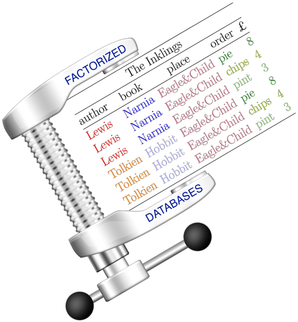
Joins → Aggregates → Optimization

https://fdbresearch.github.io



Dan Olteanu

PhD Open School
University of Warsaw
November 23, 2018

Acknowledgements

Some work reported in this course has been done in the context of the FDB project, LogicBlox, and RelationalAI by

- Zavodný, Schleich, Kara, Nikolic, Zhang, Ciucanu, and Olteanu (Oxford)
- Abo Khamis and Ngo (RelationalAI), Nguyen (U. Michigan)

Some of the following slides are derived from presentations by

- Abo Khamis (optimization diagrams)
- Kara (covers, IVM^{ϵ}, and many graphics)
- Ngo (functional aggregate queries)
- Schleich (performance and quizzes)

Lastly, Kara and Schleich proofread the slides.

I would like to thank them for their support!

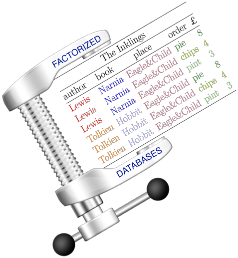
Goal of This Course

$\boxed{\text{Introduction to a principled approach to in-database computation}}$

This course starts where mainstream databases courses finish.

- Part 1: Joins
- **Part 2: Aggregates**
  - Important functionality of DB query languages and essential for applications
  - Natural generalization of aggregates over joins can express problems across Computer Science in, e.g., DB, logic, probabilistic graphical models [ANR16]
  - Algorithm with lowest known computational complexity [BKOZ13,ANR16]
  - Aggregates under data updates [NO18,KNNOZ19]
- Part 3: Optimization

Outline of Part 2: Aggregates



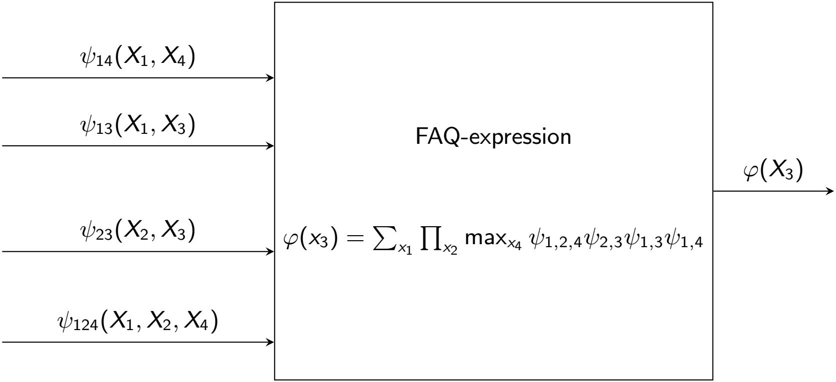
FAQ: **F**unctional **A**ggregate **Q**ueries [ANR16]

We use the following notation ($i \in [n] = \{1, \ldots, n\}$):

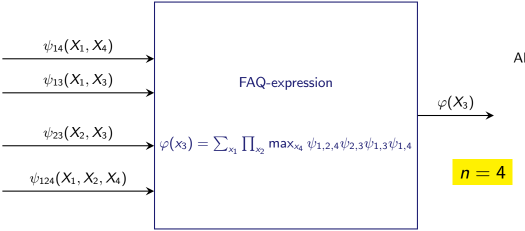
- X_i are variables,
- x_i are values in discrete domain $\mathrm{Dom}(X_i)$
- $\mathbf{x} = (x_1, \ldots, x_n) \in \mathrm{Dom}(X_1) \times \cdots \times \mathrm{Dom}(X_n)$
- For any $S \subseteq [n]$,

$$\begin{aligned} \mathbf{x}_S &= (x_i)_{i \in S} \in \prod_{i \in S} \mathrm{Dom}(X_i) \\ \text{e.g. } \mathbf{x}_{\{2,5,8\}} &= (x_2, x_5, x_8) \in \mathrm{Dom}(X_2) \times \mathrm{Dom}(X_5) \times \mathrm{Dom}(X_8) \end{aligned}$$

Functional **A**ggregate **Q**uery: The Problem



Functional Aggregate Query: The Input

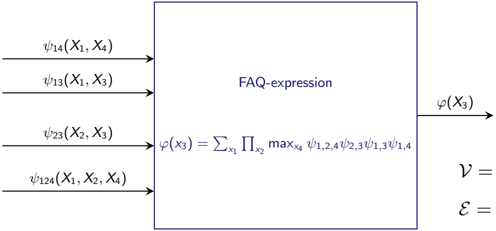


All functions have the same range **D**

- *n variables* $X_1, \ldots, X_n$
- a multi-*hypergraph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$
  - Each vertex is a variable (notation overload: $\mathcal{V} = [n]$)
  - To each hyperedge $S \in \mathcal{E}$ there corresponds a *factor* ψ_S

$$\psi_S : \prod_{i \in S} \mathrm{Dom}(X_i) \to \mathbf{D}$$

Functional Aggregate Query: The Input



All functions have the same range **D**

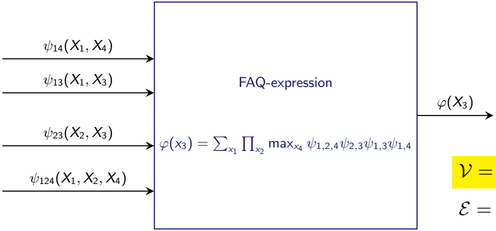
$\mathcal{V} = \{1, 2, 3, 4\}$

$\mathcal{E} = \{\{1, 4\}, \{1, 3\}, \{2, 3\}, \{1, 2, 4\}\}$

- *n variables* $X_1, \ldots, X_n$
- a multi-*hypergraph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$
  - Each vertex is a variable (notation overload: $\mathcal{V} = [n]$)
  - To each hyperedge $S \in \mathcal{E}$ there corresponds a *factor* ψ_S

$$\psi_S : \prod_{i \in S} \mathrm{Dom}(X_i) \to \mathbf{D}$$

Functional Aggregate Query: The Input



All functions have the same range **D**

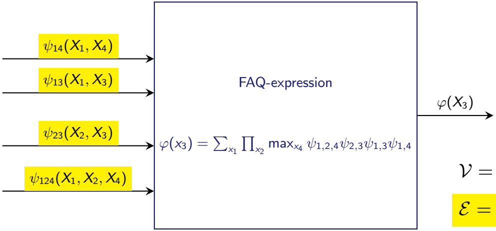
$\mathcal{V} = \{1, 2, 3, 4\}$

$\mathcal{E} = \{\{1, 4\}, \{1, 3\}, \{2, 3\}, \{1, 2, 4\}\}$

- *n variables* $X_1, \dots, X_n$
- a multi-*hypergraph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$
  - Each vertex is a variable (notation overload: $\mathcal{V} = [n]$)
  - To each hyperedge $S \in \mathcal{E}$ there corresponds a *factor* ψ_S

$$\psi_S : \prod_{i \in S} \mathrm{Dom}(X_i) \to \mathbf{D}$$

Functional Aggregate Query: The Input



All functions have the same range **D**

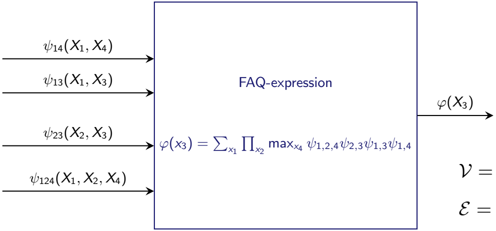
$\mathcal{V} = \{1, 2, 3, 4\}$

$\mathcal{E} = \{\{1, 4\}, \{1, 3\}, \{2, 3\}, \{1, 2, 4\}\}$

- *n variables* $X_1, \ldots, X_n$
- a multi-*hypergraph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$
  - Each vertex is a variable (notation overload: $\mathcal{V} = [n]$)
  - To each hyperedge $S \in \mathcal{E}$ there corresponds a *factor* ψ_S

$$\psi_S : \prod_{i \in S} \mathrm{Dom}(X_i) \to \mathbf{D}$$

Functional **A**ggregate **Q**uery: The Input



All functions have the same range **D**

$$\mathcal{V} = \{1, 2, 3, 4\}$$

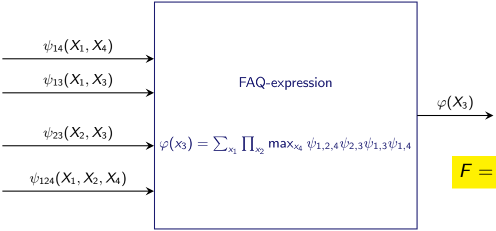
$$\mathcal{E} = \{\{1, 4\}, \{1, 3\}, \{2, 3\}, \{1, 2, 4\}\}$$

- *n variables* $X_1, \dots, X_n$
- a multi-*hypergraph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$
  - Each vertex is a variable (notation overload: $\mathcal{V} = [n]$)
  - To each hyperedge $S \in \mathcal{E}$ there corresponds a *factor* ψ_S

$$\psi_S : \prod_{i \in S} \mathrm{Dom}(X_i) \to \mathbf{D}$$

$\mathbf{R}_+$, $\{\mathrm{true}, \mathrm{false}\}$, $\{0, 1\}$, $2^{\mathcal{U}}$, etc.

Functional Aggregate Query: The Input



All functions have the same range **D**

$F = \{3\}$

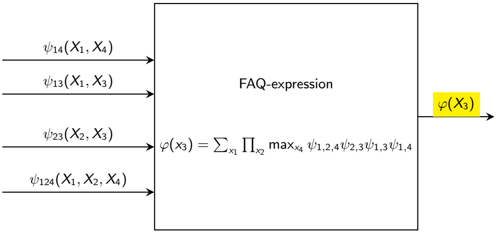
- n *variables* $X_1, \dots, X_n$
- a multi-*hypergraph* $\mathcal{H} = (\mathcal{V}, \mathcal{E})$
  - Each vertex is a variable (notation overload: $\mathcal{V} = [n]$)
  - To each hyperedge $S \in \mathcal{E}$ there corresponds a *factor* ψ_S

$$\psi_S : \prod_{i \in S} \mathrm{Dom}(X_i) \to \mathbf{D}$$

$\mathbf{R}_+$, $\{\text{true}, \text{false}\}$, $\{0, 1\}$, $2^{\mathcal{U}}$, etc.

- a set $F \subseteq \mathcal{V}$ of *free variables* (wlog, $F = [f] = \{1, \dots, f\}$)

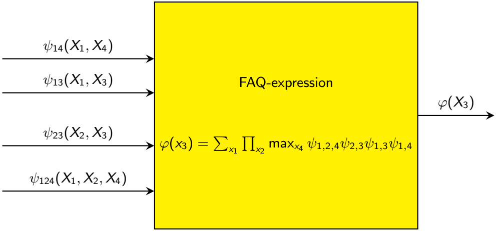
Functional **A**ggregate **Q**uery: The Output



All functions have the same range **D**

- Compute the function $\varphi : \prod_{i \in F} \mathrm{Dom}(X_i) \to \mathbf{D}$.

