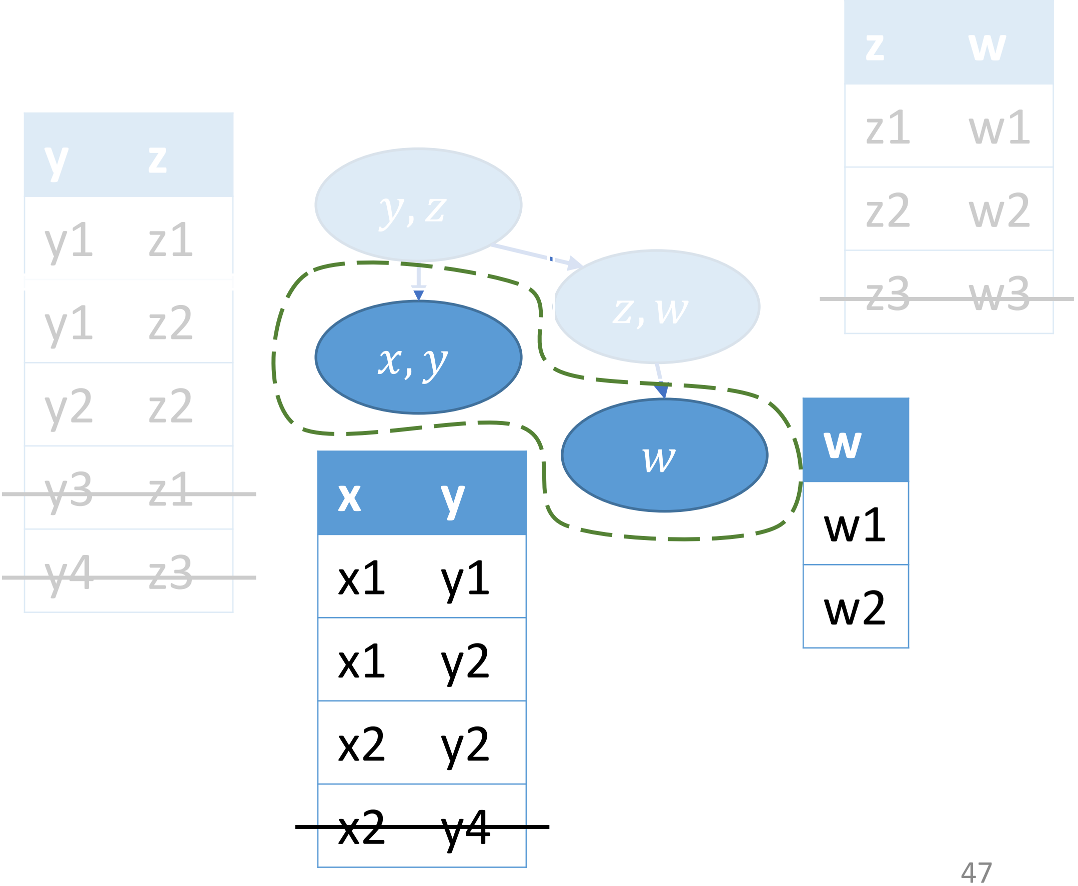
doesn't work $Q_2(x,y,w) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w), R_4(w)$

Solution:

1. Find a join tree
2. Remove dangling tuples
3. **Ignore existential variables**
4. Join

x	y	z	w
x1	y1	z1	w1
x1	y1	z2	w2
x1	y2	z2	w2
x2	y2	z2	w2

x	y	w
x1	y1	w1
x1	y1	w2
x1	y2	w2
x2	y2	w2



Definitions

[Bagan, Durand, Grandjean; CSL 07]

An acyclic CQ has a graph with:

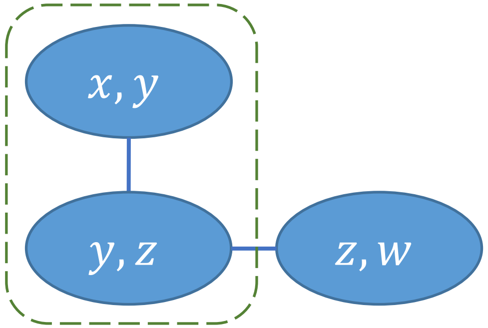
A free-connex CQ also requires:

1. a node for every atom
   possibly also subsets
2. tree
3. for every variable:
   the nodes containing it form a subtree

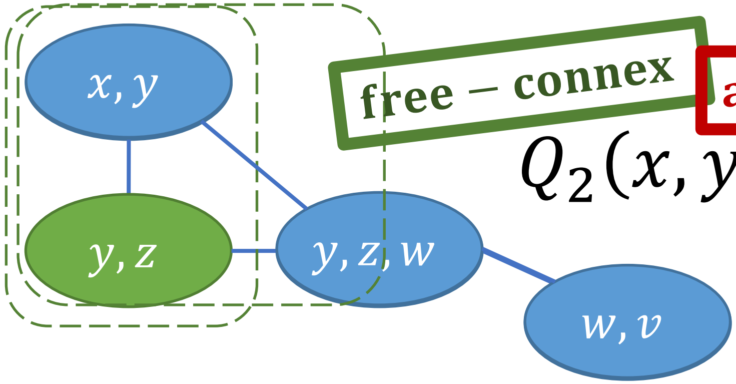
free – connex

acyclic

$$Q_1(x,y,z) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w)$$



4. a subtree with exactly the free variables



free – connex

acyclic

$$Q_2(x,y,z) \leftarrow R_1(x,y), R_2(y,z,w), R_3(w,v)$$

Eliminating Projection

given a **free-connex acyclic** CQ and an input DB,
we can construct **in linear time**
an equivalent **full acyclic** CQ and input DB

Dichotomy for CQs

[BaganDurandGrandjean CSL'2007]
[Brault-Baron 2013]

- Given a conjunctive query Q,

If Q is acyclic free-connex, Q ∈ Enum<lin,const>

If Q is acyclic not free-connex, Q ∉ Enum<lin,const>*

If Q is cyclic, Q ∉ Enum<lin,const>**

* no self-joins, assuming sBMM

** no self-joins, assuming sHyperclique

Lower Bound: acyclic non-free-connex

[Bagan, Durand, Grandjean; CSL 07]

Assumption: Boolean $n \times n$ matrices cannot be multiplied in time $O(n^2)$

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}$$

Indices of ones

$\boldsymbol{R_1}$

R	C
1	1
1	2
2	2

$\boldsymbol{R_2}$

R	C
1	2
2	2

$\boldsymbol{Q}$

R	C
1	2
2	2

Acyclic non-free-connex: $Q(x,z) \leftarrow R_1(x,y), R_2(y,z)$

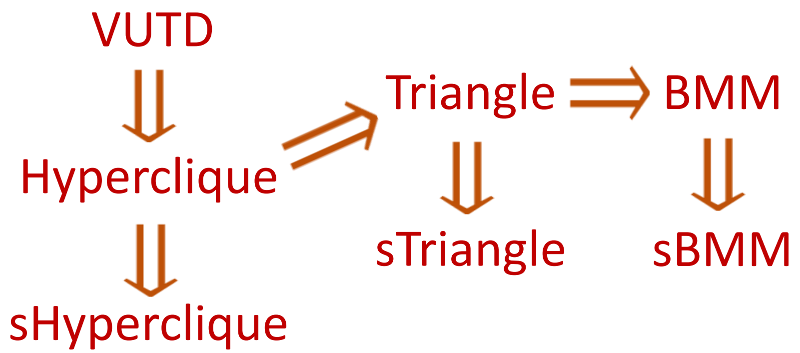
$O(n^2)$ preprocessing + $O(1)$ delay = $O(n^2)$ total $\implies$ not possible

Intractability cause: free-path $x - y - z$

Hypotheses

sBMM: Boolean matrices cannot be multiplied in linear time in the number of the 1 entries

BMM: Boolean $n \times n$ matrices cannot be multiplied in time $O(n^2)$



sTriangle: The existence of a triangle in an undirected graph with m edges cannot be decided in time $O(m)$

Triangle: The existence of a triangle in an undirected graph with n nodes cannot be decided in time $O(n^2)$

VUTD (Vertex-Unbalanced Triangle Detection):
$\forall \alpha \in (0,1]$ the existence of a triangle in a tripartite graph with $|V_1| = n$ and $|V_2| = |V_3| = \Theta(n^\alpha)$ cannot be decided in time $O(n^{1+\alpha})$

sHyperclique: $\forall k \geq 3$ the existence of a k-hyperclique in a $(k-1)$-uniform hypergraph with m edges cannot be decided in time $O(m)$

Hyperclique: $\forall k \geq 3$ the existence of a k-hyperclique in a $(k-1)$-uniform hypergraph with n nodes cannot be decided in time $O(n^{k-1})$

Plan

- Enumeration
  - Join queries
    - Self-joins
  - Conjunctive queries
  - **Unions of conjunctive queries**
- Other Evaluation Tasks
  - The tasks
  - Known complexity results

Example: Union of CQs

Posts

Amazing vacation	Alice
Amazing vacation	Bob
Angry post	Bob

Followers

Alice	Bob
Bob	Carol

Friends

Bob	Carol
Carol	Dafni

$$Q_1(post, p2, p3) \leftarrow Posts(post, p1), Followers\ (p1, p2), Friends\ (p2, p3)$$

$$\cup$$

$$Q_2(post, p1, p2) \leftarrow Posts(post, p1), Followers\ (p1, p2)$$

Post	Person 1	Person 2	
Amazing vacation	Bob	Carol	due to Q_1 or Q_2
Amazing vacation	Alice	Bob	due to Q_1
Angry post	Carol	Dafni	due to Q_2
Angry post	Bob	Carol	due to Q_1

