

Efficient Algorithm for Semiring Problems

An Intuitive Introduction to Functional Aggregate Queries

Wenchao Bai

wbai@seu.edu.cn

November 29, 2025

About This Tutorial

- **Common properties** behind optimization techniques.
- **Common structure** shared by many computational problems.
- **Unified language** able to express semiring problems.
- An intuition for **efficient algorithms** to compute any expression in this language.

Resources

- **Course materials:** Efficient Algorithms¹ (by Prof. Dan Olteanu, UZH)
- **Original paper:** FAQ: Questions Asked Frequently [1] (best paper, PODS'16)

¹<https://www.ifi.uzh.ch/en/dast/teaching/EA.html>

Outline

1 Common Properties

2 Common Structure

3 Unified Language

4 Efficient algorithms

Example: Matrix Multiplication

- **Question:** Compute $A \times B \times C$
 - **Technique:** Different parenthesizations have different cost:

$$(AB)C \quad \text{vs.} \quad A(BC)$$

- **Insight:** Using **associativity** to choose grouping reduces computation cost.

Example: MapReduce

- **Question:** Sum over a large dataset distributed across nodes
 - **Technique:** Compute partial sums parallelly at different nodes, then aggregate.

$$\sum_{i=1}^N x_i = \text{(partial sums)}$$

- **Insight:** Using **commutativity** allows arbitrary order of aggregation.

Example: Query Optimization

- **Question:** Compute the following query:

$$\sigma_{x \geq 10}(A \bowtie_x B)$$

- **Technique:** Predicate pushdown:

$$(\sigma_{x \geq 10}(A)) \bowtie_x (\sigma_{x \geq 10}(B))$$

- **Insight:** Using **distributivity** to reduce joining cost.

Takeaway: Common Properties

Associativity, **commutativity**, & **distributivity** are infrastructures for optimization.

Outline

1 Common Properties

2 Common Structure

3 Unified Language

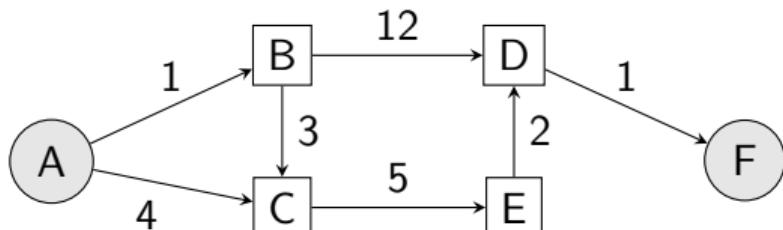
4 Efficient algorithms

Key Observation

Computational problems commonly use

- sequences of two binary operations
- applied on a finite set of values from a given domain.

Example: Shortest Distance (SD)



$$W = \begin{pmatrix} 0 & 1 & 4 & \infty & \infty & \infty \\ \infty & 0 & 3 & 12 & \infty & \infty \\ \infty & \infty & 0 & \infty & 5 & \infty \\ \infty & \infty & \infty & 0 & \infty & 1 \\ \infty & \infty & \infty & 2 & 0 & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 \end{pmatrix}$$

$$\text{SD}(A, F) = \min \left\{ \begin{array}{c} W_{A,x_1} + W_{x_1,F} \\ \dots \\ W_{A,x_1} + \dots + W_{x_{n-1},x_n} + W_{x_n,F} \end{array} \right\} = \min \left\{ \begin{array}{c} 1 + 12 + 1 \\ 4 + 5 + 2 + 1 \\ 1 + 3 + 5 + 2 + 1 \end{array} \right\}.$$

- **Binary operations:** min, +
- **Domain:** $(-\infty, \infty]$

Example: Conjunctive Query (CQ)

Orders (O for short)			Dish (D for short)		Items (I for short)	
customer	day	dish	dish	item	item	price
Elise	Monday	burger	burger	patty	patty	6
Elise	Friday	burger	burger	onion	onion	2
Steve	Friday	hotdog	burger	bun	bun	2
Joe	Friday	hotdog	hotdog	sausage	sausage	4

$$\text{CQ}(O \bowtie D \bowtie I) = \bigcup_{(v_1, v_2, v_3, v_4, v_5)} O(v_1, v_2, v_3) \cap D(v_3, v_4) \cap I(v_4, v_5)$$