Functional Aggregate Query: The Output

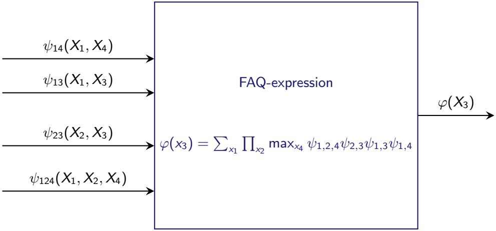


All functions have the same range **D**

- Compute the function $\varphi : \prod_{i \in F} \mathrm{Dom}(X_i) \to \mathbf{D}$.
- φ defined by the *FAQ-expression*

$$\varphi(\mathbf{x}_{[f]}) = \bigoplus^{(f+1)}_{x_{f+1}\in \mathrm{Dom}(X_{f+1})} \cdots \bigoplus^{(n-1)}_{x_{n-1}\in\mathrm{Dom}(X_{n-1})} \bigoplus^{(n)}_{x_n\in\mathrm{Dom}(X_n)} \bigotimes_{S\in\mathcal{E}} \psi_S(\mathbf{x}_S)$$

Functional Aggregate Query: The Output



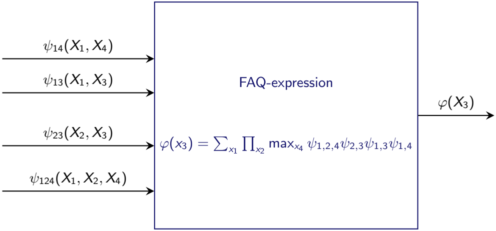
All functions have the same range **D**

- Compute the function $\varphi : \prod_{i \in F} \text{Dom}(X_i) \to \mathbf{D}$.
- φ defined by the *FAQ-expression*

$$\varphi(\mathbf{x}_{[f]}) = \bigoplus^{(f+1)}_{x_{f+1} \in \text{Dom}(X_{f+1})} \cdots \bigoplus^{(n-1)}_{x_{n-1} \in \text{Dom}(X_{n-1})} \bigoplus^{(n)}_{x_n \in \text{Dom}(X_n)} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S)$$

- For each $\bigoplus^{(i)}$

Functional Aggregate Query: The Output



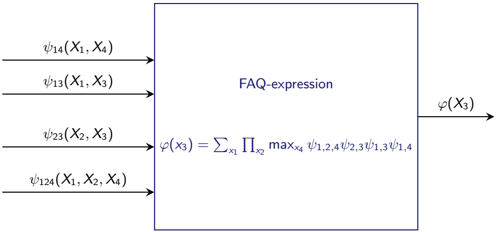
All functions have the same range **D**

- Compute the function $\varphi : \prod_{i \in F} \text{Dom}(X_i) \to \mathbf{D}$.
- φ defined by the *FAQ-expression*

$$\varphi(\mathbf{x}_{[f]}) = \bigoplus_{x_{f+1} \in \text{Dom}(X_{f+1})}^{(f+1)} \cdots \bigoplus_{x_{n-1} \in \text{Dom}(X_{n-1})}^{(n-1)} \bigoplus_{x_n \in \text{Dom}(X_n)}^{(n)} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S)$$

- For each $\bigoplus^{(i)}$
  - Either $\left(\mathbf{D}, \bigoplus^{(i)}, \bigotimes\right)$ is a *commutative semiring*

Functional Aggregate Query: The Output



All functions have the same range **D**

- Compute the function $\varphi : \prod_{i \in F} \text{Dom}(X_i) \to \mathbf{D}$.
- φ defined by the *FAQ-expression*

$$\varphi(\mathbf{x}_{[f]}) = \bigoplus_{x_{f+1} \in \text{Dom}(X_{f+1})}^{(f+1)} \cdots \bigoplus_{x_{n-1} \in \text{Dom}(X_{n-1})}^{(n-1)} \bigoplus_{x_n \in \text{Dom}(X_n)}^{(n)} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S)$$

- For each $\bigoplus^{(i)}$
  - Either $\left(\mathbf{D}, \bigoplus^{(i)}, \bigotimes\right)$ is a *commutative semiring*
  - Or $\bigoplus^{(i)} = \bigotimes$

Commutative Semirings

$(\mathbf{D}, \oplus, \otimes)$ is a commutative semiring when

- $(\mathbf{D}, \oplus)$ is a commutative monoid with identity element $\mathbf{0}$:
  - $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - $\mathbf{0} \oplus a = a \oplus \mathbf{0} = a$
  - $a \oplus b = b \oplus a$
- $(\mathbf{D}, \otimes)$ is a commutative monoid with identity element $\mathbf{1}$:
  - $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - $\mathbf{1} \otimes a = a \otimes \mathbf{1} = a$
  - $a \oplus b = b \oplus a$
- Multiplication distributes over addition:
  - $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
- Multiplication by $\mathbf{0}$ annihilates $\mathbf{D}$:
  - $\mathbf{0} \otimes a = a \otimes \mathbf{0} = \mathbf{0}$

Commutative Semirings

$(\mathbf{D}, \oplus, \otimes)$ is a commutative semiring when

- $(\mathbf{D}, \oplus)$ is a commutative monoid with identity element $\mathbf{0}$:
  - $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - $\mathbf{0} \oplus a = a \oplus \mathbf{0} = a$
  - $a \oplus b = b \oplus a$
- $(\mathbf{D}, \otimes)$ is a commutative monoid with identity element $\mathbf{1}$:
  - $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - $\mathbf{1} \otimes a = a \otimes \mathbf{1} = a$
  - $a \oplus b = b \oplus a$
- Multiplication distributes over addition:
  - $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
- Multiplication by $\mathbf{0}$ annihilates $\mathbf{D}$:
  - $\mathbf{0} \otimes a = a \otimes \mathbf{0} = \mathbf{0}$

Common examples (there are many more!)

Boolean	$(\{\text{true}, \text{false}\}, \vee, \wedge)$
sum-product	$(\mathbb{R}, +, \times)$
max-product	$(\mathbb{R}_+, \max, \times)$
set	$(2^U, \cup, \cap)$

SumProduct $\subset$ FAQ

Problem (SumProduct)

Given a commutative semiring $(\mathbf{D}, \oplus, \otimes)$, *compute the function*

$$\varphi(x_1, \ldots, x_f) = \bigoplus_{x_{f+1}} \bigoplus_{x_{f+2}} \cdots \bigoplus_{x_n} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S)$$

For $\oplus = +$ and $\otimes = *$, φ can be expressed in SQL as:

$$\begin{aligned}
&\text{SELECT } x_1, \ldots, x_f, \text{SUM}(R_1.\text{val} * \cdots * R_n.\text{val})\\
&\text{FROM } R_1 \text{ NATURAL JOIN } \ldots R_n\\
&\text{GROUP BY } x_1, \ldots, x_f;
\end{aligned}$$

where each function ψ_i over variables $\mathbf{X}_S$ is encoded as a relation R_i over $\mathbf{X}_S$ and an additional variable val to account for the values of ψ_i.

This formulation is equivalent to:

- SumProduct [D99]
- Marginalize a Product Function [AM00]

Many examples for SumProduct

- $(\{\text{true}, \text{false}\}, \vee, \wedge)$
  - Constraint satisfaction problems
  - **Boolean conjunctive query evaluation**
  - SAT
  - k-colorability
  - etc.
- $(U, \cup, \cap)$
  - Conjunctive query evaluation
- $(\mathbb{R}, +, \times)$
  - Permanent
  - DFT
  - **Inference in probabilistic graphical models**
  - $\#$CSP
  - **Matrix chain multiplication**
  - Aggregates in DB
- $(\mathbf{R}_+, \max, \times)$
  - **MAP queries in probabilistic graphical models**

SumProduct Example 1: Boolean Query Evaluation

Boolean Conjunctive Queries:

- Boolean query Φ with set $rels(\Phi)$ of relation symbols
- Each relation symbol $R \in rels(\Phi)$ has variables $vars(R)$

$$\Phi = \exists X_1 \ldots \exists X_n : \bigwedge_{R \in rels(\Phi)} R(vars(R))$$

FAQ encoding:

$$\phi = \bigvee_{\mathbf{x}} \bigwedge_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S), \text{ where}$$

- ϕ has the hypergraph $(\mathcal{V}, \mathcal{E})$ with
- $\mathcal{V} = \bigcup_{R \in rels(\Phi)} vars(R)$ and $\mathcal{E} = \{vars(R) \mid R \in rels(\Phi)\}$
- For each $S \in \mathcal{E}$, there is a factor ψ_S such that $\psi_S(\mathbf{x}_S) = (\mathbf{x}_S \in R)$

SumProduct Example 2: Matrix Chain Multiplication

$\boxed{\text{Compute the product } \mathbf{A} = \mathbf{A}_1 \cdots \mathbf{A}_n \text{ of } n \text{ matrices}}$

- Each matrix $\mathbf{A}_i$ is over field $\mathbb{F}$ and has dimensions $p_i \times p_{i+1}$

FAQ encoding:

- We use $n+1$ variables $X_1, \ldots, X_{n+1}$ with domains $\mathrm{Dom}(X_i) = [p_i]$
- Each matrix $\mathbf{A}_i$ can be viewed as a function of two variables:

$$\psi_{i,i+1} : \mathrm{Dom}(X_i) \times \mathrm{Dom}(X_{i+1}) \to \mathbb{F}, \text{ where } \psi_{i,i+1}(x_i, x_{i+1}) = (\mathbf{A}_i)_{x_i x_{i+1}}$$

The problem is now to compute the FAQ expression

$$\phi(x_1, x_{n+1}) = \sum_{x_2 \in \mathrm{Dom}(X_2)} \cdots \sum_{x_n \in \mathrm{Dom}(X_n)} \prod_{i \in [n]} \psi_{i,i+1}(x_i, x_{i+1}).$$

SumProduct Example 3: Queries in Graphical Models

- Discrete undirected graphical model represented by a hypergraph $(\mathcal{V}, \mathcal{E})$
- $\mathcal{V} = \{X_1, \ldots, X_n\}$ consists of n discrete random variables
- There is a factor $\psi_S : \prod_{i \in S} \mathrm{Dom}(X_i) \to \mathbb{R}_+$ for each edge $S \in \mathcal{E}$

FAQ expression to compute the marginal Maximum A Posteriori estimates:

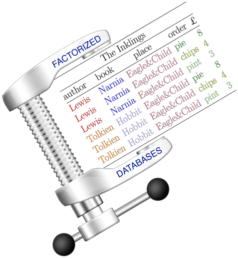
$$\phi(x_1, \ldots, x_f) = \max_{x_{f+1} \in \mathrm{Dom}(X_{f+1})} \cdots \max_{x_n \in \mathrm{Dom}(X_n)} \prod_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S)$$

FAQ expression to compute the marginal distribution of variables $X_1, \ldots, X_f$:

$$\phi(x_1, \ldots, x_f) = \sum_{x_{f+1} \in \mathrm{Dom}(X_{f+1})} \cdots \sum_{x_n \in \mathrm{Dom}(X_n)} \prod_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S)$$

For conditional distributions $\mathit{prob}(\mathbf{X}_A \mid \mathbf{X}_B = \mathbf{x}_B)$, we set $\mathbf{X}_B$ to $\mathbf{x}_B$.

Outline of Part 2: Aggregates



How to compute a SumProduct FAQ φ

- Find a variable order for φ

- Compute φ by eliminating variables in the given order

  This is a dynamic programming algorithm. Two flavours:

  - FDB: Top-down with memoization (caching) [BKOZ13]

    We exemplify two variants:

    1. Compute the factorized join and the aggregates in one pass over the factorization
    2. Translate the factorized computation into relational queries

  - InsideOut: Bottom-up with indicator projections [ANR16]

- The complexity is given by the width of the variable order:

Given a database of size N, an FAQ φ, a variable order for φ with width w, φ can be computed in time $\mathcal{O}(N^w + |\text{OUT}|)$, where $|\text{OUT}|$ is the output size.

Finding a Variable Order for a SumProduct FAQ φ

First attempt: Same variable order Δ as for the join part of φ

One wrinkle: What if not all variables are free?

Finding a Variable Order for a SumProduct FAQ φ

First attempt: Same variable order Δ as for the join part of φ

One wrinkle: What if not all variables are free?

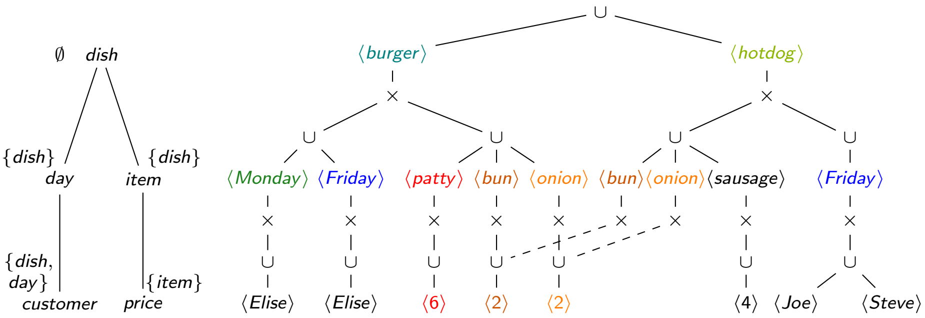
- The free variables sit above the bound variables in Δ. [BKOZ13,OZ15]
- Equivalent constraint for hypertree decompositions: [ANR16]

  Take a hypertree decomposition for the join part of φ such that the bags with the free variables form a connected subtree.