Cases for UCQs

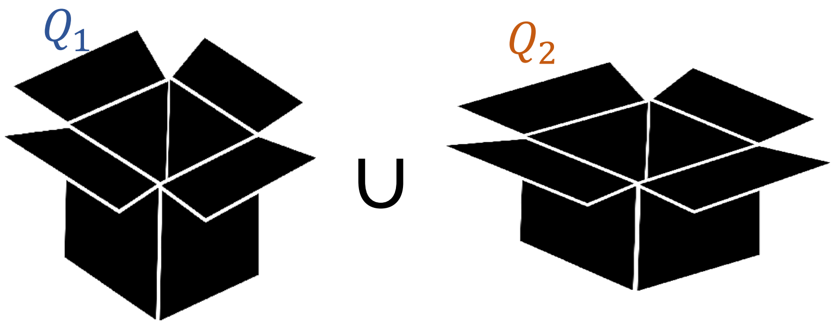
All CQs are Easy

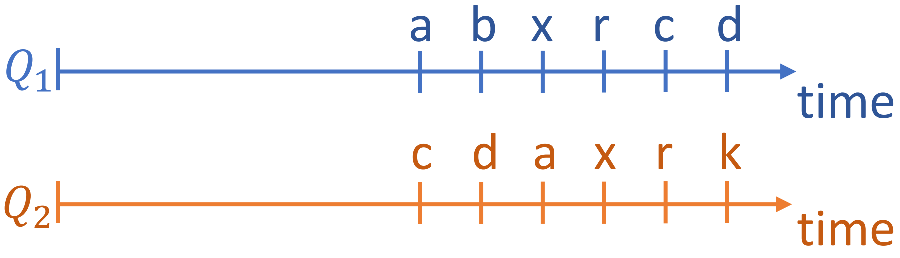
always easy

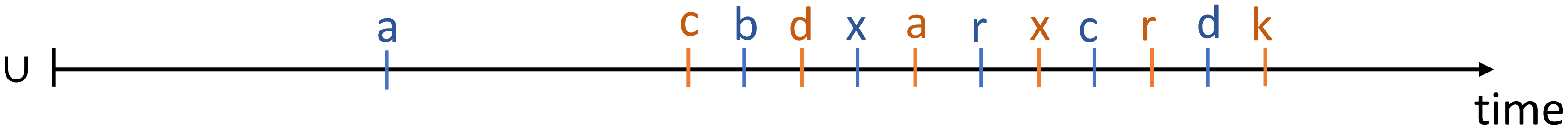
Some Easy, Some Hard

All CQs are Hard

Easy ∪ Easy Is Always Easy







Generated (lookup):

a b c d

x

Queue:

a
c
b
d
x

Output:

…

Enumeration: union of easy CQs

[Durand, Strozecki; CSL 11]

```
while A.hasNext():
    a = A.next()
    if a ∉ B:
        print a
    else:
        print B.next()
while B.hasNext():
    print B.next()
```

prints $A \setminus B$ → print a

prints B → print B.next() (both)

$A \setminus B$ and B are a partition of $A \cup B$

Cases for UCQs

[C, Kröll; PODS 19]

All CQs are Easy

always easy

Some Easy, Some Hard

sometimes hard

sometimes easy

All CQs are Hard

Lower Bound: acyclic non-free-connex

[Bagan, Durand, Grandjean; CSL 07]

Assumption: Boolean $n \times n$ matrices cannot be multiplied in time $O(n^2)$

$$\begin{pmatrix}1 & 1\\0 & 1\end{pmatrix}\begin{pmatrix}0 & 1\\0 & 1\end{pmatrix} = \begin{pmatrix}0 & 1\\0 & 1\end{pmatrix}$$

Indices of ones

R_1

R	C
1	1
1	2
2	2

R_2

R	C
1	2
2	2

Q

R	C
1	2
2	2

Acyclic non-free-connex: $Q(x,z) \leftarrow R_1(x,y), R_2(y,z)$

$O(n^2)$ preprocessing + $O(1)$ delay = $O(n^2)$ total $\Longrightarrow$ not possible

Intractability cause: free-path $x - y - z$

Why this isn't hard

not free connex

hard part

$$Q_1(x,z,w) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w)$$

$$\cup$$

$$Q_2(a,b,c) \leftarrow R_1(a,b), R_2(b,c)$$

Q_1		
1	2	⊥
2	2	⊥

Q_2		
1	1	2
1	2	2
2	2	2

$O(n^3)$ solutions:
The computation does not contradict the assumption

R_1	
1	1
1	2
2	2

R_2	
1	2
2	2

R_3	
2	⊥

The hardness results do not hold within a union

Example: Tractable Union

acyclic non free-connex

hard part

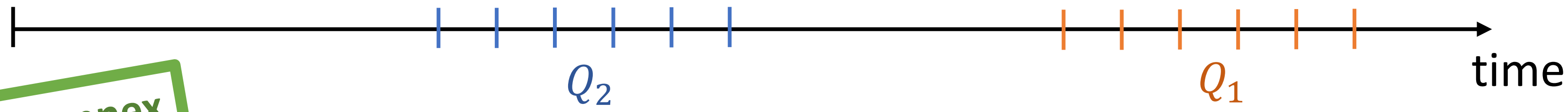
$$Q_1(x,z,w) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w)$$

Body-homomorphism ∪

free-connex

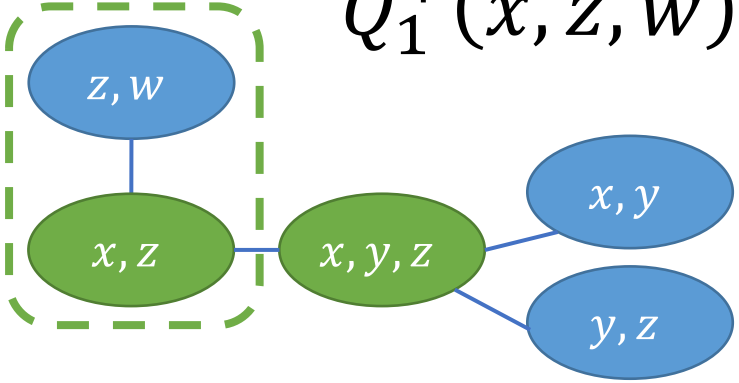
$$Q_2(a,b,c) \leftarrow R_1(a,b), R_2(b,c)$$

∈ Enum<lin,const>



free-connex

$$Q_1^+(x,z,w) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w), Q_2(x,y,z)$$



	Step	Output	Side Effect
1	Solve Q_2	Q_2	Find $R_1 \bowtie R_2$
2	Solve Q_1^+	Q_1	

Cheater's Lemma

[C, Kröll; PODS 19]

If an enumeration problem can be solved with:

- Usually constant delay
- Almost no duplicates

constant number of linear delay steps

constant number of duplicates per answer

Then*, it is ∈ Enum<lin,const>

Can be solved in: linear preprocessing, constant delay, no duplicates

* using polynomial space

Complexity Measures

[**C**, Kröll; TODS 21]

- (Instance-optimal) linear total time
  - Total time $O(n+N)$

- Linear partial time
  - Time before the ith answer is $O(n+i)$

- Linear preprocessing and constant delay
  - Time before the first answer $O(n)$
  - Time between successive answers $O(1)$

equivalent
assuming we can use polynomial space

n = input size, N = output size

Cases for UCQs

[**C**, Kröll; PODS 19]

All CQs are Easy
- always easy

Some Easy, Some Hard
- sometimes hard
- sometimes easy

All CQs are Hard
- sometimes hard
- sometimes easy

Hard ∪ Hard = Easy

[**C**, Kröll; PODS 19]

- Example: CQs with **isomorphic bodies**.

hard part

$$Q_1(x,z,w,u) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w), R_4(w,u)$$
$$Q_2(x,y,z,u) \leftarrow R_1(x,y), R_2(y,z), R_3(z,w), R_4(w,u)$$

hard part

Body-homomorphisms

	Step	Output	Side Effect
1	Solve ${Q_1}'$	$\subseteq Q_1$	Find $R_3 \bowtie R_4$
2	Solve Q_2^+	Q_2	Find $R_1 \bowtie R_2$
3	Solve Q_1^+	Q_1	

Dichotomy for Unions of 2 CQs

[**C**, Bringmann; 22]

- Given a union of two conjunctive queries Q,

If Q has an acyclic free-connex union extension,
Q ∈ Enum<lin,const>

Otherwise, Q ∉ Enum<lin,const>*

* no self-joins, assuming VUTD

There exists a family of UCQs with no free-connex union extensions s.t.
VUTD hypothesis holds ⇔ no query of the family is in Enum<lin,const>

Example: Intractable Union (Assuming VUTD)

[C, Bringmann; 22]

acyclic non free-connex

hard part

$$Q_1(x,y,w) \leftarrow R_1(x,z), R_2(z,y), R_3(y,w)$$

$\cup$ Body-homomorphism

free-connex

$$Q_2(x,y,w) \leftarrow R_1(x,t_1), R_2(t_2,y), R_3(w,t_3)$$

<u>VUTD (Vertex-Unbalanced Triangle Detection)</u> :
$\forall \alpha \in (0,1]$ the existence of a triangle in a tripartite graph
with $|V_1| = n$ and $|V_2| = |V_3| = \Theta(n^\alpha)$ cannot be decided in time $O(n^{1+\alpha})$

- Q_2 can't help Q_1 : it doesn't provide z
- Construction: assigns large vertex set to z, small vertex sets to x and y, constant $\perp$ to w
- Answers:
  - Ignore answers to Q_2 (there are $O(n^{2\alpha})$ such answers)
  - Check whether answers to Q_1 form an edge (if so, triangle detected)