- **Binary operations:** \cup, \cap
- **Domain:** set of tuples

Common Structure Shared by These Problems

Binary operators \oplus and \otimes over set \mathbf{D} form a **commutative semiring** $(\mathbf{D}, \oplus, \otimes, \mathbf{0}, \mathbf{1})$:

- \oplus is associative:
- \oplus is commutative:
- $\mathbf{0}$ is the additive identity:
- \otimes is associative:
- \otimes is commutative:
- $\mathbf{1}$ is the multiplicative identity:
- \otimes distributes over \oplus :
- $\mathbf{0}$ is the multiplicative annihilator:

$$a \oplus (b \oplus c) = (a \oplus b) \oplus c$$

$$a \oplus b = b \oplus a$$

$$a \oplus \mathbf{0} = a$$

$$a \otimes (b \otimes c) = (a \otimes b) \otimes c$$

$$a \otimes b = b \otimes a$$

$$a \otimes \mathbf{1} = a$$

$$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$$

$$a \otimes \mathbf{0} = \mathbf{0}$$

Additional condition for **ring**:

- each element a has an additive inverse $-a$:

$$a \oplus (-a) = \mathbf{0}$$

Shortest Distance (SD) as Semiring

SD forms a **min-sum semiring**: $((-\infty, \infty], \min, +, \infty, 0)$:

- $\oplus = \min$ is associative: $\min(a, \min(b, c)) = \min(\min(a, b), c)$
- $\oplus = \min$ is commutative: $\min(a, b) = \min(b, a)$
- $\mathbf{0} = \infty$ is the additive identity: $\min(a, \infty) = a$
- $\otimes = +$ is associative: $a + (b + c) = (a + b) + c$
- $\otimes = +$ is commutative: $a + b = b + a$
- $\mathbf{1} = 0$ is the multiplicative identity: $a + 0 = a$
- $\otimes = +$ distributes over $\oplus = \min$: $a + \min(b, c) = \min(a + b, a + c)$
- $\mathbf{0} = \infty$ is the multiplicative annihilator: $a + \infty = \infty$

SD cannot form a ring since,

- additive inverse does not exist: $\forall a \neq \infty, \nexists x \in (-\infty, \infty]$, such that $\min(a, x) = \infty$

Conjunctive Query (CQ) as Semiring

CQ forms a **union-intersection semiring**: $(2^U, \cup, \cap, \emptyset, U)$:

- U is the cartesian product over all attributes' domains.

- $\oplus = \cup$ is associative:

$$A \cup (B \cup C) = (A \cup B) \cup C$$

- $\oplus = \cup$ is commutative:

$$A \cup B = B \cup A$$

- $\mathbf{0} = \emptyset$ is the additive identity:

$$A \cup \emptyset = A$$

- $\otimes = \cap$ is associative:

$$A \cap (B \cap C) = (A \cap B) \cap C$$

- $\otimes = \cap$ is commutative:

$$A \cap B = B \cap A$$

- $\mathbf{1} = U$ is the multiplicative identity:

$$A \cap U = A$$

- $\otimes = \cap$ distributes over $\oplus = \cup$:

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

- $\mathbf{0} = \emptyset$ is the multiplicative annihilator:

$$A \cap \emptyset = \emptyset$$

CQ cannot form a ring since,

- additive inverse does not exist:

$$\forall A \neq \emptyset, \nexists X \in 2^U, \text{ such that } A \cup X = \emptyset$$

Sample Problems and Their Semirings²

Category	Problem	Type	Domain	\oplus	\otimes	0	1
Path Queries	Shortest Distance	Min-Sum	$(-\infty, \infty]$	min	+	∞	0
	Connectivity	Boolean	$\{F, T\}$	\vee	\wedge	F	T
	Largest Capacity	Max-Min	$[-\infty, \infty]$	max	min	$-\infty$	∞
Satisfiability	Maximum Reliability	Max-Product	$[0, 1]$	max	\times	0	1
Database Queries	Map Coloring	Boolean	$\{F, T\}$	\vee	\wedge	F	T
Database Queries	Conjunctive Queries	Union-Intersection	$2^{\mathcal{U}}$	\cup	\cap	\emptyset	\mathcal{U}
	Factorised Agg-Joins	Sum-Product	\mathbb{Z}	+	\times	0	1
...