Implication on complexity:

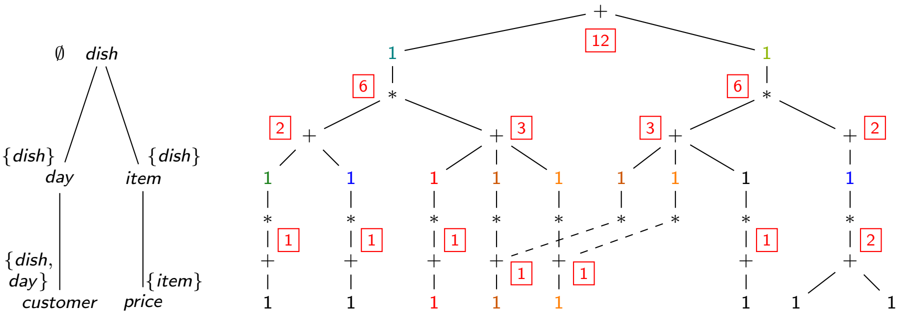
- The width for φ is at least the width for its join part
  $\Rightarrow$ φ may be more expensive than its join part if it has free variables
- This new width is called the *FAQ-width* in the literature [ANR16]

Computing COUNT over Factorized Join using FDB



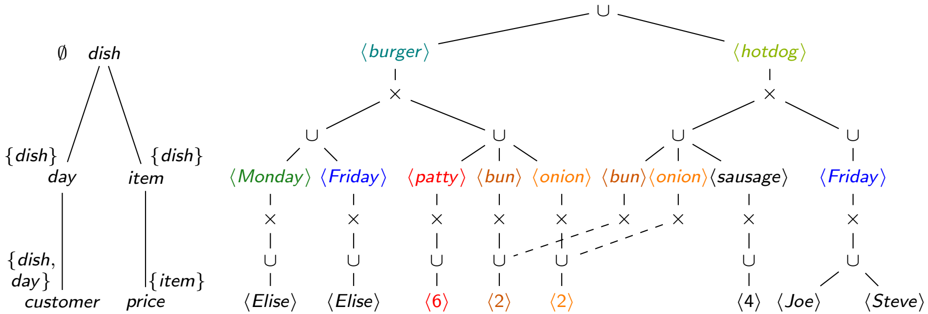
- $\varphi = \sum_{\dots} O(customer, day, dish) \cdot D(dish, item) \cdot I(item, price)$
- In SQL: `SELECT COUNT(*) FROM O NATURAL JOIN .. I;`
- We change the semiring to $(\mathbb{N}, +, *)$:
  - values $\mapsto 1$ $\quad\quad \cup \mapsto +$ $\quad\quad \times \mapsto *$

Computing COUNT over Factorized Join using FDB



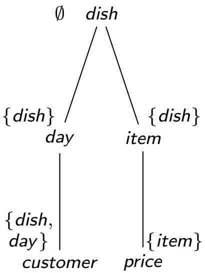
- $\varphi = \sum_{\dots} O(customer, day, dish) \cdot D(dish, item) \cdot I(item, price)$
- In SQL: `SELECT COUNT(*) FROM O NATURAL JOIN .. I;`
- We change the semiring to $(\mathbb{N}, +, *)$:
  - values $\mapsto 1$ $\quad\cup \mapsto +\quad$ $\times \mapsto *$

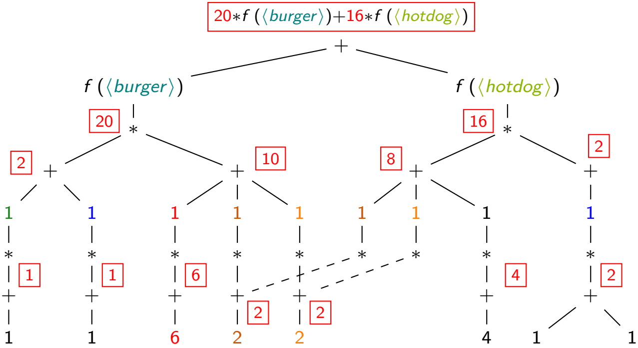
Computing SUM over Factorized Join using FDB



- $\varphi = \sum_{\ldots} f(\text{dish}) \cdot \text{price} \cdot O(\mathit{customer}, \mathit{day}, \mathit{dish}) \cdot D(\mathit{dish}, \mathit{item}) \cdot I(\mathit{item}, \mathit{price})$
- In SQL: `SELECT SUM(`f`(dish) * price) FROM O NATURAL JOIN .. I;`
  - Assume there is a function f that turns `dish` into reals or indicator vectors.
  - All values except for `dish` & `price` $\mapsto 1$, $\cup \mapsto +$, $\times \mapsto *$.

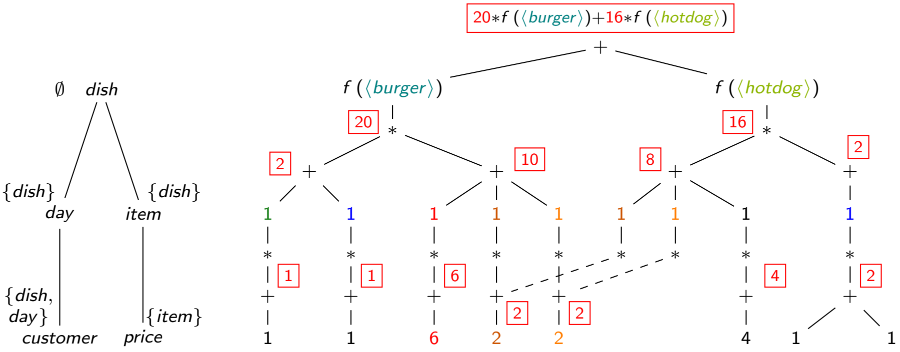
Computing SUM over Factorized Join using FDB





- $\varphi = \sum_{\dots} f(\text{dish}) \cdot \text{price} \cdot O(\textit{customer}, \textit{day}, \textit{dish}) \cdot D(\textit{dish}, \textit{item}) \cdot I(\textit{item}, \textit{price})$
- In SQL: `SELECT SUM(`f`(dish) * price) FROM O NATURAL JOIN .. I;`
  - Assume there is a function f that turns `dish` into reals or indicator vectors.
  - All values except for `dish` & `price` $\mapsto 1$, $\cup \mapsto +$, $\times \mapsto *$.

Computing SUM over Factorized Join using FDB



If f turns `dish` into indicator vectors:

- $\varphi(dish) = \sum_{\dots} \text{price} \cdot O(customer, day, dish) \cdot D(dish, item) \cdot I(item, price)$
- In SQL:
  `SELECT dish, SUM(price) FROM O NATURAL JOIN..I GROUP BY dish;`

To Compute or Not To Compute the Factorized Join

Aggregates can be computed *without* materializing the factorized join

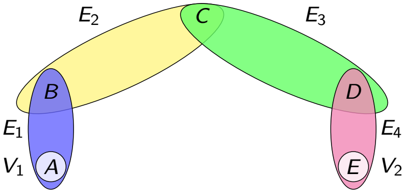
[OZ15,OS16,ANNOS18a+b]

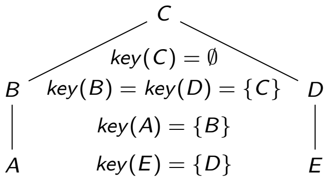
- The factorized join becomes the *trace* of the aggregate computation
- This is called *factorized aggregate computation*

Example of Factorized Computation via Query Rewriting

The 4-path count query Q_4 on a graph with 4 copies of the edge relation E:

$$Q_4() = \sum_{a,b,c,d,e} \underbrace{V_1(a) \cdot E_1(a,b) \cdot E_2(b,c) \cdot E_3(c,d) \cdot E_4(d,e) \cdot V_2(e)}_{J(a,b,c,d,e)}$$

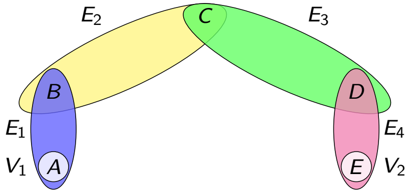


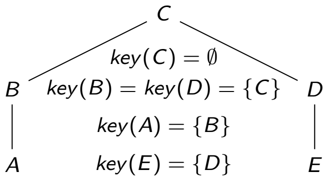


Example of Factorized Computation via Query Rewriting

The 4-path count query Q_4 on a graph with 4 copies of the edge relation E:

$$Q_4() = \sum_{a,b,c,d,e} \underbrace{V_1(a) \cdot E_1(a,b) \cdot E_2(b,c) \cdot E_3(c,d) \cdot E_4(d,e) \cdot V_2(e)}_{J(a,b,c,d,e)}$$





Sizes for listing/factorized representations of the result of the join J of Q_4

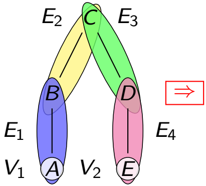
- $\rho^*(J) = 3 \Rightarrow$ listing representation has size $O(|E|^3)$.
- $\mathit{fhtw}(J) = 1 \Rightarrow$ factorization with caching has size $O(|E|)$.

Example of Factorized Computation via Query Rewriting

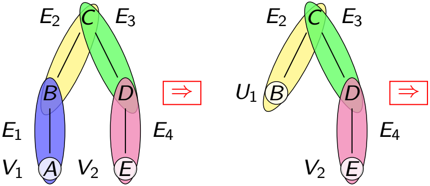
We would like to compute Q_4:

- in $O(|E|)$ time (no free variables, so use best variable order)
- using optimized queries that are derived from the variable order of Q_4
- without materializing the factorized join J

Example of Factorized Computation via Query Rewriting

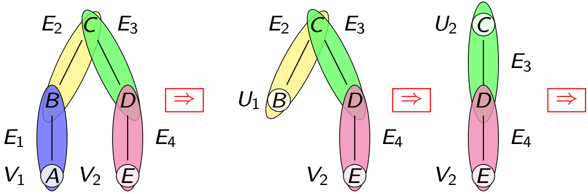


Example of Factorized Computation via Query Rewriting



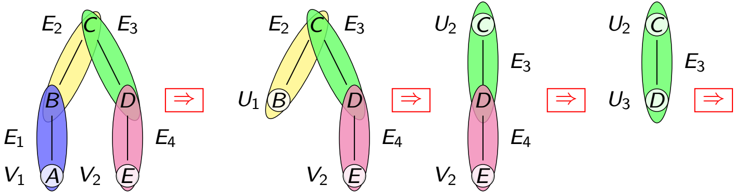
$$U_1(b) = \sum_a V_1(a) \cdot E_1(b, a)$$

Example of Factorized Computation via Query Rewriting



$$U_1(b) = \sum_a V_1(a) \cdot E_1(b, a) \qquad U_2(c) = \sum_b E_2(c, b) \cdot U_1(b)$$

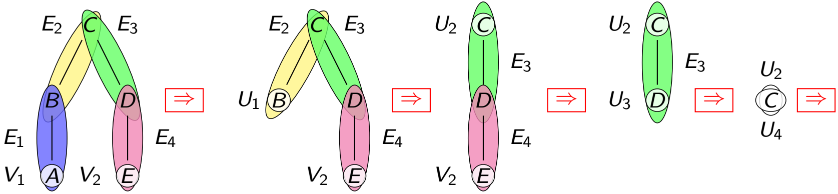
Example of Factorized Computation via Query Rewriting



$$U_1(b) = \sum_a V_1(a) \cdot E_1(b, a) \qquad U_2(c) = \sum_b E_2(c, b) \cdot U_1(b)$$

$$U_3(d) = \sum_e V_2(e) \cdot E_4(d, e)$$

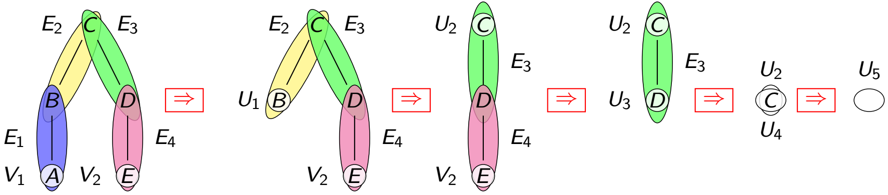
Example of Factorized Computation via Query Rewriting



$$U_1(b) = \sum_a V_1(a) \cdot E_1(b, a) \qquad U_2(c) = \sum_b E_2(c, b) \cdot U_1(b)$$

$$U_3(d) = \sum_e V_2(e) \cdot E_4(d, e) \qquad U_4(c) = \sum_d E_3(c, d) \cdot U_3(d)$$