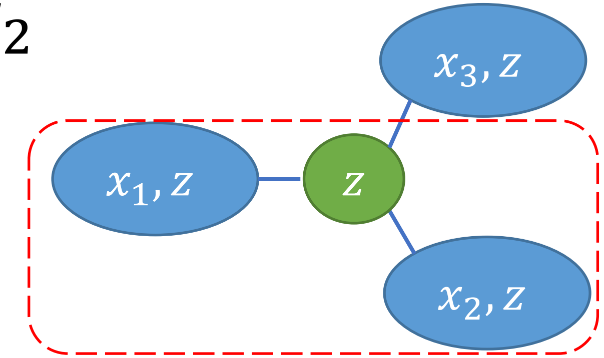
Beyond 2 CQs: almost open problem

- Example:

$$Q_1(x_1,x_2,x_3), Q_2(x_1,x_2,z), Q_3(x_1,x_3,z), Q_4(x_2,x_3,z) \leftarrow R_1(x_1,z), R_2(x_2,z), R_3(x_3,z)$$

$\notin$ Enum<lin,const>

- Q_1 hard: reduce from matrix multiplication to x_1, z, x_2
- Others easy: free-connex acyclic

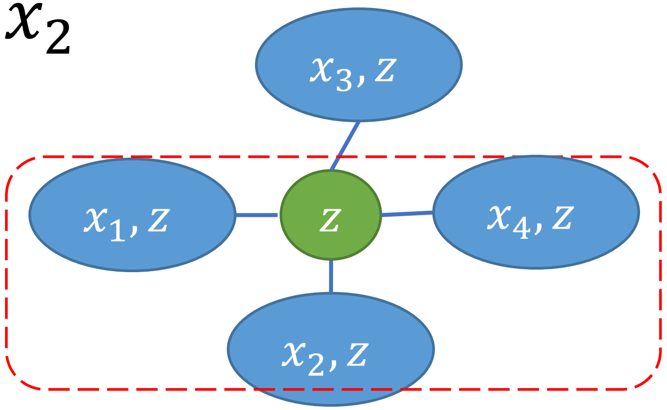


- Cannot use matrix multiplication reduction (others have too many answers)
- Can reduce from 4-clique: detection in $O(n^3)$ time

Beyond 2 CQs: open problem

- Example: $$Q_1(x_1, x_2, x_3, x_4), Q_2(x_1, x_2, x_3, z), Q_3(x_1, x_2, x_4, z),$$ $$Q_4(x_1, x_3, x_4, z), Q_5(x_2, x_3, x_4, z) \leftarrow$$ $$R_1(x_1, z), R_2(x_2, z), R_3(x_3, z), R_4(x_4, z)$$

- Q_1 hard: reduce from matrix multiplication to x_1, z, x_2
- Others easy: free-connex

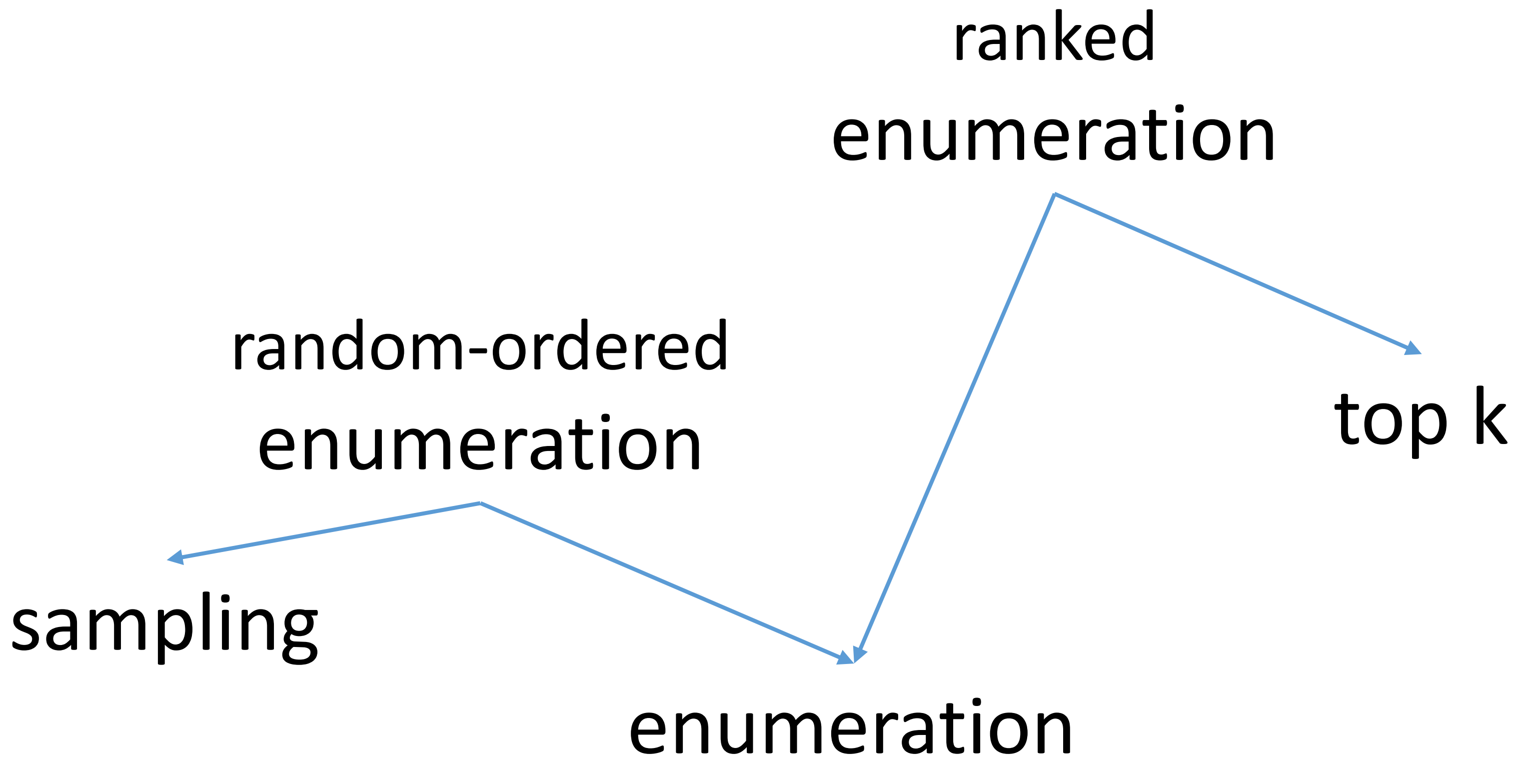


- Cannot use matrix multiplication reduction (others have too many answers)
- Cannot reduce from 5-clique (it is not a valid assumption that we can't solve the $(k+1)$-clique problem in time $O(n^k)$ for large k values).

Plan

- Enumeration
  - Join queries
    - Self-joins
  - Conjunctive queries
  - Unions of conjunctive queries
- **Other Evaluation Tasks**
  - **The tasks**
  - Known complexity results

Overview of Tasks



Quantile Computation via Ranked Access

Employees

Name	Role	Address
Jack	Junior dev	Boston
Jill	Senior dev	Brookline
Joanna	Senior dev	Braintree

Remuneration

Period	Role	Salary
11/2020	Junior dev	4000
11/2020	Senior dev	4500
12/2020	Junior dev	7000
12/2020	Senior dev	7100

Travel

Address	Cost
Boston	50
Brookline	100
Braintree	200

- What is the median monthly cost of an employee?
  - Solution 1:
    join, sort, access the middle
  - Solution 2:
    count, ranked enumeration until the middle
  - Solution 3:
    count, ranked access to the middle

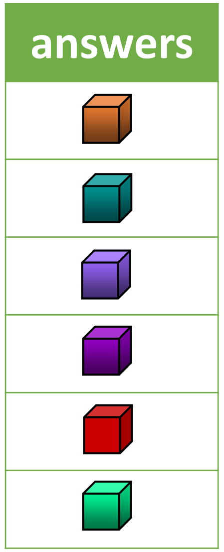
Join Results

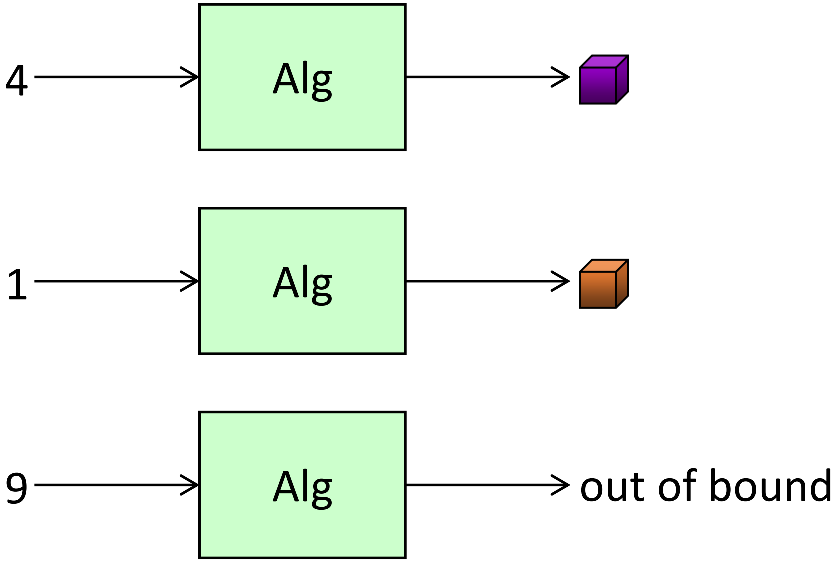
Name	Role	Address	Period	Salary	Cost	
Jack	Junior dev	Boston	11/2020	4000	50	
Jill	Senior dev	Brookline	11/2020	4500	100	
Joanna	Senior dev	Braintree	11/2020	4500	200	← 3rd
Jack	Junior dev	Boston	12/2020	7000	50	
Jill	Senior dev	Brookline	12/2020	7100	100	
Joanna	Senior dev	Braintree	12/2020	7100	200	