²See topic 2 (Commutative Semirings) for more detail:
https://www_ifi_uzh_ch/en/dast/teaching/EA.html

Takeaway: The Power of Semirings

Why are Semirings Relevant in Computer Science?

- They enable generic problem solving
 - by changing the semiring, the algorithm remains the same
- They reduce computational complexity
 - thanks to the **distributivity** law
- Permutability is an important property behind optimization techniques.
 - thanks to the **associativity** and **commutativity** laws

Different semirings give different semantics of

- the same problem
- the same algorithm
- the same complexity
- the same implementation

Outline

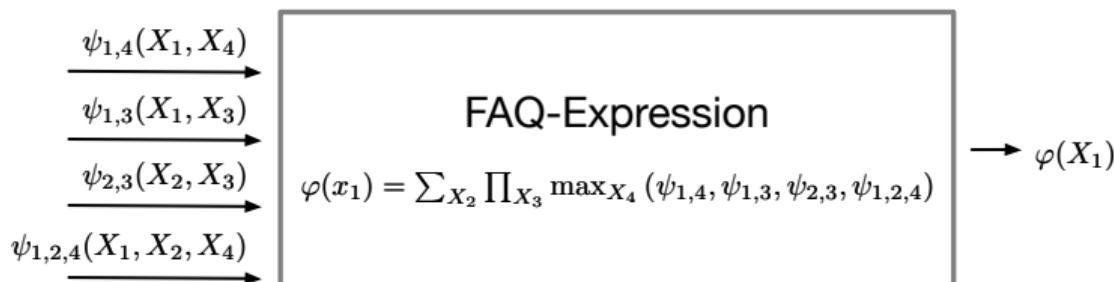
1 Common Properties

2 Common Structure

3 Unified Language

4 Efficient algorithms

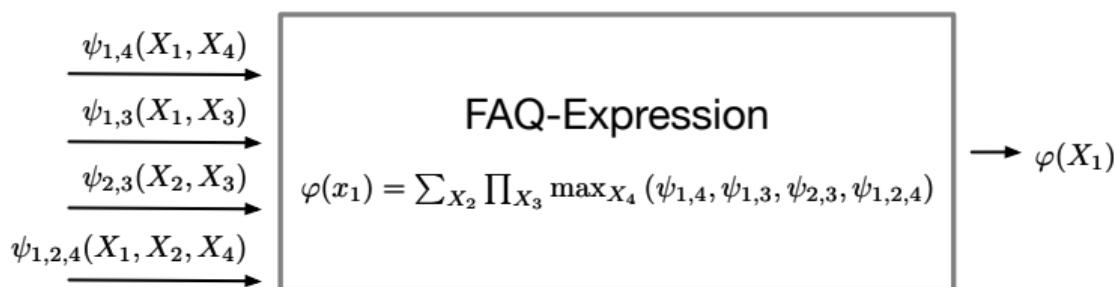
Functional Aggregate Query: The Input (1/2)



- Variables: $\mathcal{V} = \{X_1, \dots, X_n\}$
 - $F \subseteq \mathcal{V}$: free variables (input variables)³, e.g., X_1 is a free variable of $\varphi(X_1)$.
 - $\mathcal{V} \setminus F$: bound variables, e.g., $\{X_2, X_3, X_4\}$ are bound variables of $\varphi(X_1)$.
 - E.g., in the query $SD(A, B)$ “the shortest dist. between A and B ”, $F = \{A, B\}$.

³w.l.o.g., $F = \mathbf{X}_{[f]} = \{X_1, \dots, X_f\}$, i.e., the first f variables.