Example of Factorized Computation via Query Rewriting

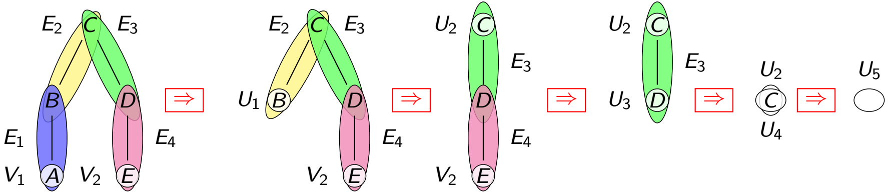


$$U_1(b) = \sum_a V_1(a) \cdot E_1(b, a) \qquad U_2(c) = \sum_b E_2(c, b) \cdot U_1(b)$$

$$U_3(d) = \sum_e V_2(e) \cdot E_4(d, e) \qquad U_4(c) = \sum_d E_3(c, d) \cdot U_3(d)$$

$$U_5 = \sum_c U_2(c) \cdot U_4(c)$$

Example of Factorized Computation via Query Rewriting



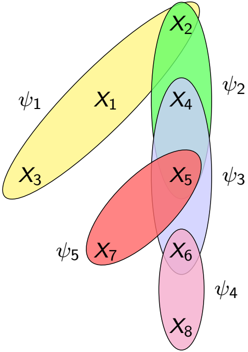
This computation strategy corresponds to the following query rewriting:

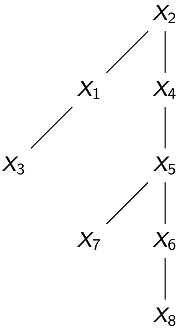
$$\sum_{a,b,c,d,e} V_1(a) \cdot E_1(b,a) \cdot E_2(c,b) \cdot E_3(c,d) \cdot E_4(d,e) \cdot V_2(e) =$$

$$\sum_c \Big(\sum_b E_2(c,b) \cdot \big(\sum_a V_1(a) \cdot E_1(b,a) \big) \Big) \cdot \Big(\sum_d E_3(c,d) \cdot \big(\sum_e E_4(d,e) \cdot V_2(e) \big) \Big)$$

Example of FAQ Computation using InsideOut

$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

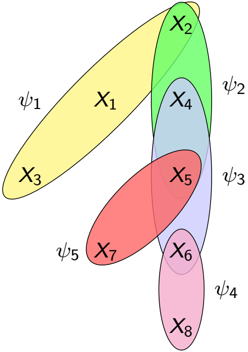


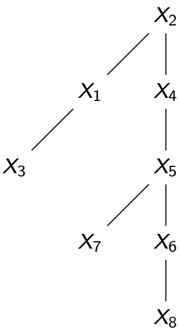


$$key(X_2) = \emptyset$$
$$key(X_1) = \{X_2\}$$
$$key(X_3) = \{X_1, X_2\}$$
$$key(X_4) = \{X_2\}$$
$$key(X_5) = \{X_2, X_4\}$$
$$key(X_6) = \{X_4, X_5\}$$
$$key(X_8) = \{X_6\}$$
$$key(X_7) = \{X_5\}$$

Example of FAQ Computation using InsideOut

$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$





$key(X_2) = \emptyset$

$key(X_1) = \{X_2\}$

$key(X_3) = \{X_1, X_2\}$

$key(X_4) = \{X_2\}$

$key(X_5) = \{X_2, X_4\}$

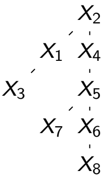
$key(X_6) = \{X_4, X_5\}$

$key(X_8) = \{X_6\}$

$key(X_7) = \{X_5\}$

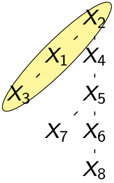
- $\rho^*(\varphi) = 4$, $s(\varphi) = 2$, $fhtw(\varphi) = 1$. The above variable order Δ has the free variables x_1, x_2, x_4 on top of the others and $fhtw(\Delta) = 1$.
- The query result has size: $O(N)$ when factorized; $O(N^2)$ when listed

Example of FAQ Computation using InsideOut



$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

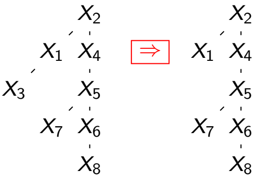
Example of FAQ Computation using InsideOut



$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \underbrace{\left(\sum_{x_3} \psi_1(x_1, x_2, x_3) \right)}_{\psi_6(x_1, x_2)} \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

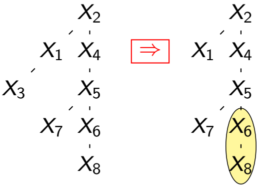
Example of FAQ Computation using InsideOut



$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

Example of FAQ Computation using InsideOut

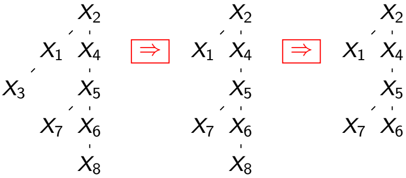


$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \underbrace{\left(\sum_{x_8} \psi_4(x_6, x_8) \right)}_{\psi_7(x_6)} \cdot \psi_5(x_5, x_7)$$

Example of FAQ Computation using InsideOut

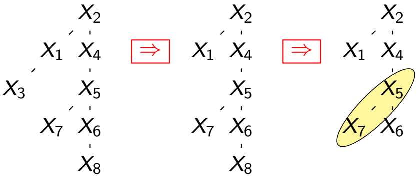


$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

Example of FAQ Computation using InsideOut



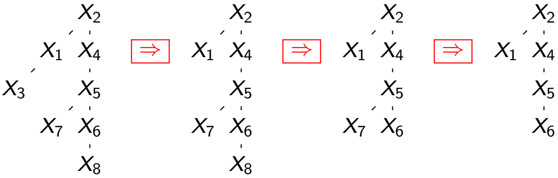
$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \underbrace{\left(\sum_{x_7} \psi_5(x_5, x_7) \right)}_{\psi_8(x_5)}$$

Example of FAQ Computation using InsideOut



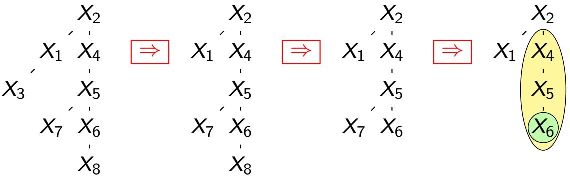
$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

Example of FAQ Computation using InsideOut



$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

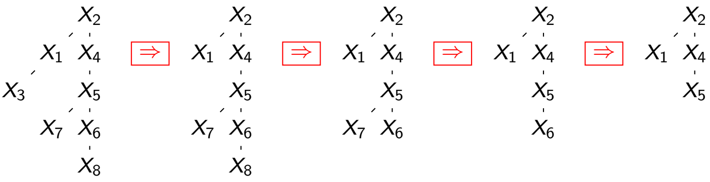
$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \underbrace{\left(\sum_{x_6} \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \right)}_{\psi_9(x_4, x_5)} \cdot \psi_8(x_5)$$

Example of FAQ Computation using InsideOut



$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

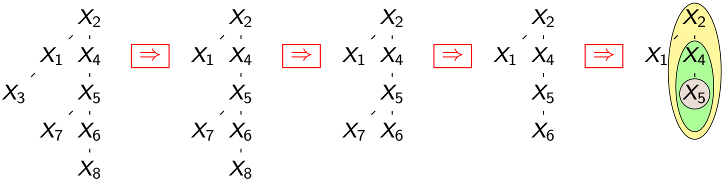
$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_9(x_4, x_5) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

Example of FAQ Computation using InsideOut



$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

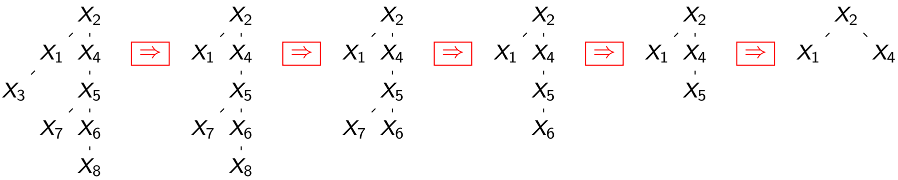
$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_9(x_4, x_5) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \psi_6(x_1, x_2) \cdot \underbrace{\left(\sum_{x_5} \psi_2(x_2, x_4, x_5) \cdot \psi_9(x_4, x_5) \cdot \psi_8(x_5) \right)}_{\psi_{10}(x_2, x_4)}$$

Example of FAQ Computation using InsideOut



$$\varphi(x_1, x_2, x_4) = \sum_{x_3, x_5, x_6, x_7, x_8} \psi_1(x_1, x_2, x_3) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7, x_8} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_4(x_6, x_8) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6, x_7} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_5(x_5, x_7) \qquad \widetilde{O}(N)$$

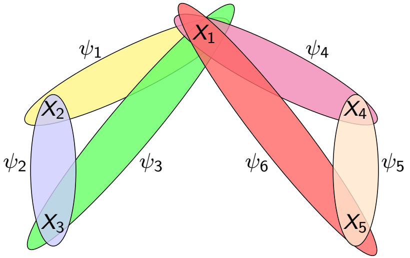
$$\varphi(x_1, x_2, x_4) = \sum_{x_5, x_6} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_3(x_4, x_5, x_6) \cdot \psi_7(x_6) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

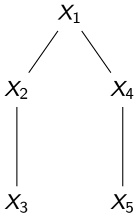
$$\varphi(x_1, x_2, x_4) = \sum_{x_5} \psi_6(x_1, x_2) \cdot \psi_2(x_2, x_4, x_5) \cdot \psi_9(x_4, x_5) \cdot \psi_8(x_5) \qquad \widetilde{O}(N)$$

$$\varphi(x_1, x_2, x_4) = \psi_6(x_1, x_2) \cdot \psi_{10}(x_2, x_4) \qquad \widetilde{O}(N)$$

Example of FAQ Computation with Indicator Projections

$$\varphi(x_1) = \sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_2(x_2,x_3)\cdot\psi_3(x_3,x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$

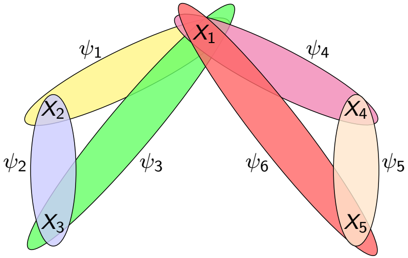


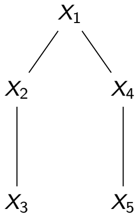


$$key(X_1) = \emptyset$$
$$key(X_2) = \{X_1\}$$
$$key(X_3) = \{X_1, X_2\}$$
$$key(X_4) = \{X_1\}$$
$$key(X_5) = \{X_1, X_4\}$$

Example of FAQ Computation with Indicator Projections

$$\varphi(x_1) = \sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_2(x_2,x_3)\cdot\psi_3(x_3,x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$





$key(X_1) = \emptyset$

$key(X_2) = \{X_1\}$

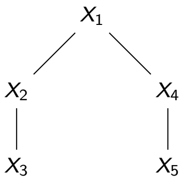
$key(X_3) = \{X_1, X_2\}$

$key(X_4) = \{X_1\}$

$key(X_5) = \{X_1, X_4\}$

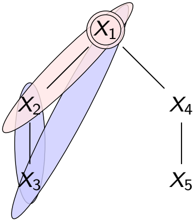
- $\rho^*(\varphi) = 2.5$, $s(\varphi) = 1.5$, $fhtw(\varphi) = 1.5$. The above variable order Δ has the free variable x_1 on top of the others and $fhtw(\Delta) = 1.5$.
- The (unary) query result has size $O(N)$ when factorized or listed.