Count = 6

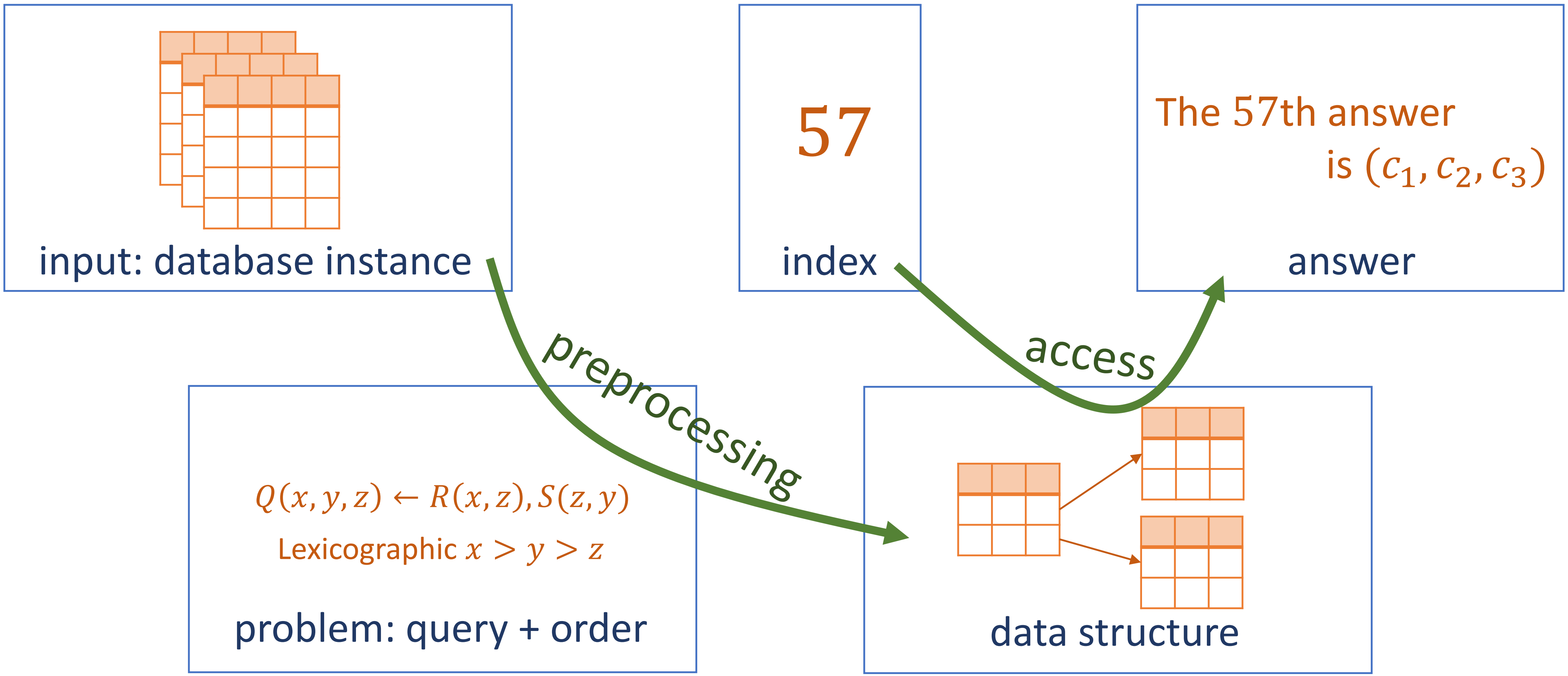
Definition: Access Tasks

- Given i, returns the ith answer or “out of bound”.
- Ranked Access: user-specified order