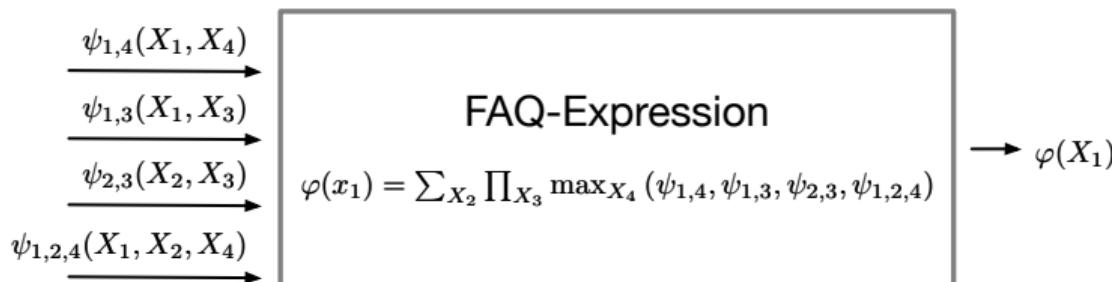
Functional Aggregate Query: The Input (2/2)



- Variables: $\mathcal{V} = \{X_1, \dots, X_n\}$
- Multi-Hypergraph: $\mathcal{H} = (\mathcal{V}, \mathcal{E})$
 - \mathcal{V} : set of vertices (variables)
 - $\mathcal{E} \subseteq 2^{[n]}$: $\forall S \in \mathcal{E}$, we have a factor ψ_S . All factors have the same range \mathbf{D} .

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

Functional Aggregate Query: The Output

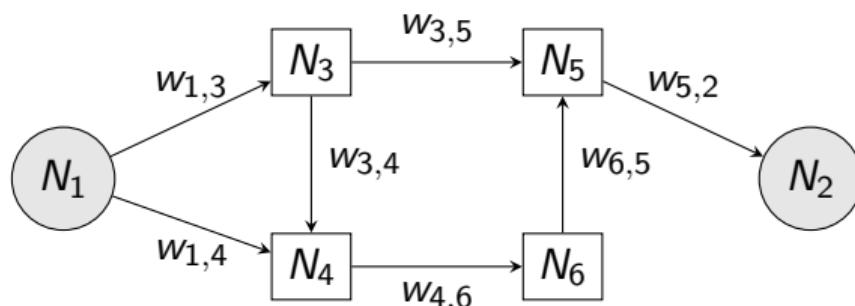


- Compute the function $\varphi : \prod_{i \in F} \text{Dom}(X_i) \rightarrow \mathbf{D}$.
- φ is defined by the **FAQ-Expression**:

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

- For each $\oplus^{(i)}$, either $(\mathbf{D}, \oplus^{(i)}, \otimes, \mathbf{0}, \mathbf{1})$ is a commutative semiring, or $\oplus^{(i)} = \otimes$.

Path Query as FAQ (1/5)



- Variables \mathcal{V} : $\{X_1, \dots, X_6\}$, where $\forall X_i \in \mathcal{V}, \text{Dom}(X_i) = V(G) = \{N_1, \dots, N_6\}$.
- Free variables F : $\{X_1, X_2\}$ are assigned as the source and target vertices.
- Hyperedges \mathcal{E} : vertex pair $E(G) = V(G)^2$.
- Factors ψ_S : function $\mathcal{E} \rightarrow \mathbf{D}$, where $S \in \mathcal{E}$.

Path Query as FAQ (2/5)

$$\begin{aligned}\varphi(\mathbf{x}_{[2]}) &= \bigoplus_{x_3, x_4, x_5, x_6 \in V(G)} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S) \\ &= \psi(N_1, N_2) \oplus \left(\bigoplus_{x_3 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, N_2) \right) && // 1 \text{ and } 2 \text{ hops} \\ &= \dots && // 3 \text{ and } 4 \text{ hops} \\ &= \oplus \left(\bigoplus_{x_3, \dots, x_6 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, x_4) \otimes \dots \otimes \psi(x_6, N_2) \right) && // 5 \text{ hops}\end{aligned}$$

- **Shortest distance:** $\oplus = \min$, $\otimes = +$, ψ returns edge weights, $\mathbf{D} = \mathbb{R} \cup \{\infty\}$