Example of FAQ Computation with Indicator Projections



$$\varphi(x_1) = \sum_{x_2, x_3, x_4, x_5} \psi_1(x_1, x_2) \cdot \psi_2(x_2, x_3) \cdot \psi_3(x_3, x_1) \cdot \psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1)$$

Example of FAQ Computation with Indicator Projections



$$\varphi(x_1) = \sum_{x_2, x_3, x_4, x_5} \psi_1(x_1, x_2) \cdot \psi_2(x_2, x_3) \cdot \psi_3(x_3, x_1) \cdot \psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1)$$

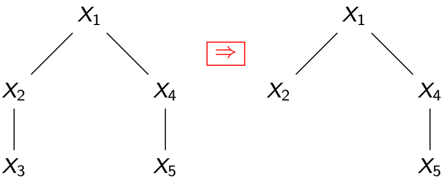
$$\varphi(x_1) = \sum_{x_2, x_4, x_5} \psi_1(x_1, x_2) \cdot \underbrace{\Big(\sum_{x_3} \psi'_1(x_1, x_2) \cdot \psi_2(x_2, x_3) \cdot \psi_3(x_3, x_1) \cdot \psi'_4(x_1) \cdot \psi'_6(x_1) \Big)}_{\psi_7(x_1, x_2)} \cdot$$

$$\psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1)$$

ψ'_1 is an indicator projection of ψ_1 (similarly, ψ'_4 and ψ'_6):

- It has the same support as ψ_1, i.e., same tuples (x_1, x_2)
- $\psi'_1(x_1, x_2) = \mathbf{1}$ even in case $\psi_1(x_1, x_2) \neq \mathbf{1}$ and $\psi_1(x_1, x_2) \neq \mathbf{0}$

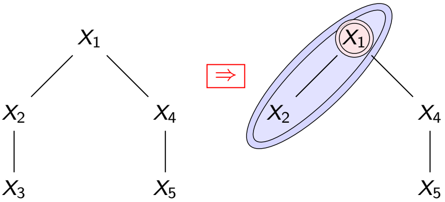
Example of FAQ Computation with Indicator Projections



$$\varphi(x_1) = \sum_{x_2, x_3, x_4, x_5} \psi_1(x_1, x_2) \cdot \psi_2(x_2, x_3) \cdot \psi_3(x_3, x_1) \cdot \psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1)$$

$$\varphi(x_1) = \sum_{x_2, x_4, x_5} \psi_1(x_1, x_2) \cdot \psi_7(x_1, x_2) \cdot \psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1) \qquad \widetilde{O}(N^{1.5})$$

Example of FAQ Computation with Indicator Projections



$$\varphi(x_1) = \sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_2(x_2,x_3)\cdot\psi_3(x_3,x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$

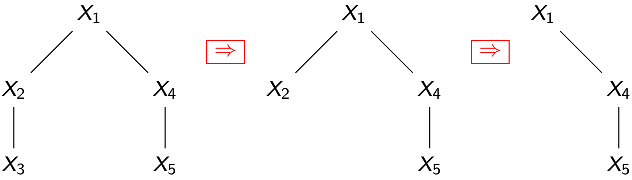
$$\varphi(x_1) = \sum_{x_2,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_7(x_1,x_2)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \sum_{x_4,x_5} \underbrace{\left(\sum_{x_2} \psi_1(x_1,x_2)\cdot\psi_7(x_1,x_2)\cdot\psi_4'(x_1)\cdot\psi_6'(x_1)\right)}_{\psi_8(x_1)}\cdot$$

$$\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$

The indicator projections ψ_4' and ψ_6' are redundant here, as they were already used for computing ϕ_7.

Example of FAQ Computation with Indicator Projections

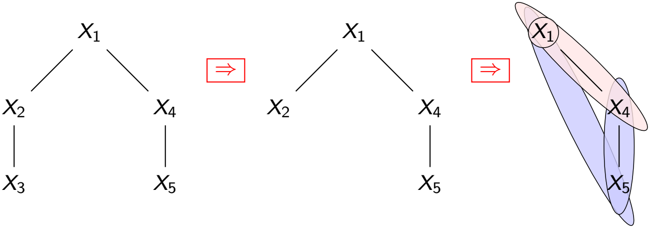


$$\varphi(x_1) = \sum_{x_2, x_3, x_4, x_5} \psi_1(x_1, x_2) \cdot \psi_2(x_2, x_3) \cdot \psi_3(x_3, x_1) \cdot \psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1)$$

$$\varphi(x_1) = \sum_{x_2, x_4, x_5} \psi_1(x_1, x_2) \cdot \psi_7(x_1, x_2) \cdot \psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \sum_{x_4, x_5} \psi_8(x_1) \cdot \psi_4(x_1, x_4) \cdot \psi_5(x_4, x_5) \cdot \psi_6(x_5, x_1) \qquad \widetilde{O}(N)$$

Example of FAQ Computation with Indicator Projections



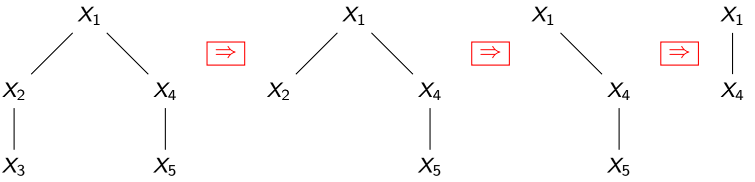
$$\varphi(x_1) = \sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_2(x_2,x_3)\cdot\psi_3(x_3,x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$

$$\varphi(x_1) = \sum_{x_2,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_7(x_1,x_2)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \sum_{x_4,x_5} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N)$$

$$\varphi(x_1) = \sum_{x_4} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\underbrace{\left(\sum_{x_5} \psi'_8(x_1)\cdot\psi'_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)\right)}_{\psi_9(x_1,x_4)}$$

Example of FAQ Computation with Indicator Projections



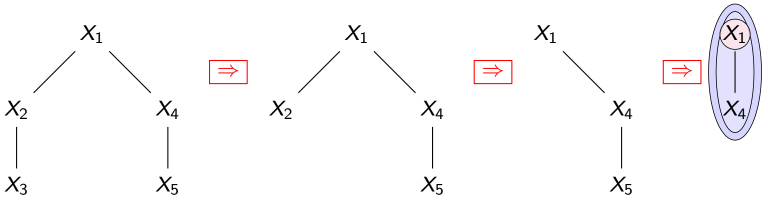
$$\varphi(x_1) = \sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_2(x_2,x_3)\cdot\psi_3(x_3,x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$

$$\varphi(x_1) = \sum_{x_2,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_7(x_1,x_2)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \sum_{x_4,x_5} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N)$$

$$\varphi(x_1) = \sum_{x_4} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_9(x_1,x_4) \qquad \widetilde{O}(N^{1.5})$$

Example of FAQ Computation with Indicator Projections



$$\varphi(x_1) = \sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_2(x_2,x_3)\cdot\psi_3(x_3,x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$

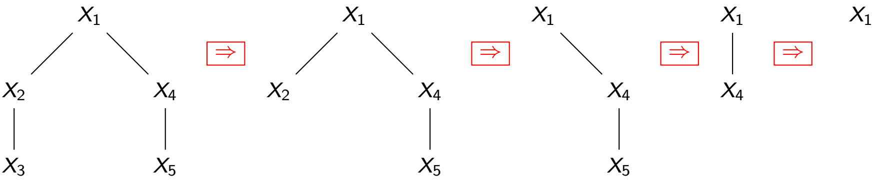
$$\varphi(x_1) = \sum_{x_2,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_7(x_1,x_2)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \sum_{x_4,x_5} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N)$$

$$\varphi(x_1) = \sum_{x_4} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_9(x_1,x_4) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \psi_8(x_1)\cdot\Big(\underbrace{\sum_{x_4} \psi'_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_9(x_1,x_4)}_{\psi_{10}(x_1)}\Big)$$

Example of FAQ Computation with Indicator Projections



$$\varphi(x_1) = \sum_{x_2,x_3,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_2(x_2,x_3)\cdot\psi_3(x_3,x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1)$$

$$\varphi(x_1) = \sum_{x_2,x_4,x_5} \psi_1(x_1,x_2)\cdot\psi_7(x_1,x_2)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \sum_{x_4,x_5} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_5(x_4,x_5)\cdot\psi_6(x_5,x_1) \qquad \widetilde{O}(N)$$

$$\varphi(x_1) = \sum_{x_4} \psi_8(x_1)\cdot\psi_4(x_1,x_4)\cdot\psi_9(x_1,x_4) \qquad \widetilde{O}(N^{1.5})$$

$$\varphi(x_1) = \psi_8(x_1)\cdot\psi_{10}(x_1) \qquad \widetilde{O}(N)$$

Is Factorized Aggregate Computation Practical?

A glimpse at performance experiments [ANNOS18b]

Retailer dataset (records)	excerpt (17M)	full (86M)
PostgreSQL computing the join	50.63 sec	216.56 sec
Aggregates for a linear regression model		
FDB computing join+aggregates	25.51 sec	380.31 sec
Number of aggregates (scalar+group-by)	595+2,418	595+145k
Aggregates for a polynomial regression model		
FDB computing join+aggregates	132.43 sec	1,819.80 sec
Number of aggregates (scalar+group-by)	158k+742k	158k+37M

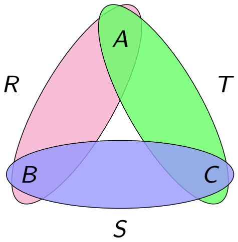
In this experiment:

- FDB only used one core of a commodity machine
- For both PostgreSQL and FDB, the dataset was entirely in memory
- The aggregates represent gradients (or parts thereof) used for learning degree 1 and 2 polynomial regression models

Outline of Part 2: Aggregates

Problem Setting

Maintain the triangle count Q
under single-tuple updates to R, S, and T!



Q counts the number of tuples in the join of R, S, and T.

$$Q = \sum_{a,b,c} R(a,b) \cdot S(b,c) \cdot T(c,a)$$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	3

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	3

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	3

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$3 \cdot 1 \cdot 3 = 9$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	3

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$3 \cdot 1 \cdot 3 = 9$

$\downarrow$

$Q(\mathbf{D})$

$\emptyset$	
$()$	$4 + 6 + 9 = 19$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	3

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$3 \cdot 1 \cdot 3 = 9$

$\uparrow$

$\delta R = \{(a_2, b_1) \mapsto -2\}$

A	B	
a_2	b_1	-2

$\downarrow$

$Q(\mathbf{D})$

$\emptyset$	
$()$	$4 + 6 + 9 = 19$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	3

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$3 \cdot 1 \cdot 3 = 9$

$\uparrow$

$\delta R = \{(a_2, b_1) \mapsto -2\}$

A	B	
a_2	b_1	-2

$\downarrow$

$Q(\mathbf{D})$

$\emptyset$	
$(\,)$	$4 + 6 + 9 = 19$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	1

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$3 \cdot 1 \cdot 3 = 9$

$\delta R = \{(a_2, b_1) \mapsto -2\}$

A	B	
a_2	b_1	-2

$Q(\mathbf{D})$

$\emptyset$	
$()$	$4 + 6 + 9 = 19$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	1

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$3 \cdot 1 \cdot 3 = 9$

$\uparrow$

$\delta R = \{(a_2, b_1) \mapsto -2\}$

A	B	
a_2	b_1	-2

$\downarrow$

$Q(\mathbf{D})$

$\emptyset$	
$()$	$4 + 6 + 9 = 19$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	1