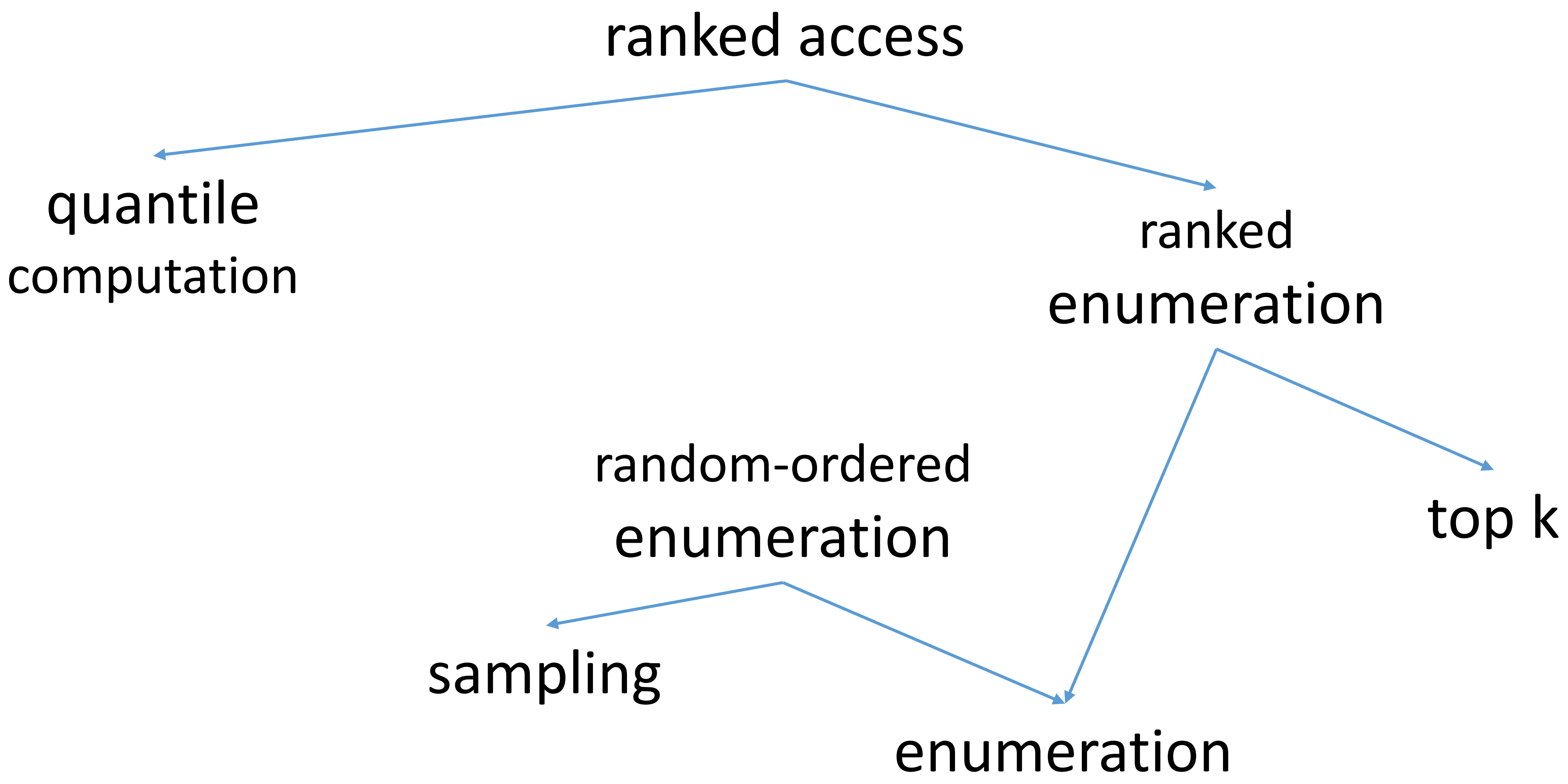




Goal: efficient ranked access

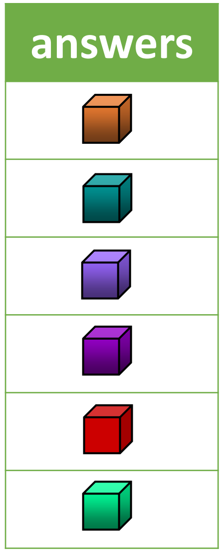


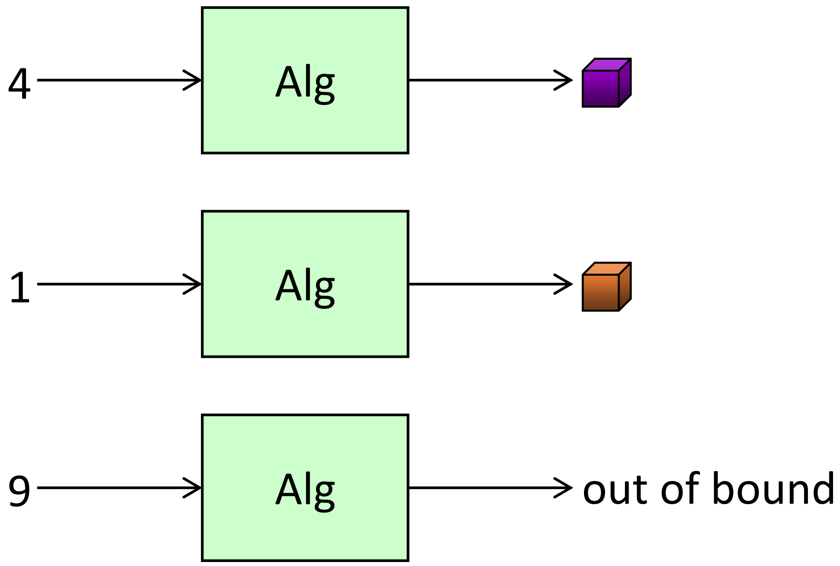
Overview of Tasks



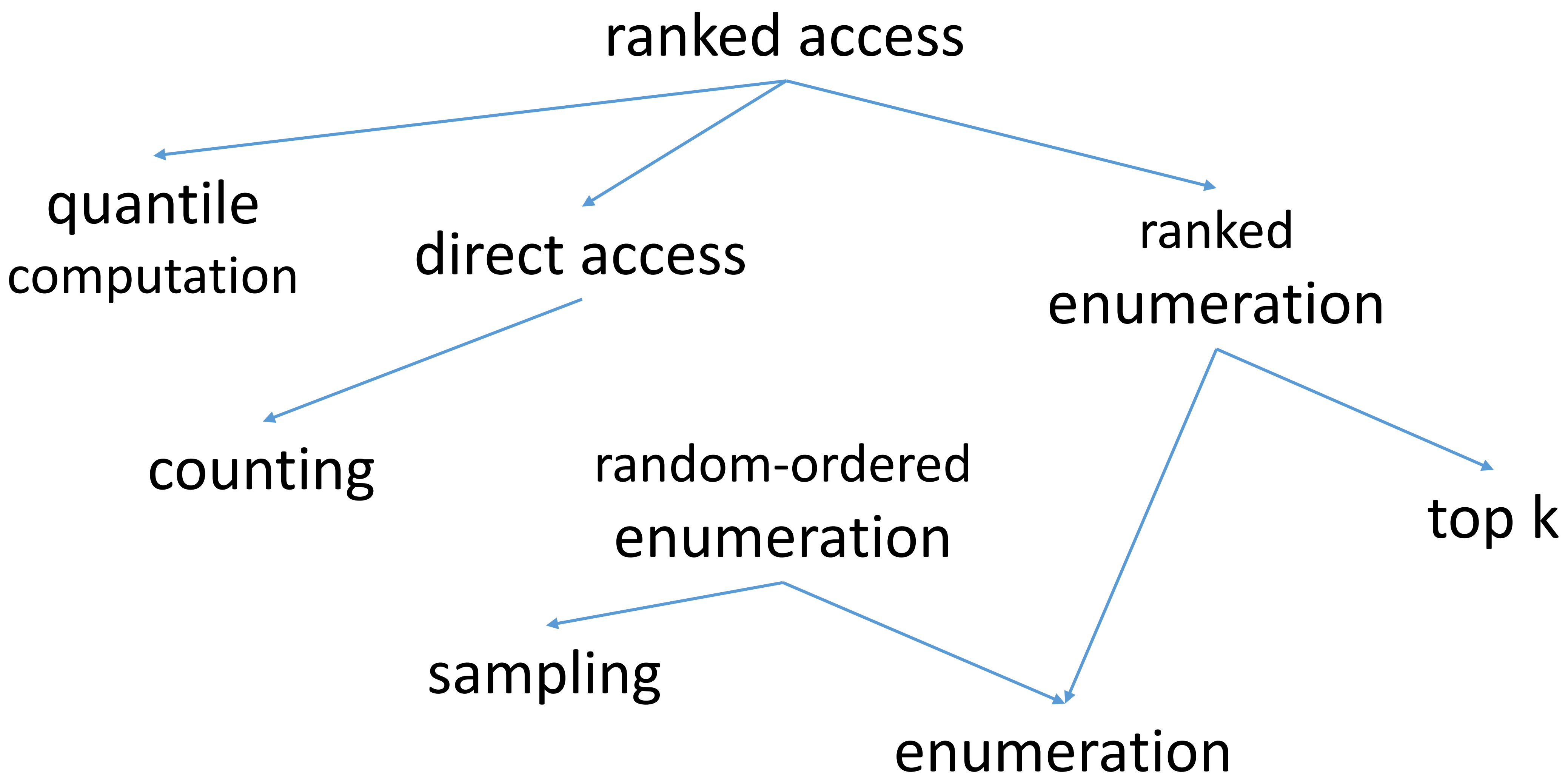
Definition: Access Tasks

- Given i, returns the i^{th} answer or “out of bound”.
- Ranked Access: user-specified order
- Direct Access: no constraints on the ordering used





Overview of Tasks

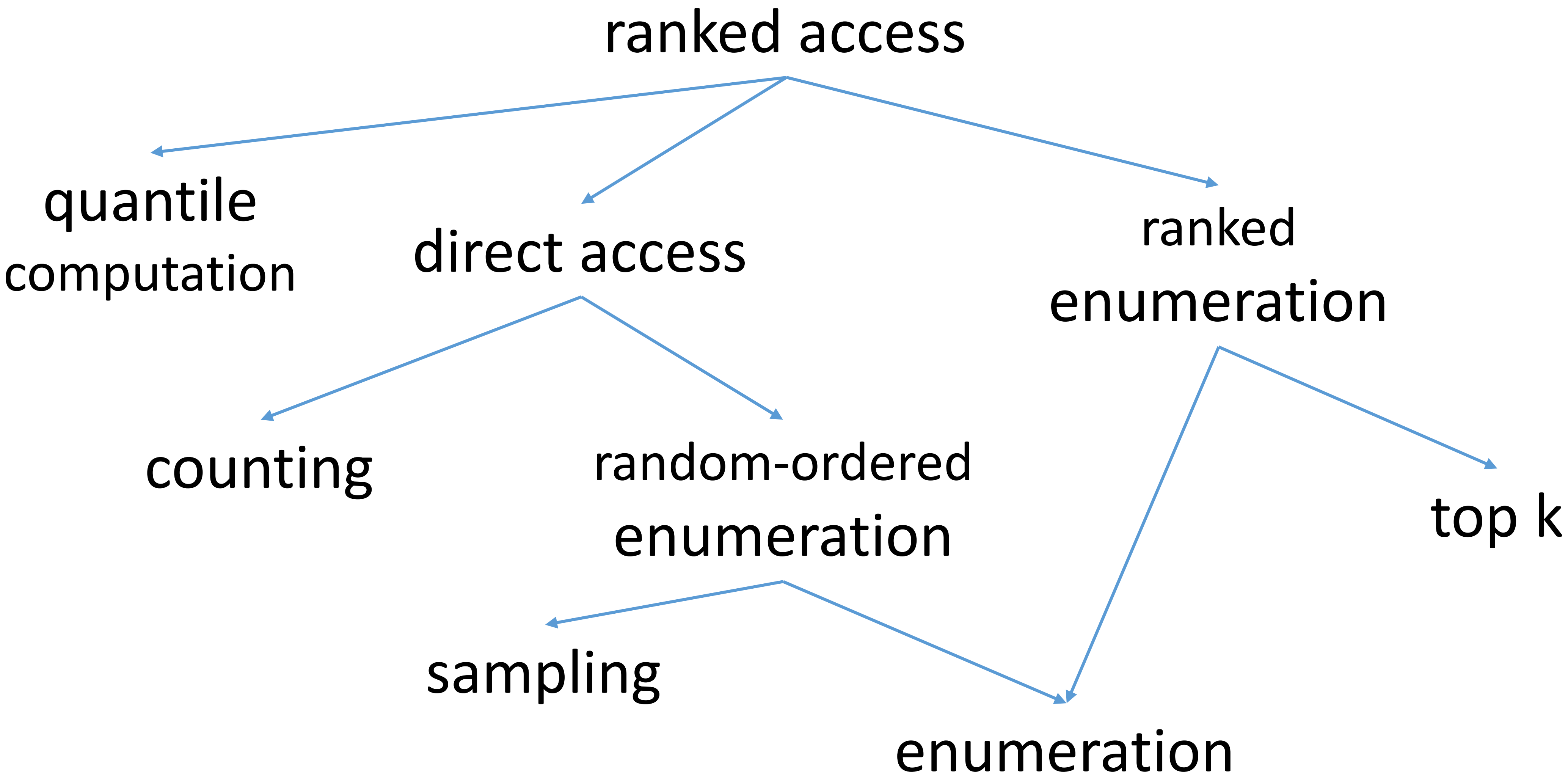


* with log time per answer after linear preprocessing

Counting via Direct Access

- Assumption: the number of answers is bounded by a polynomial
- Direct Access returns “out of bound” if needed
  - Allows checking if $|answers| > k$
- Binary search for $|answers|$
  - Requires $O(\log(|answers|))$ calls for Direct Access
  - If $|answers|$ is polynomial, $\log(|answers|) = O(\log(input))$
  - This takes $O(\log(input) \cdot cost(access))$ time

Overview of Tasks



* with log time per answer after linear preprocessing

Random-Ordered Enumeration via Direct Access

[**C**, Zeevi, Berkholz, Kimelfeld, Schweikardt; PODS 20]

1) Find the number N of answers

6

2) Find a random permutation of 1,…,N

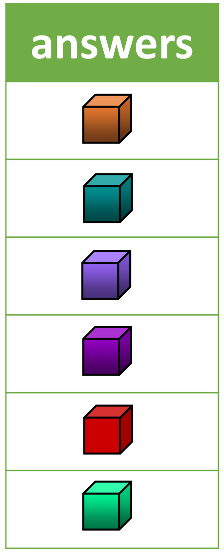
5 6 4 2 1 3

3) Direct access to answers

Direct Access
+
Binary Search

Modified Fisher-Yates Shuffle

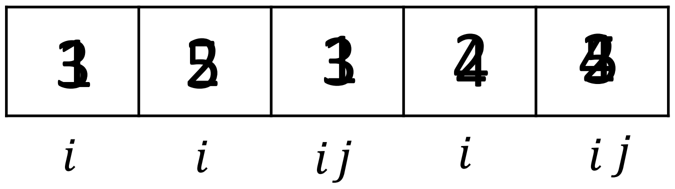
Direct Access



Fisher-Yates Shuffle

[Durstenfeld 1964]

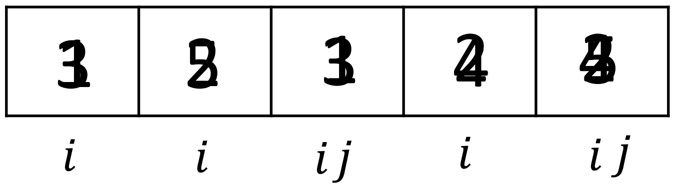
Place $1, \ldots, n$ in array
For i in $1, \ldots, n$:
  choose j randomly from $\{i, \ldots, n\}$
  replace i and j