Path Query as FAQ (3/5)

$$\begin{aligned}\varphi(\mathbf{x}_{[2]}) &= \bigoplus_{x_3, x_4, x_5, x_6 \in V(G)} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S) \\ &= \psi(N_1, N_2) \oplus \left(\bigoplus_{x_3 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, N_2) \right) && // 1 \text{ and } 2 \text{ hops} \\ &= \dots && // 3 \text{ and } 4 \text{ hops} \\ &= \oplus \left(\bigoplus_{x_3, \dots, x_6 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, x_4) \otimes \dots \otimes \psi(x_6, N_2) \right) && // 5 \text{ hops}\end{aligned}$$

- **Largest capacity:** $\oplus = \max$, $\otimes = \min$, ψ returns edge weights, $\mathbf{D} = \mathbb{R} \cup \{-\infty, \infty\}$

Path Query as FAQ (4/5)

$$\begin{aligned}\varphi(\mathbf{x}_{[2]}) &= \bigoplus_{x_3, x_4, x_5, x_6 \in V(G)} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S) \\ &= \psi(N_1, N_2) \oplus \left(\bigoplus_{x_3 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, N_2) \right) && // 1 \text{ and } 2 \text{ hops} \\ &= \dots && // 3 \text{ and } 4 \text{ hops} \\ &= \oplus \left(\bigoplus_{x_3, \dots, x_6 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, x_4) \otimes \dots \otimes \psi(x_6, N_2) \right) && // 5 \text{ hops}\end{aligned}$$

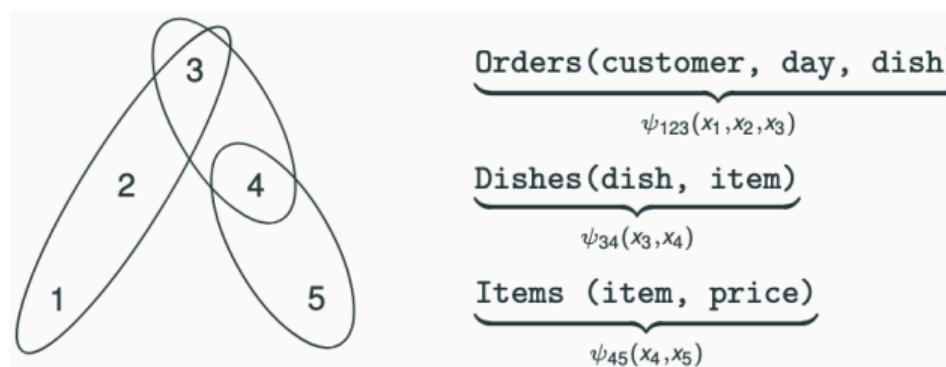
- **Connectivity:** $\oplus = \vee$, $\otimes = \wedge$, ψ returns edge existance, $\mathbf{D} = \{\mathbf{F}, \mathbf{T}\}$

Path Query as FAQ (5/5)

$$\begin{aligned}\varphi(\mathbf{x}_{[2]}) &= \bigoplus_{x_3, x_4, x_5, x_6 \in V(G)} \bigotimes_{S \in \mathcal{E}} \psi_S(\mathbf{x}_S) \\ &= \psi(N_1, N_2) \oplus \left(\bigoplus_{x_3 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, N_2) \right) && // 1 \text{ and } 2 \text{ hops} \\ &= \dots && // 3 \text{ and } 4 \text{ hops} \\ &= \oplus \left(\bigoplus_{x_3, \dots, x_6 \in V(G)} \psi(N_1, x_3) \otimes \psi(x_3, x_4) \otimes \dots \otimes \psi(x_6, N_2) \right) && // 5 \text{ hops}\end{aligned}$$

- **Shortest path:** $\oplus = \cup$, $\otimes = \text{concat}$, ψ returns edge itself or \emptyset , $\mathbf{D} = E(G) \cup \{\emptyset\}$