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$1 \cdot 1 \cdot 3 = 3$

$\delta R = \{(a_2, b_1) \mapsto -2\}$

A	B	
a_2	b_1	-2

$Q(\mathbf{D})$

$\emptyset$	
$()$	$4 + 6 + 9 = 19$

Updates to the Triangle Count

R

A	B	
a_1	b_1	2
a_2	b_1	1

S

B	C	
b_1	c_1	2
b_1	c_2	1

T

C	A	
c_1	a_1	1
c_2	a_1	3
c_2	a_2	3

$R \cdot S \cdot T$

A	B	C	
a_1	b_1	c_2	$2 \cdot 2 \cdot 1 = 4$
a_1	b_1	c_2	$2 \cdot 1 \cdot 3 = 6$
a_2	b_1	c_3	$1 \cdot 1 \cdot 3 = 3$

$\uparrow$

$\delta R = \{(a_2, b_1) \mapsto -2\}$

A	B	
a_2	b_1	-2

$\downarrow$

$Q(\mathbf{D})$

$\emptyset$	
$()$	$4 + 6 + 3 = 13$

Data Updates need the Additive Inverse

Data updates can be inserts (tuples with positive multiplicity) and deletes (tuples with negative multiplicity):

- Semirings are enough if we only want inserts or no updates

Recall that FAQs use commutative semirings $(\mathbf{D}, \oplus, \otimes)$:

- $(\mathbf{D}, \oplus)$ is a commutative monoid with identity element $\mathbf{0}$:
  - $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - $\mathbf{0} \oplus a = a \oplus \mathbf{0} = a$
  - $a \oplus b = b \oplus a$
- $(\mathbf{D}, \otimes)$ is a commutative monoid with identity element $\mathbf{1}$:
  - $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - $\mathbf{1} \otimes a = a \otimes \mathbf{1} = a$
  - $a \oplus b = b \oplus a$
- Multiplication distributes over addition:
  - $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
- Multiplication by $\mathbf{0}$ annihilates $\mathbf{D}$:
  - $\mathbf{0} \otimes a = a \otimes \mathbf{0} = \mathbf{0}$

From Semirings to Rings

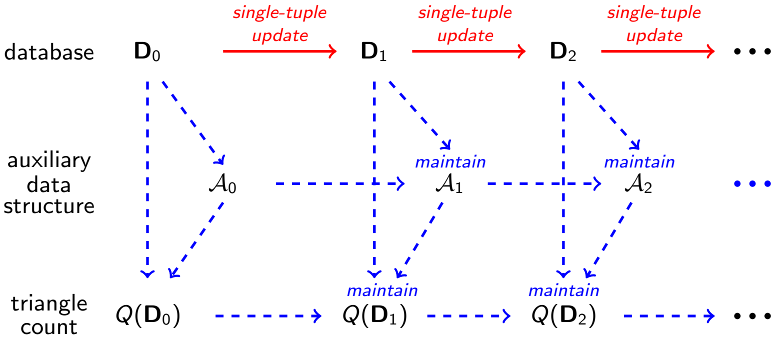
We need a commutative ring $(\mathbf{D}, \oplus, \otimes)$ if we want to support deletes as well:

- $(\mathbf{D}, \oplus)$ is an abelian group with identity element $\mathbf{0}$:
  - $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - $\mathbf{0} \oplus a = a \oplus \mathbf{0} = a$
  - $a \oplus b = b \oplus a$
  - $\exists -a \in \mathbf{D} : a \oplus (-a) = (-a) \oplus a = \mathbf{0}$
- $(\mathbf{D}, \otimes)$ is a commutative monoid with identity element $\mathbf{1}$:
  - $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - $\mathbf{1} \otimes a = a \otimes \mathbf{1} = a$
  - $a \oplus b = b \oplus a$
- Multiplication distributes over addition:
  - $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
- Multiplication by $\mathbf{0}$ annihilates $\mathbf{D}$:
  - $\mathbf{0} \otimes a = a \otimes \mathbf{0} = \mathbf{0}$

Examples: $\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}, \mathbb{R}^n$, polynomial ring.

We used the ring $(\mathbb{Z}, +, *)$ in our previous example.

The Maintenance Problem



Given a current database $\mathbf{D}$ and a single-tuple update,
what are the time and space complexities for maintaining $Q(\mathbf{D})$?

Much Ado about Triangles

The Triangle Query Served as Milestone in Many Fields

- Worst-case optimal join algorithms *[Algorithmica 1997, SIGMOD R. 2013]*
- Parallel query evaluation *[Found. & Trends DB 2018]*
- Randomized approximation in static settings *[FOCS 2015]*
- Randomized approximation in data streams
  [SODA 2002, COCOON 2005, PODS 2006, PODS 2016, Theor. Comput. Sci. 2017]

Investigation of Answering Queries under Updates

- Theoretical developments *[PODS 2017, ICDT 2018]*
- Systems developments *[F. & T. DB 2012, VLDB J. 2014, SIGMOD 2017, 2018]*
- Lower bounds *[STOC 2015, ICM 2018]*

Naïve Maintenance

"Compute from scratch!"

$$\delta R = \{(a', b') \mapsto m\}$$

$$\sum_{a,b,c} \big[\underbrace{R(a,b) + \delta R(a,b)}_{newR} \big] \cdot S(b,c) \cdot T(c,a)$$

$$=$$

$$\sum_{a,b,c} \; newR(a,b) \cdot S(b,c) \cdot T(c,a)$$

Maintenance Complexity

- Time: $\mathcal{O}(|\mathbf{D}|^{1.5})$ using worst-case optimal join algorithms
- Space: $\mathcal{O}(|\mathbf{D}|)$ to store input relations

Classical Incremental View Maintenance [CY12]

"*Compute the difference!*"

$$\delta R = \{(a', b') \mapsto m\}$$

$$\sum_{a,b,c} \left[R(a,b) + \delta R(a,b)\right] \cdot S(b,c) \cdot T(c,a)$$
$$=$$
$$\sum_{a,b,c} R(a,b) \cdot S(b,c) \cdot T(c,a)$$
$$+$$
$$\delta R(a', b') \cdot \sum_{c} S(b', c) \cdot T(c, a')$$

Maintenance Complexity

- Time: $\mathcal{O}(|\mathbf{D}|)$ to intersect C-values from S and T
- Space: $\mathcal{O}(|\mathbf{D}|)$ to store input relations

Factorized Incremental View Maintenance [NO18]

"*Compute the difference by using pre-materialized views!*"

$$\delta R = \{(a', b') \mapsto m\}$$

Pre-materialize $V_{ST}(b, a) = \sum_c S(b, c) \cdot T(c, a)$!

$$\sum_{a,b,c} \left[R(a, b) + \delta R(a, b)\right] \cdot S(b, c) \cdot T(c, a)$$
$$=$$
$$\sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a)$$
$$+$$
$$\delta R(a', b') \cdot V_{ST}(b', a')$$

Maintenance Complexity

- Time for updates to R: $\mathcal{O}(1)$ to look up in V_{ST}
- Time for updates to S and T: $\mathcal{O}(|\mathbf{D}|)$ to maintain V_{ST}
- Space: $\mathcal{O}(|\mathbf{D}|^2)$ to store input relations and V_{ST}

Closing the Complexity Gap

Complexity bounds for the maintenance of the triangle count

Known Upper Bound

Maintenance Time: $\mathcal{O}(|\mathbf{D}|)$

Space: $\mathcal{O}(|\mathbf{D}|)$

Lower Bound

Amortized maintenance time: not $\mathcal{O}(|\mathbf{D}|^{0.5-\gamma})$ for any $\gamma > 0$

(under reasonable complexity theoretic assumptions)

Closing the Complexity Gap

Complexity bounds for the maintenance of the triangle count

Known Upper Bound

Maintenance Time: $\mathcal{O}(|\mathbf{D}|)$

Space: $\mathcal{O}(|\mathbf{D}|)$

Can the triangle count be maintained in sublinear time?

Lower Bound

Amortized maintenance time: not $\mathcal{O}(|\mathbf{D}|^{0.5-\gamma})$ for any $\gamma > 0$

(under reasonable complexity theoretic assumptions)

Closing the Complexity Gap

Complexity bounds for the maintenance of the triangle count

Known Upper Bound

Maintenance Time: $\mathcal{O}(|\mathbf{D}|)$

Space: $\mathcal{O}(|\mathbf{D}|)$

Can the triangle count be maintained in sublinear time?

Yes!

IVM$^{\varepsilon}$ [KNNOZ19]

Amortized maintenance time:

$\mathcal{O}(|\mathbf{D}|^{0.5})$

This is worst-case optimal!

Lower Bound

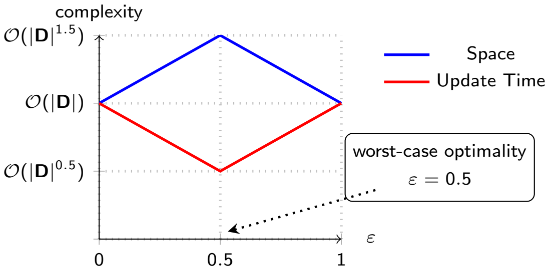
Amortized maintenance time: not $\mathcal{O}(|\mathbf{D}|^{0.5-\gamma})$ for any $\gamma > 0$

(under reasonable complexity theoretic assumptions)

IVM$^\varepsilon$ Exhibits a Time-Space Tradeoff [KNNOZ19]

Given $\varepsilon \in [0,1]$ and a database $\mathbf{D}$, IVM$^\varepsilon$ maintains the triangle count with

- $\mathcal{O}(|\mathbf{D}|^{\max\{\varepsilon,1-\varepsilon\}})$ amortized update time
- $\mathcal{O}(|\mathbf{D}|^{1+\min\{\varepsilon,1-\varepsilon\}})$ space
- $\mathcal{O}(|\mathbf{D}|^{3/2})$ preprocessing time
- $\mathcal{O}(1)$ answer time.



Known maintenance approaches are recovered by IVM$^\varepsilon$.

Main Ideas in IVM$^{\varepsilon}$

- Compute the difference like in classical IVM!

- Materialize views like in Factorized IVM!

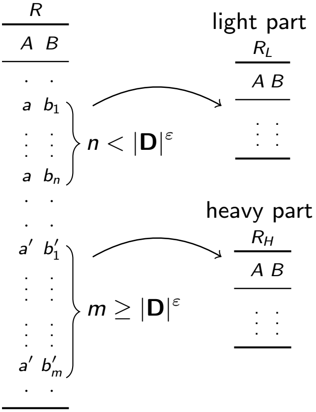
- New ingredient: Use adaptive processing based on data skew!

  $\Longrightarrow$ Treat *heavy* values differently from *light* values!

Quick Look inside IVM$^{\varepsilon}$

Partition R into

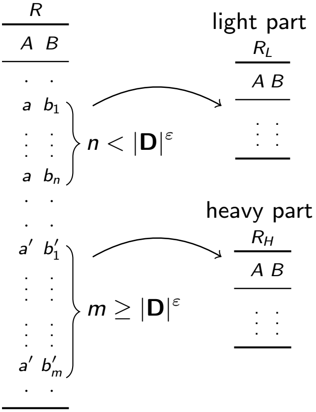
- a light part $R_L = \{t \in R \mid |\sigma_{A=t.A}| < |\mathbf{D}|^{\varepsilon}\}$,
- a heavy part $R_H = R \backslash R_L$!



Quick Look inside IVM$^{\varepsilon}$

Partition R into

- a light part $R_L = \{t \in R \mid |\sigma_{A=t.A}| < |\mathbf{D}|^{\varepsilon}\}$,
- a heavy part $R_H = R \setminus R_L$!



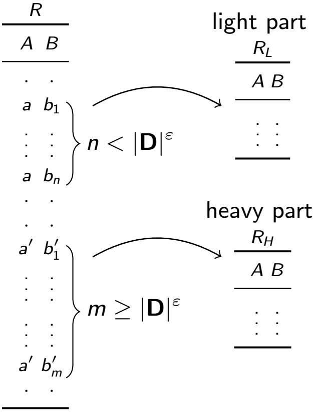
Derived Bounds

- for all A-values a: $|\sigma_{A=a} R_L| < |\mathbf{D}|^{\varepsilon}$
- $|\pi_A R_H| \leq |\mathbf{D}|^{1-\varepsilon}$

Quick Look inside IVM$^\varepsilon$

Partition R into

- a light part
  $R_L = \{t \in R \mid |\sigma_{A=t.A}| < |\mathbf{D}|^\varepsilon\}$,
- a heavy part $R_H = R \backslash R_L$!