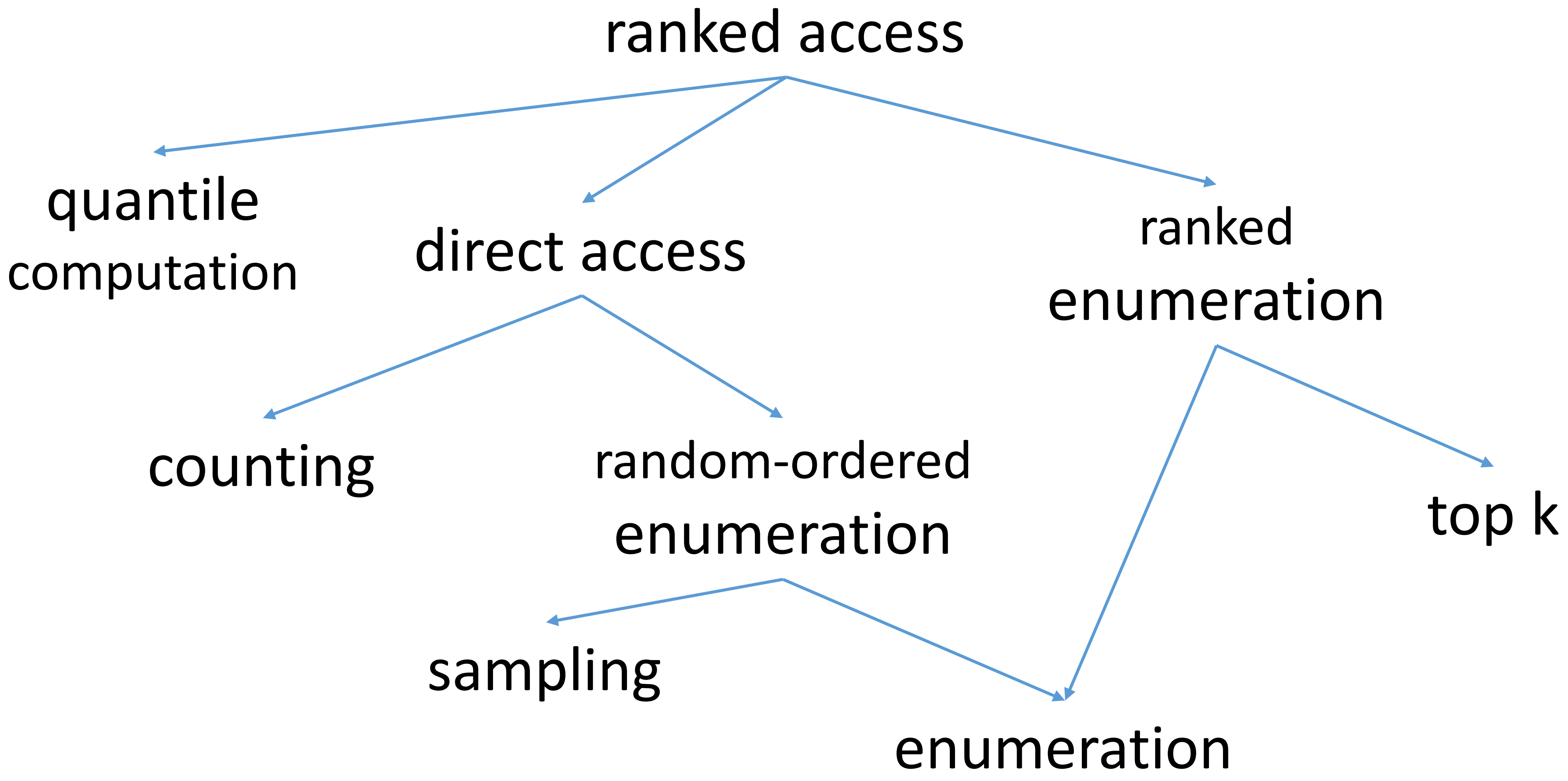
Fisher-Yates Shuffle

Constant delay variant:

place $1, \dots, n$ in array (lazy initialization)
for i in $1, \dots, n$:
    choose j randomly from $\{i, \dots, n\}$
    replace i and j
    print $a[i]$



Overview of Tasks



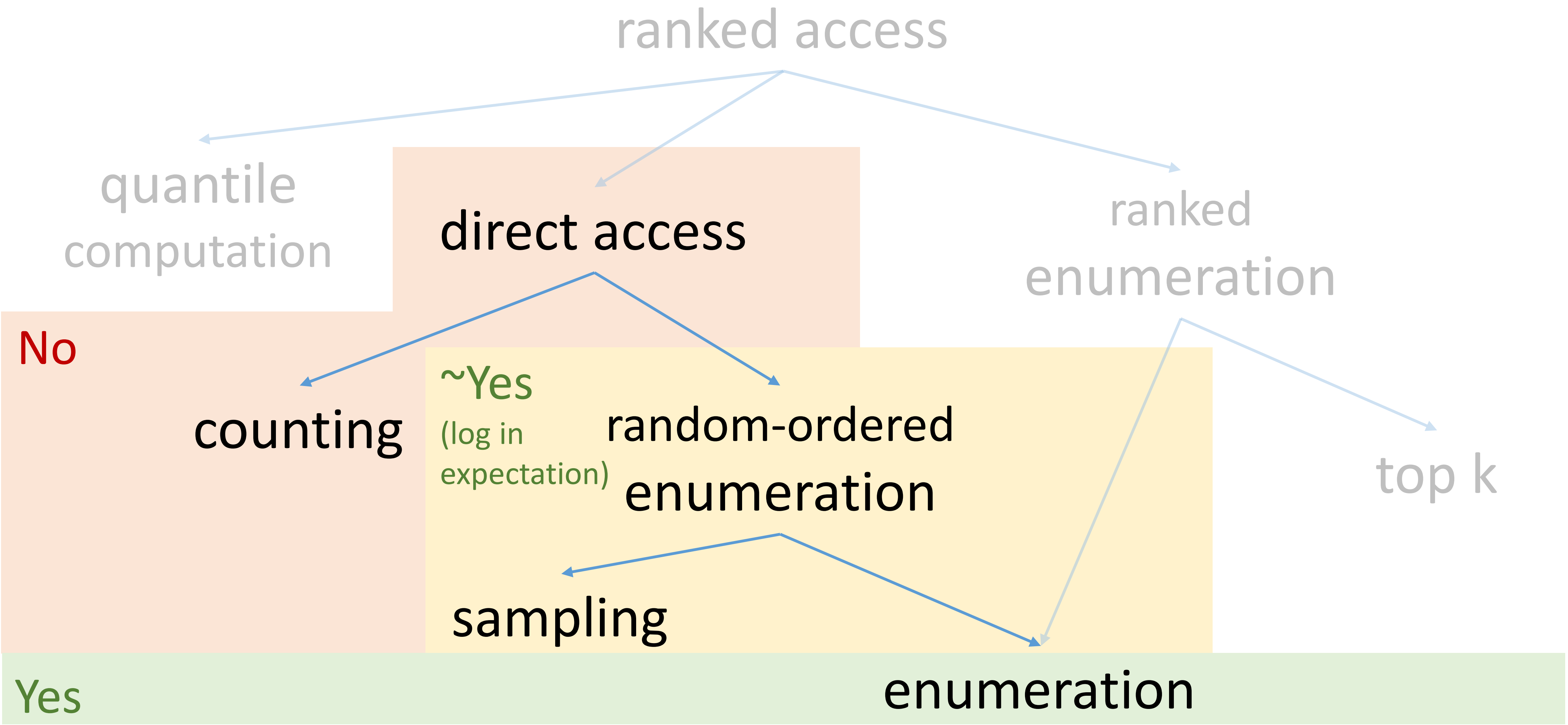
* with log time per answer after linear preprocessing

Plan

- Enumeration
  - Join queries
    - Self-joins
  - Conjunctive queries
  - Unions of conjunctive queries
- Other Evaluation Tasks
  - The tasks
  - **Known complexity results**

Can be solved efficiently* for all unions of free-connex CQs?

[**C**, Zeevi, Berkholz, Kimelfeld, Schweikardt; PODS 20]