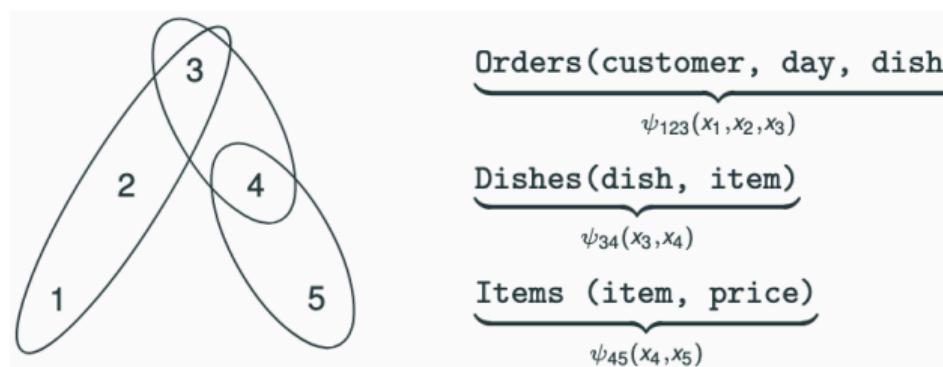
DB Query as FAQ (1/3)



- Q1: SELECT * FROM Orders NATURAL JOIN Dish NATURAL JOIN Items;
- **FAQ** over **union-intersection** semiring, where ψ maps tuple to $\{\emptyset, \{\text{tuple}\}\}$:

$$\varphi() = \bigcup_{x_1, x_2, x_3, x_4, x_5} \psi_{1,2,3}(x_1, x_2, x_3) \cap \psi_{3,4}(x_3, x_4) \cap \psi_{4,5}(x_4, x_5)$$

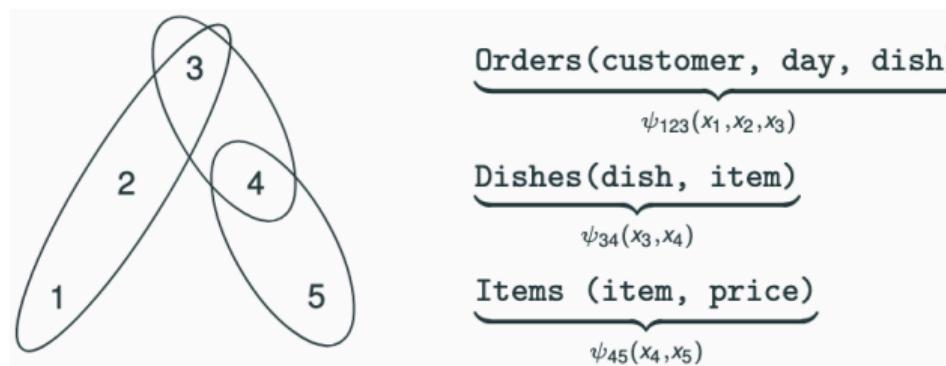
DB Query as FAQ (2/3)



- Q2: `SELECT customer,COUNT(*) from Q1 GROUP BY customer;`
- **FAQ** over **sum-product** semiring, where ψ maps tuple to $\{0, 1\}$:

$$\varphi(x_1) = \sum_{x_2, x_3, x_4, x_5} \psi_{1,2,3}(x_1, x_2, x_3) \cdot \psi_{3,4}(x_3, x_4) \cdot \psi_{4,5}(x_4, x_5)$$

DB Query as FAQ (3/3)



- Q3: `SELECT customer,day,SUM(price) from Q1 GROUP BY customer,day;`
- **FAQ** over **sum-product** semiring, where $\psi_{4,5}$ maps (x_4, x_5) to x_5 ; others are the same as Q2:

$$\varphi(x_1, x_2) = \sum_{x_3, x_4, x_5} \psi_{1,2,3}(x_1, x_2, x_3) \cdot \psi_{3,4}(x_3, x_4) \cdot \psi_{4,5}(x_4, x_5)$$

Takeaway: A Unified Language

- FAQ is a **unified language** to express many problems in computer science.
- See appendix for more problems expressible in FAQ over different semirings.

Outline

1 Common Properties

2 Common Structure

3 Unified Language

4 Efficient algorithms

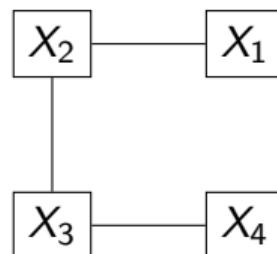
The Nature of FAQ

- A collection of **factors**.
- A **hypergraph** to guide the factor assembling.