Derived Bounds

- for all A-values a:
  $|\sigma_{A=a} R_L| < |\mathbf{D}|^\varepsilon$
- $|\pi_A R_H| \leq |\mathbf{D}|^{1-\varepsilon}$

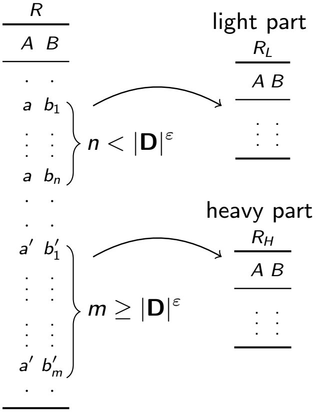
Likewise, partition

- $S = S_L \cup S_H$ based on B, and
- $T = T_L \cup T_H$ based on C!

Quick Look inside IVM$^\varepsilon$

Partition R into

- a light part
  $R_L = \{t \in R \mid |\sigma_{A=t.A}| < |\mathbf{D}|^\varepsilon\}$,
- a heavy part $R_H = R \setminus R_L$!



Derived Bounds

- for all A-values a:
  $|\sigma_{A=a} R_L| < |\mathbf{D}|^\varepsilon$
- $|\pi_A R_H| \leq |\mathbf{D}|^{1-\varepsilon}$

Likewise, partition

- $S = S_L \cup S_H$ based on B, and
- $T = T_L \cup T_H$ based on C!

Q is the sum of skew-aware views

$$R_U(a, b) \cdot S_V(b, c) \cdot T_W(c, a)$$

with $U, V, W \in \{L, H\}$.

Adaptive Maintenance Strategy

Given an update $\delta R_* = \{(a', b') \mapsto m\}$, compute the difference for each skew-aware view using different strategies:

Skew-aware View	Evaluation from left to right	Time
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_L(c,a)$	$\delta R_*(a', b') \cdot \sum_{c} S_L(b', c) \cdot T_L(c, a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$

Adaptive Maintenance Strategy

Given an update $\delta R_* = \{(a', b') \mapsto m\}$, compute the difference for each skew-aware view using different strategies:

Skew-aware View	Evaluation from left to right	Time
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_L(c,a)$	$\delta R_*(a',b') \cdot \sum_c S_L(b',c) \cdot T_L(c,a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_H(b,c) \cdot T_H(c,a)$	$\delta R_*(a',b') \cdot \sum_c T_H(c,a') \cdot S_H(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$

Adaptive Maintenance Strategy

Given an update $\delta R_* = \{(a', b') \mapsto m\}$, compute the difference for each skew-aware view using different strategies:

Skew-aware View	Evaluation from left to right	Time
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_L(c,a)$	$\delta R_*(a',b') \cdot \sum_c S_L(b',c) \cdot T_L(c,a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_H(b,c) \cdot T_H(c,a)$	$\delta R_*(a',b') \cdot \sum_c T_H(c,a') \cdot S_H(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_H(c,a)$	$\delta R_*(a',b') \cdot \sum_c S_L(b',c) \cdot T_H(c,a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
	or	
	$\delta R_*(a',b') \cdot \sum_c T_H(c,a') \cdot S_L(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$

Adaptive Maintenance Strategy

Given an update $\delta R_* = \{(a', b') \mapsto m\}$, compute the difference for each skew-aware view using different strategies:

Skew-aware View	Evaluation from left to right	Time
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_L(c,a)$	$\delta R_*(a',b') \cdot \sum_c S_L(b',c) \cdot T_L(c,a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_H(b,c) \cdot T_H(c,a)$	$\delta R_*(a',b') \cdot \sum_c T_H(c,a') \cdot S_H(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_H(c,a)$	$\delta R_*(a',b') \cdot \sum_c S_L(b',c) \cdot T_H(c,a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
	or	
	$\delta R_*(a',b') \cdot \sum_c T_H(c,a') \cdot S_L(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_H(b,c) \cdot T_L(c,a)$	$\delta R_*(a',b') \cdot V_{ST}(b',a')$	$\mathcal{O}(1)$

Adaptive Maintenance Strategy

Given an update $\delta R_* = \{(a', b') \mapsto m\}$, compute the difference for each skew-aware view using different strategies:

Skew-aware View	Evaluation from left to right	Time
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_L(c,a)$	$\delta R_*(a',b') \cdot \sum_c S_L(b',c) \cdot T_L(c,a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_H(b,c) \cdot T_H(c,a)$	$\delta R_*(a',b') \cdot \sum_c T_H(c,a') \cdot S_H(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_L(b,c) \cdot T_H(c,a)$	$\delta R_*(a',b') \cdot \sum_c S_L(b',c) \cdot T_H(c,a')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
	or	
	$\delta R_*(a',b') \cdot \sum_c T_H(c,a') \cdot S_L(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$
$\sum_{a,b,c} R_*(a,b) \cdot S_H(b,c) \cdot T_L(c,a)$	$\delta R_*(a',b') \cdot V_{ST}(b',a')$	$\mathcal{O}(1)$

Overall update time: $\mathcal{O}(|\mathbf{D}|^{\max\{\varepsilon, 1-\varepsilon\}})$

Materialized Auxiliary Views

$$V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$$
$$V_{ST}(b,a) = \sum_c S_H(b,c) \cdot T_L(c,a)$$
$$V_{TR}(c,b) = \sum_a T_H(c,a) \cdot R_L(a,b)$$

- Maintenance of $V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$

Update	Compute the difference for V_{RS}	Time
$\delta R_H = \{(a',b') \mapsto m\}$	$\delta R_H(a',b') \cdot S_L(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
$\delta S_L = \{(b',c') \mapsto m\}$	$\delta S_L(b',c') \cdot R_H(a,b')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$

Materialized Auxiliary Views

$$V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$$
$$V_{ST}(b,a) = \sum_c S_H(b,c) \cdot T_L(c,a)$$
$$V_{TR}(c,b) = \sum_a T_H(c,a) \cdot R_L(a,b)$$

- Maintenance of $V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$

Update	Compute the difference for V_{RS}	Time
$\delta R_H = \{(a',b') \mapsto m\}$	$\delta R_H(a',b') \cdot S_L(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
$\delta S_L = \{(b',c') \mapsto m\}$	$\delta S_L(b',c') \cdot R_H(a,b')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$

- Size of $V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$

$$\begin{aligned} |V_{RS}(a,c)| &\leq |R_H| \cdot \max_b\{|S_L(b,c)|\} &= \mathcal{O}(|\mathbf{D}|^{1+\varepsilon}) \\ |V_{RS}(a,c)| &\leq |S_L| \cdot \max_b\{|R_H(a,b)|\} &= \mathcal{O}(|\mathbf{D}|^{1+(1-\varepsilon)}) \end{aligned}$$

Materialized Auxiliary Views

$$V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$$
$$V_{ST}(b,a) = \sum_c S_H(b,c) \cdot T_L(c,a)$$
$$V_{TR}(c,b) = \sum_a T_H(c,a) \cdot R_L(a,b)$$

- Maintenance of $V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$

Update	Compute the difference for V_{RS}	Time
$\delta R_H = \{(a',b') \mapsto m\}$	$\delta R_H(a',b') \cdot S_L(b',c)$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{\varepsilon})$
$\delta S_L = \{(b',c') \mapsto m\}$	$\delta S_L(b',c') \cdot R_H(a,b')$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\varepsilon})$

- Size of $V_{RS}(a,c) = \sum_b R_H(a,b) \cdot S_L(b,c)$

$$\begin{array}{ccccc} |V_{RS}(a,c)| & \leq & |R_H| \cdot \max_b\{|S_L(b,c)|\} & = & \mathcal{O}(|\mathbf{D}|^{1+\varepsilon}) \\ |V_{RS}(a,c)| & \leq & |S_L| \cdot \max_b\{|R_H(a,b)|\} & = & \mathcal{O}(|\mathbf{D}|^{1+(1-\varepsilon)}) \end{array}$$

- Overall: Update Time $\mathcal{O}(|\mathbf{D}|^{\max\{\varepsilon,1-\varepsilon\}})$ and Space $\mathcal{O}(|\mathbf{D}|^{1+\min\{\varepsilon,1-\varepsilon\}})$

Rebalancing Partitions

Full details available in the paper [KNNOZ19]

- Updates can change the frequencies of values and the heavy/light threshold!
- This may require rebalancing of partitions:

  $\Longrightarrow$ Minor rebalancing: Transfer tuples from one to the other part of the same relation!

  $\Longrightarrow$ Major rebalancing: Recompute partitions and views from scratch!
- Both forms of rebalancing require superlinear time.
- The rebalancing times amortize over sequences of updates.

Lower Bound for Maintaining the Triangle Count

- The lower bound already holds for the Boolean Triangle Detection Problem, which is a special case of the Triangle Count.

For any $\gamma > 0$, there is no algorithm that incrementally maintains the Triangle Detection Problem with

amortized update time	answer time
$\mathcal{O}(\lvert\mathbf{D}\rvert^{\frac{1}{2}-\gamma})$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\gamma})$

unless the Online Vector-Matrix-Vector Multiplication (OuMv) Conjecture fails.

Lower Bound for Maintaining the Triangle Count

- The lower bound already holds for the Boolean Triangle Detection Problem, which is a special case of the Triangle Count.

For any $\gamma > 0$, there is no algorithm that incrementally maintains the Triangle Detection Problem with

amortized update time	answer time
$\mathcal{O}(\lvert\mathbf{D}\rvert^{\frac{1}{2}-\gamma})$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\gamma})$

unless the Online Vector-Matrix-Vector Multiplication (OuMv) Conjecture fails.

The OuMv Problem

Input: An $n \times n$ Boolean matrix $\mathbf{M}$ and n pairs $(\mathbf{u}_1, \mathbf{v}_1), \ldots, (\mathbf{u}_n, \mathbf{v}_n)$ of Boolean column-vectors of size n arriving one after the other.

Goal: After seeing each pair $(\mathbf{u}_r, \mathbf{v}_r)$, output $\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r$

The OuMv Conjecture [HKNS15]

For any $\gamma > 0$, there is no algorithm that solves the OuMv Problem in time $\mathcal{O}(n^{3-\gamma})$.

Proof Sketch (1/2)

- Assume there is an algorithm $\mathcal{A}$ maintaining Triangle Detection with

amortized update time	answer time
$\mathcal{O}(\lvert\mathbf{D}\rvert^{\frac{1}{2}-\gamma})$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\gamma})$

  for some $\gamma > 0$.

- Goal: Design an algorithm $\mathcal{B}$ using algorithm $\mathcal{A}$ as oracle that solves OuMv in subcubic time. $\Longrightarrow$ **Contradicts the OuMv Conjecture!**

Proof Sketch (1/2)

- Assume there is an algorithm $\mathcal{A}$ maintaining Triangle Detection with

amortized update time	answer time
$\mathcal{O}(\lvert\mathbf{D}\rvert^{\frac{1}{2}-\gamma})$	$\mathcal{O}(\lvert\mathbf{D}\rvert^{1-\gamma})$

  for some $\gamma > 0$.

- Goal: Design an algorithm $\mathcal{B}$ using algorithm $\mathcal{A}$ as oracle that solves OuMv in subcubic time. $\Longrightarrow$ **Contradicts the OuMv Conjecture!**

Algorithm $\mathcal{B}$

- Use relation S to encode the matrix $\mathbf{M}$.
- In each round $r \in [n]$:
  - Use relations R and T to encode $\mathbf{u}_r$ and $\mathbf{v}_r$, respectively, such that

$$\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r = 1 \qquad \text{if and only if} \qquad R \bowtie S \bowtie T \neq \emptyset$$

  - Check whether $R \bowtie S \bowtie T$ contains a triangle.