* with log time per answer after linear preprocessing

Example: Difficult Counting

free-connex acyclic

$$Q_1(x, y, z) \leftarrow R(x, y), S(y, z)$$

$$\cup$$

free-connex acyclic

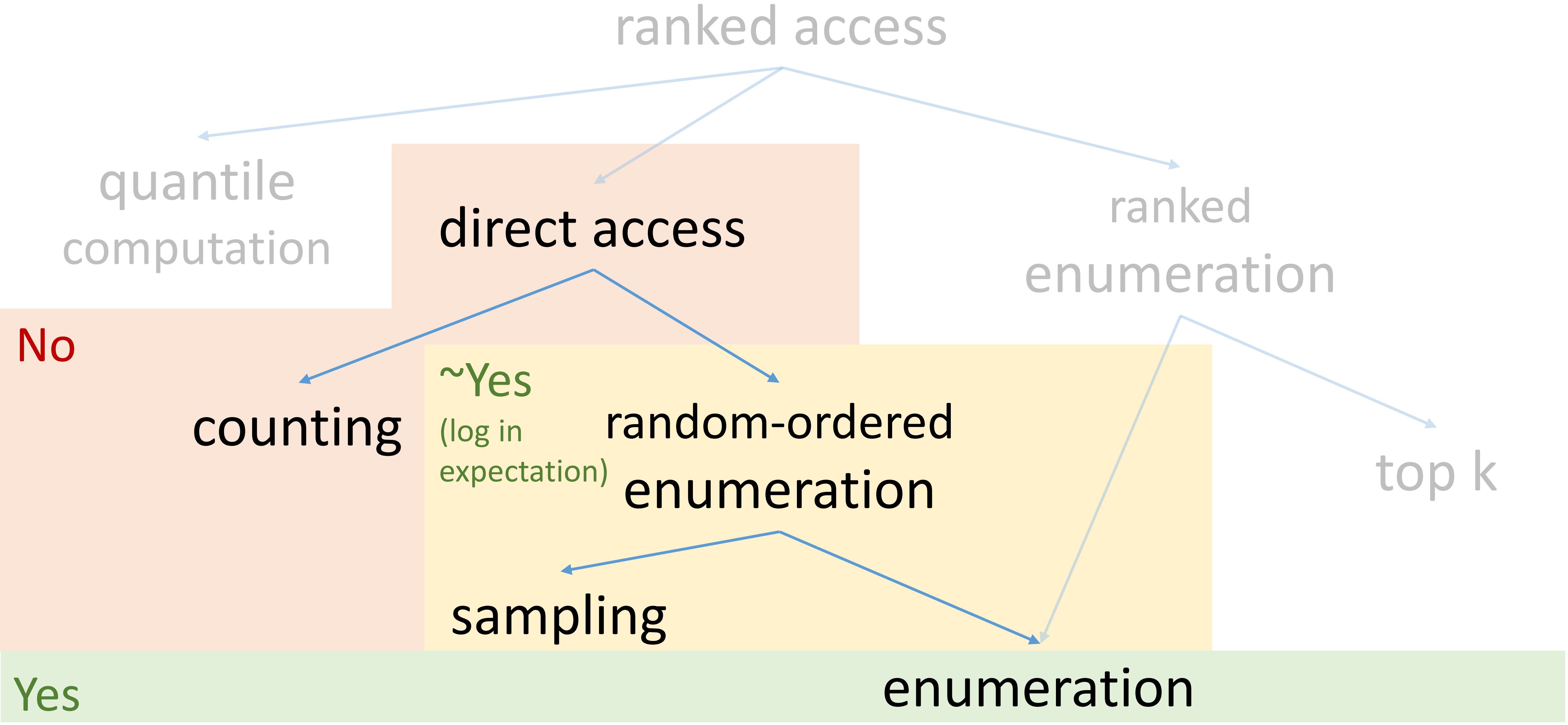
$$Q_2(x, y, z) \leftarrow S(y, z), T(x, z)$$

- $Q_1 \cap Q_2(x, y, z) \leftarrow R(x, y), S(y, z), T(x, z)$ cyclic
  - Cannot determine whether $|Q_1 \cap Q_2| > 0$ in linear time, assuming sTriangle
- $|Q_1 \cap Q_2| = |Q_1| + |Q_2| - |Q_1 \cup Q_2|$

can be computed in linear time

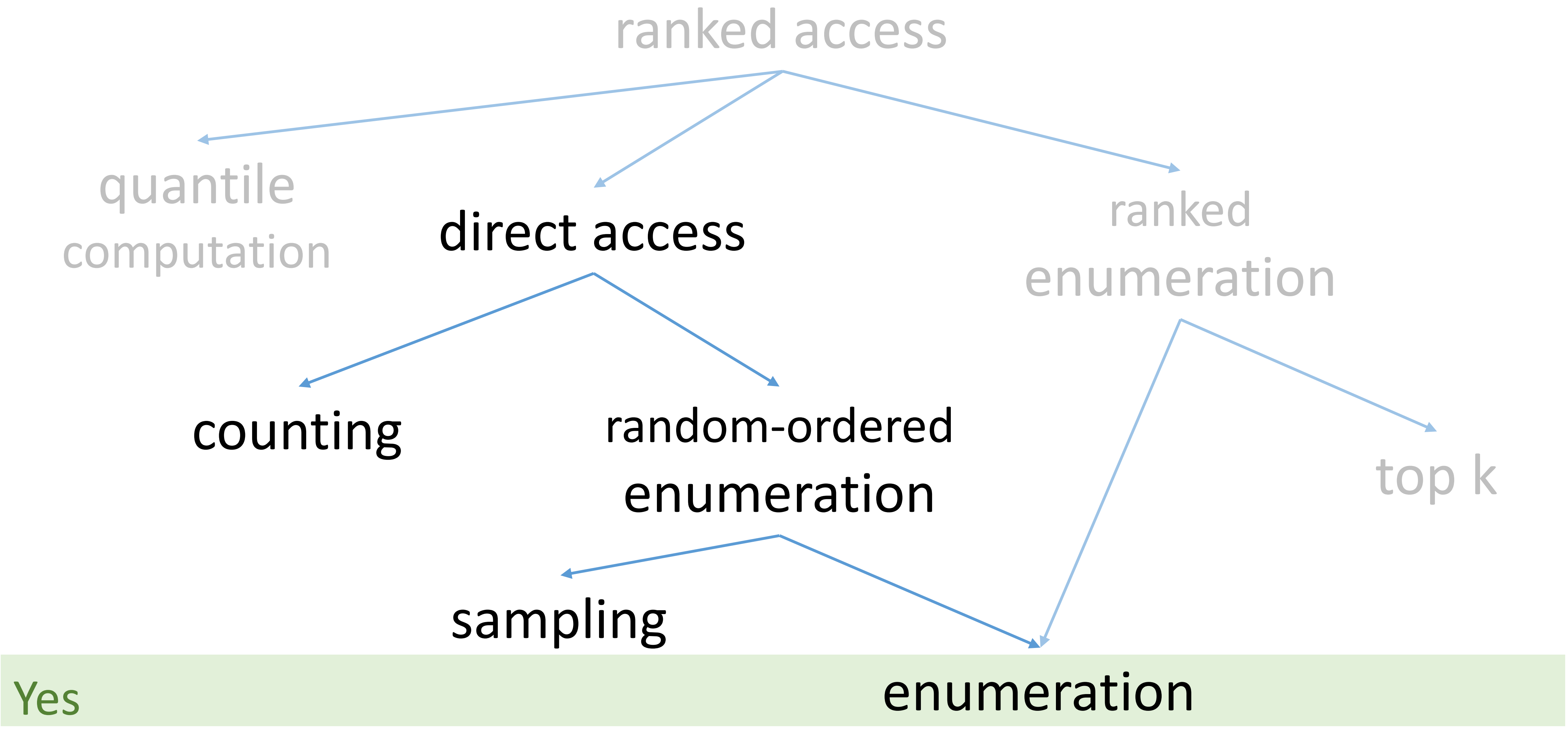
Can be solved efficiently* for all unions of free-connex CQs?

[C, Zeevi, Berkholz, Kimelfeld, Schweikardt; PODS 20]



* with log time per answer after linear preprocessing

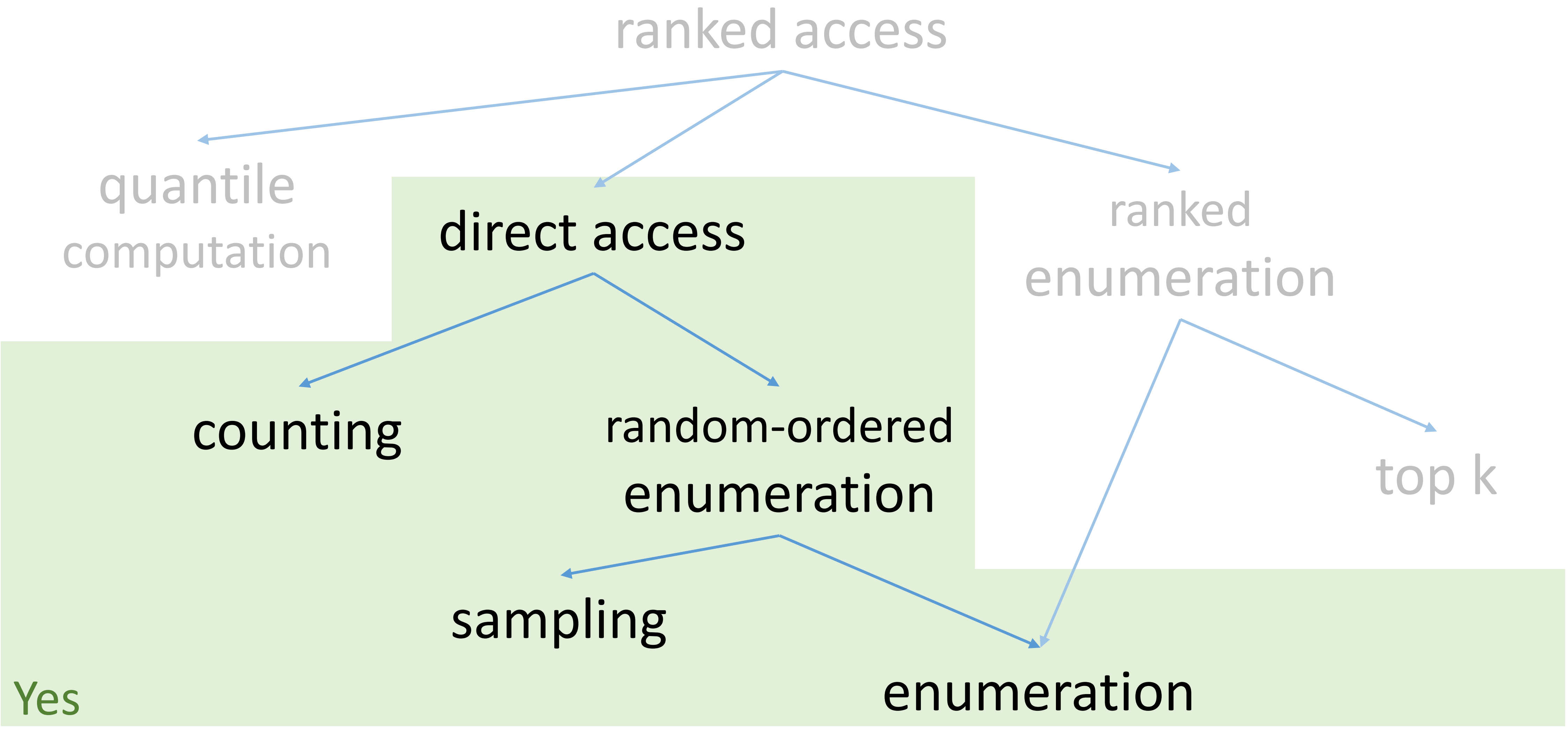
Can be solved efficiently* for all free-connex CQs?