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

(2) In each round $r \in [n]$:

- Delete all tuples in $R(A, B)$ and $T(C, A)$

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

(2) In each round $r \in [n]$:
- Delete all tuples in $R(A, B)$ and $T(C, A)$
- Insert at most n tuples into $R(A, B)$ and $T(C, A)$ such that $R = \{(a, i) \mid \mathbf{u}_r(i) = 1\}$ and $T = \{(i, a) \mid \mathbf{v}_r(i) = 1\}$ for a constant a

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

(2) In each round $r \in [n]$:

- Delete all tuples in $R(A, B)$ and $T(C, A)$
- Insert at most n tuples into $R(A, B)$ and $T(C, A)$ such that $R = \{(a, i) \mid \mathbf{u}_r(i) = 1\}$ and $T = \{(i, a) \mid \mathbf{v}_r(i) = 1\}$ for a constant a
- Check $R \bowtie S \bowtie T \neq \emptyset$: This holds iff $\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r = 1$
  iff $\exists i, j \in [n]$ with $\mathbf{u}_r(i) = 1$, $\mathbf{M}(i, j) = 1$, and $\mathbf{v}_r(j) = 1$

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B,C) = \{(i,j) \mid \mathbf{M}(i,j) = 1\}$

(2) In each round $r \in [n]$:

- Delete all tuples in $R(A,B)$ and $T(C,A)$
- Insert at most n tuples into $R(A,B)$ and $T(C,A)$ such that
  $R = \{(a,i) \mid \mathbf{u}_r(i) = 1\}$ and $T = \{(i,a) \mid \mathbf{v}_r(i) = 1\}$ for a constant a
- Check $R \bowtie S \bowtie T \neq \emptyset$: This holds iff $\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r = 1$
  iff $\exists i,j \in [n]$ with $\mathbf{u}_r(i) = 1$, $\mathbf{M}(i,j) = 1$, and $\mathbf{v}_r(j) = 1$

Time analysis:

- The database size is $\mathcal{O}(n^2)$.

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

(2) In each round $r \in [n]$:

- Delete all tuples in $R(A, B)$ and $T(C, A)$
- Insert at most n tuples into $R(A, B)$ and $T(C, A)$ such that $R = \{(a, i) \mid \mathbf{u}_r(i) = 1\}$ and $T = \{(i, a) \mid \mathbf{v}_r(i) = 1\}$ for a constant a
- Check $R \bowtie S \bowtie T \neq \emptyset$: This holds iff $\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r = 1$
  iff $\exists i, j \in [n]$ with $\mathbf{u}_r(i) = 1$, $\mathbf{M}(i, j) = 1$, and $\mathbf{v}_r(j) = 1$

Time analysis:

- The database size is $\mathcal{O}(n^2)$.
- Time in (1): $\mathcal{O}(\underbrace{n^2}_{\#\text{updates}} \cdot \underbrace{(n^2)^{\frac{1}{2}-\gamma}}_{\text{update time}}) = \mathcal{O}(n^2 \cdot n^{1-2\gamma}) = \mathcal{O}(n^{3-2\gamma})$

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

(2) In each round $r \in [n]$:

- Delete all tuples in $R(A, B)$ and $T(C, A)$
- Insert at most n tuples into $R(A, B)$ and $T(C, A)$ such that $R = \{(a, i) \mid \mathbf{u}_r(i) = 1\}$ and $T = \{(i, a) \mid \mathbf{v}_r(i) = 1\}$ for a constant a
- Check $R \bowtie S \bowtie T \neq \emptyset$: This holds iff $\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r = 1$
  iff $\exists i, j \in [n]$ with $\mathbf{u}_r(i) = 1$, $\mathbf{M}(i, j) = 1$, and $\mathbf{v}_r(j) = 1$

Time analysis:

- The database size is $\mathcal{O}(n^2)$.
- Time in (1): $\mathcal{O}(\underbrace{n^2}_{\#\text{updates}} \cdot \underbrace{(n^2)^{\frac{1}{2}-\gamma}}_{\text{update time}}) = \mathcal{O}(n^2 \cdot n^{1-2\gamma}) = \mathcal{O}(n^{3-2\gamma})$
- Time for each round in (2): $\mathcal{O}(\underbrace{4n}_{\#\text{updates}} \cdot \underbrace{(n^2)^{\frac{1}{2}-\gamma}}_{\text{update time}} + \underbrace{(n^2)^{1-\gamma}}_{\text{answer time}}) = \mathcal{O}(n^{2-2\gamma})$

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

(2) In each round $r \in [n]$:

- Delete all tuples in $R(A, B)$ and $T(C, A)$
- Insert at most n tuples into $R(A, B)$ and $T(C, A)$ such that $R = \{(a, i) \mid \mathbf{u}_r(i) = 1\}$ and $T = \{(i, a) \mid \mathbf{v}_r(i) = 1\}$ for a constant a
- Check $R \bowtie S \bowtie T \neq \emptyset$: This holds iff $\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r = 1$
  iff $\exists i, j \in [n]$ with $\mathbf{u}_r(i) = 1$, $\mathbf{M}(i, j) = 1$, and $\mathbf{v}_r(j) = 1$

Time analysis:

- The database size is $\mathcal{O}(n^2)$.
- Time in (1): $\mathcal{O}(\underbrace{n^2}_{\#\text{updates}} \cdot \underbrace{(n^2)^{\frac{1}{2}-\gamma}}_{\text{update time}}) = \mathcal{O}(n^2 \cdot n^{1-2\gamma}) = \mathcal{O}(n^{3-2\gamma})$
- Time for each round in (2): $\mathcal{O}(\underbrace{4n}_{\#\text{updates}} \cdot \underbrace{(n^2)^{\frac{1}{2}-\gamma}}_{\text{update time}} + \underbrace{(n^2)^{1-\gamma}}_{\text{answer time}}) = \mathcal{O}(n^{2-2\gamma})$
- Time for n rounds in (2): $\mathcal{O}(n \cdot n^{2-2\gamma}) = \mathcal{O}(n^{3-2\gamma})$

Proof Sketch (2/2)

Algorithm $\mathcal{B}$ in more detail:

(1) Insert at most n^2 tuples into S such that $S(B, C) = \{(i, j) \mid \mathbf{M}(i, j) = 1\}$

(2) In each round $r \in [n]$:

- Delete all tuples in $R(A, B)$ and $T(C, A)$
- Insert at most n tuples into $R(A, B)$ and $T(C, A)$ such that $R = \{(a, i) \mid \mathbf{u}_r(i) = 1\}$ and $T = \{(i, a) \mid \mathbf{v}_r(i) = 1\}$ for a constant a
- Check $R \bowtie S \bowtie T \neq \emptyset$: This holds iff $\mathbf{u}_r^T \mathbf{M} \mathbf{v}_r = 1$
  iff $\exists i, j \in [n]$ with $\mathbf{u}_r(i) = 1$, $\mathbf{M}(i, j) = 1$, and $\mathbf{v}_r(j) = 1$

Time analysis:

- The database size is $\mathcal{O}(n^2)$.
- Time in (1): $\mathcal{O}(\underbrace{n^2}_{\#\text{updates}} \cdot \underbrace{(n^2)^{\frac{1}{2}-\gamma}}_{\text{update time}}) = \mathcal{O}(n^2 \cdot n^{1-2\gamma}) = \mathcal{O}(n^{3-2\gamma})$
- Time for each round in (2): $\mathcal{O}(\underbrace{4n}_{\#\text{updates}} \cdot \underbrace{(n^2)^{\frac{1}{2}-\gamma}}_{\text{update time}} + \underbrace{(n^2)^{1-\gamma}}_{\text{answer time}}) = \mathcal{O}(n^{2-2\gamma})$
- Time for n rounds in (2): $\mathcal{O}(n \cdot n^{2-2\gamma}) = \mathcal{O}(n^{3-2\gamma})$
- Overall time: **subcubic!** $\mathcal{O}(n^{3-2\gamma})$

Notes on the OuMv Conjecture

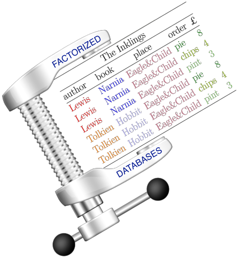
The hardness of many dynamic problems is based on Online Matrix-vector multiplication problem (OMv)

- OuMv is at least as hard as OMv

Examples: [HKNS15]

- source-target reachability
- source-target shortest path (in unweighted graphs)
- transitive closure

Outline of Part 2: Aggregates



References

D99 **Bucket Elimination: A Unifying Framework for Reasoning.**
Dechter. In Artif. Intell. 113(1-2): 41-85 (1999)
`https://www.ics.uci.edu/~dechter/publications/r48b.pdf`

AM00 **The generalized distributive law.**
Aji, McEliece. In IEEE Trans. Information Theory 46(2): 325-343 (2000)
`https://ieeexplore.ieee.org/document/825794/`

CY12 **Materialized Views.**
Chirkova, Yang. In Foundations & Trends DB, 4(4):295–405, 2012.
`https://ieeexplore.ieee.org/document/8187272?arnumber=8187272`

BKOZ13 **Aggregation and Ordering in Factorised Databases.**
Bakibayev, Kocisky, Olteanu, Zavodny. In PVLDB 2013.
`https://arxiv.org/abs/1307.0441`

HKNS15 **Unifying and Strengthening Hardness for Dynamic Problems via the Online Matrix-Vector Multiplication Conjecture.**
Henzinger, Krinninger, Nanongkai, Saranurak. In STOC 2015.
`https://arxiv.org/abs/1511.06773`

ANR16 **FAQ: Questions Asked Frequently.**
Abo Khamis, Ngo, Rudra. In PODS 2016.
`https://arxiv.org/abs/1504.04044`

References

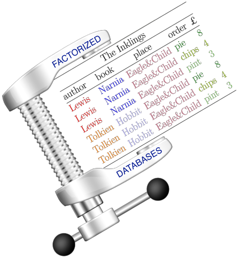
OS16 **Factorized Databases.**
Olteanu, Schleich. In SIGMOD Record 45(2): 5-16 (2016).
`https://dl.acm.org/citation.cfm?doid=3003665.3003667`

NO18 **Incremental View Maintenance with Triple Lock Factorization Benefits.**
Nikolic, Olteanu. In SIGMOD 2018.
`https://arxiv.org/abs/1703.07484`

ANNOS18b **AC/DC: In-Database Learning Thunderstruck.**
Abo Khamis, Ngo, Nguyen, Olteanu, Schleich. In DEEM@SIGMOD 2018.
`https://arxiv.org/abs/1803.07480`

KNNOZ19 **Counting triangles under updates in worst-case optimal time.**
Kara, Ngo, Nikolic, Olteanu, Zhang. In ICDT 2019.
`http://arxiv.org/abs/1804.02780`

Outline of Part 2: Aggregates



QUIZ on Aggregates (1/3)

For each of of the following functional aggregate queries:

1. Give a hypertree decomposition and variable order.
2. If you were to compute it as stated below (with all sums done after the products), what would be its time complexity? (Assume all functions have the same size.)
3. Is there an equivalent rewriting of φ that would allow for quadratic or even linear time complexity?

The *n*-hop query:

$$\varphi(x_1, x_{n+1}) = \sum_{x_2,\ldots,x_n} \psi_1(X_1, X_2) \cdot \psi_2(X_2, X_3) \cdot \psi_3(X_3, X_4) \cdot \ldots \cdot \psi_n(X_n, X_{n+1}).$$

QUIZ on Aggregates (2/3)

For each of of the following functional aggregate queries:

1. Give a hypertree decomposition and variable order.
2. If you were to compute it as stated below (with all sums done after the products), what would be its time complexity? Assume all functions have the same size.
3. Is there an equivalent rewriting of φ that would allow for quadratic or even linear time complexity?

Query:

$$\varphi = \sum_a \sum_b \sum_c \sum_f \sum_d \sum_e \psi_1(a, b) \cdot \psi_2(a, c) \cdot \psi_3(c, d) \cdot \psi_4(b, c, e) \cdot \psi_5(e, f).$$

QUIZ on Aggregates (3/3)

Give the update time and necessary space for the maintenance of the following FAQs as a function of the database size and the heavy/light threshold parameter $\epsilon \in [0, 1]$:

- $\varphi_1 = \sum_{a,b,c,d} R(a, b) \cdot S(b, c) \cdot T(c, d)$
- $\varphi_2 = \sum_{a,b,c,d} R(a, b) \cdot S(b, c) \cdot T(b, d)$
- $\varphi_3 = \sum_{a,b,c,d} R(a, b) \cdot S(b, c) \cdot T(c, d) \cdot W(d, a)$