* with log time per answer after linear preprocessing

Can be solved efficiently* for all free-connex CQs?

[Brault-Baron 2013]

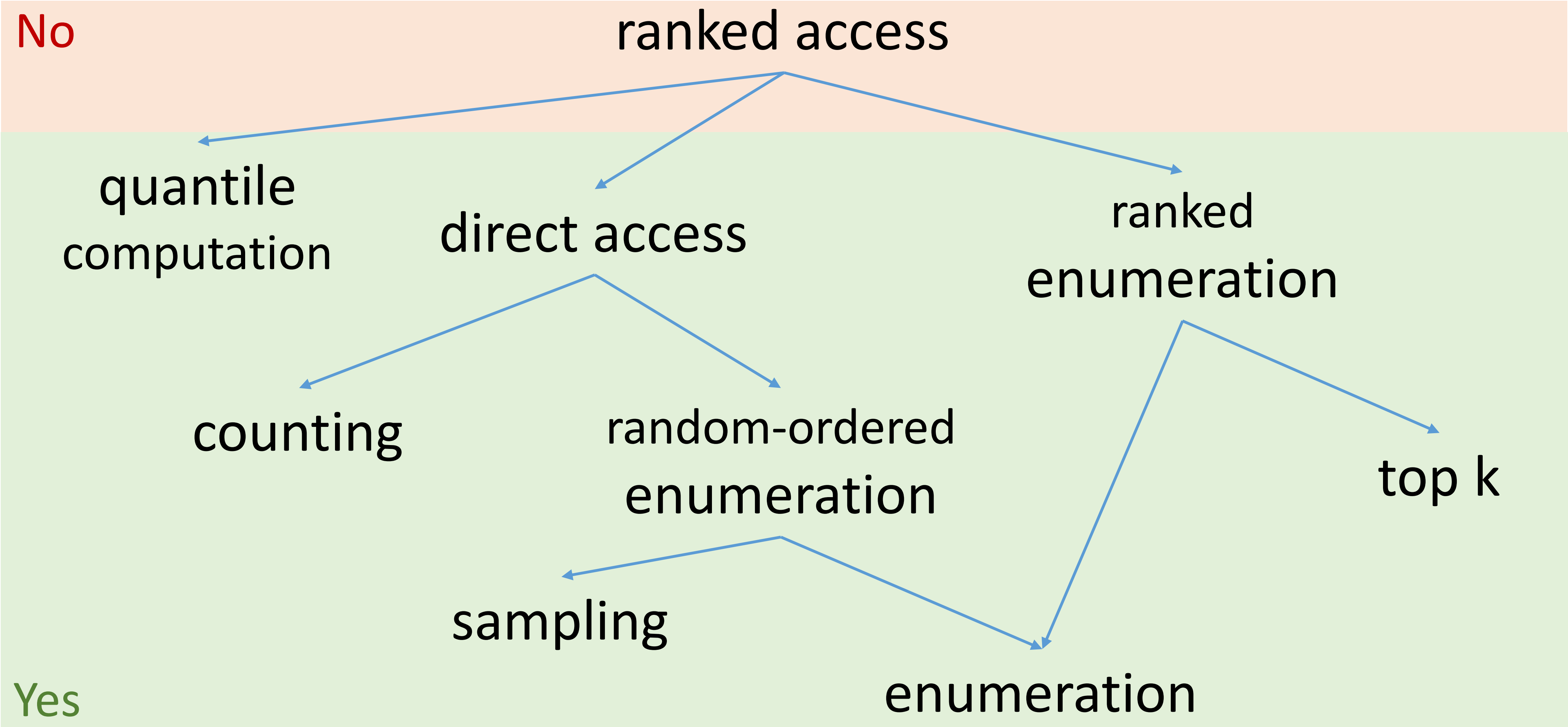


* with log time per answer after linear preprocessing

Can be solved efficiently* for all free-connex CQs?

[**C**, Tziavelis, Gatterbauer, Kimelfeld, Riedewald; PODS 21]
[Tziavelis, Gatterbauer, Riedewald; VLDB 21]

For lexicographic orders:

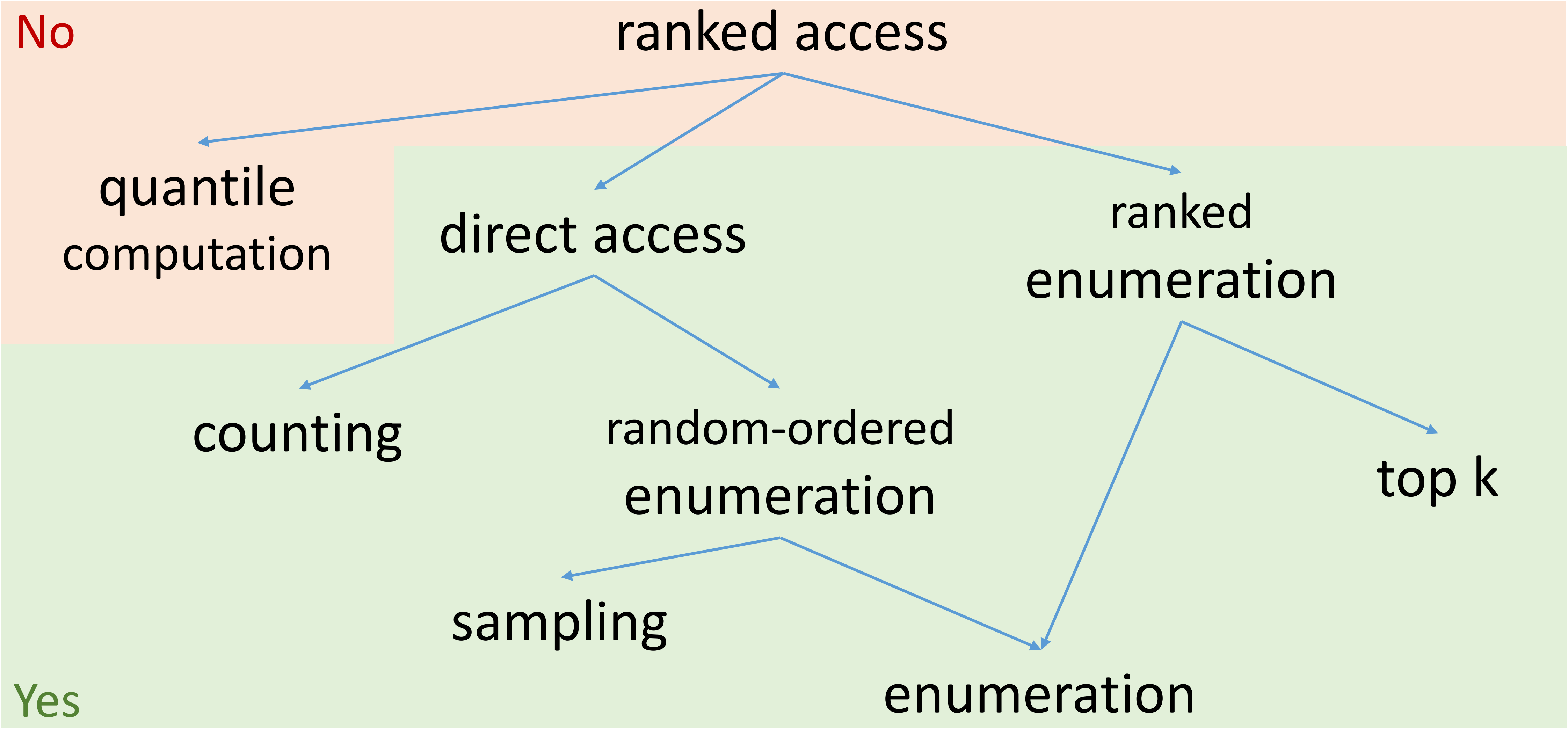


* with log time per answer after linear preprocessing

Can be solved efficiently* for all free-connex CQs?

For sum of weights orders:

[C, Tziavelis, Gatterbauer, Kimelfeld, Riedewald; PODS 21]
[Tziavelis, Gatterbauer, Riedewald; VLDB 21]



* with log time per answer after linear preprocessing

Dichotomy

[**C**, Tziavelis, Gatterbauer, Kimelfeld, Riedewald; PODS 21]

- Given a conjunctive query Q,

Tractability is trivial

$Q_1(x, z) \leftarrow R(x, y, z), S(y, z)$ ✓

If Q is acyclic with an atom containing all free variables,
Q ∈ ΣWeightAccess<lin,log>

Otherwise, Q ∉ ΣWeightAccess<lin,log>*

$Q_2(x, z) \leftarrow R(x, y), S(y, z)$ ✗

* no self-joins, assuming 3SUM and sHyperclique

Hardness

3SUM hypothesis

given 3 sets of integers $|A| = |B| = |C| = n$,
deciding $\exists\ a \in A, b \in B, c \in C$ s.t. $a + b + c = 0$
cannot be done in time $O(n^{2-\varepsilon})$ for any $\varepsilon > 0$

$\notin \Sigma\text{WeightAccess}\langle n^{2-\varepsilon}, n^{1-\varepsilon}\rangle$

$$Q_2(x, z) \leftarrow R(x, y), S(y, z)$$

x	y
a_1	0
a_2	0

A

y	z
0	b_1
0	b_2

B

x	y	z	w
a_1	0	b_1	$a_1 + b_1$
a_1	0	b_2	$a_1 + b_2$
a_2	0	b_1	$a_2 + b_1$
a_2	0	b_2	$a_2 + b_2$

Binary search for $-c$ ($\forall c$)

(log number of access calls)

Plan

- Enumeration
  - Join queries
    - Self-joins
  - Conjunctive queries
  - Unions of conjunctive queries
- Other Evaluation Tasks
  - The tasks
  - Known complexity results