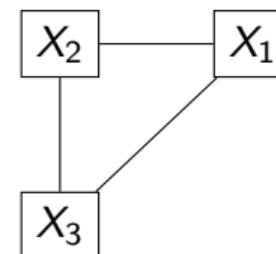
Hypergraphs: The Good and The Bad

Consider the following two FAQs. φ_2 is the same as φ_2 in the case when $X_4 = X_1$.

- **Acyclic FAQ:** $\varphi_1 = \bigoplus x_{[4]} \psi_{1,2}(x_1, x_2) \otimes \psi_{2,3}(x_2, x_3) \otimes \psi_{3,4}(x_3, x_4)$
- **Cyclic FAQ:** $\varphi_2 = \bigoplus x_{[3]} \psi_{1,2}(x_1, x_2) \otimes \psi_{2,3}(x_2, x_3) \otimes \psi_{3,1}(x_1, x_3)$

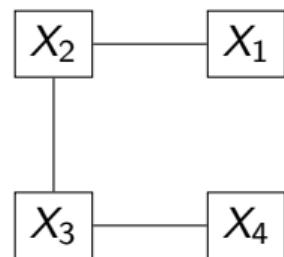


Hypergraph of φ_1 .



Hypergraph of φ_2 .

Example: The Acyclic FAQ over Boolean Semiring (1/2)

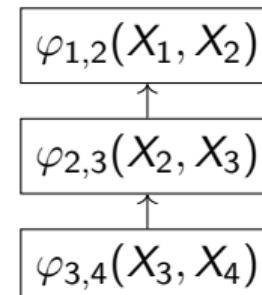
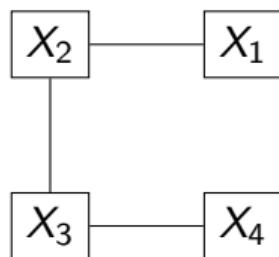


Consider the instance of φ_1 over the Boolean semiring:

$$\varphi = \bigvee_{\mathbf{x}_{[4]} \in \prod_{i \in [4]} \text{Dom}(X_i)} \psi_{1,2}(x_1, x_2) \wedge \psi_{2,3}(x_2, x_3) \wedge \psi_{3,4}(x_3, x_4)$$

φ asks whether there's a tuple (x_1, \dots, x_4) such that all factors $\psi_{i,j}(x_i, x_j) = \text{True}$.

Example: The Acyclic FAQ over Boolean Semiring (2/2)



$$\varphi = \bigvee_{\mathbf{x}_{[4]} \in \prod_{i \in [4]} \text{Dom}(X_i)} \psi_{1,2}(x_1, x_2) \wedge \psi_{2,3}(x_2, x_3) \wedge \psi_{3,4}(x_3, x_4)$$

@ $\varphi_{3,4}$ Send up x_4 -values: $V_{(3,4) \rightarrow (2,3)}(x_4) = \bigvee_{x_3} \psi_{3,4}(x_3, x_4)$

@ $\varphi_{2,3}$ Send up x_2 -values: $V_{(2,3) \rightarrow (1,2)}(x_2) = \bigvee_{x_3} \psi_{2,3}(x_2, x_3) \wedge V_{(3,4) \rightarrow (2,3)}(x_3)$

@ $\varphi_{1,2}$ Sum up: $\varphi() = \bigvee_{x_1} \psi_{1,2}(x_1, x_2) \wedge V_{(2,3) \rightarrow (1,2)}(x_2)$

The Power of Acyclicity

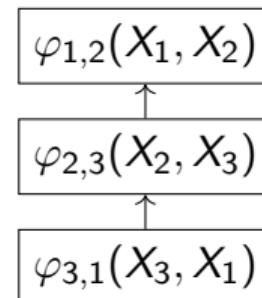
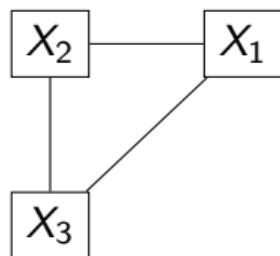
All computation steps are local and their cost upper bounded by the factor sizes

- Typical assumption: $|\psi_{i,j}| \leq N$ for some value N .
- We pass along at most N values between factors.
- Local computation is just filtering local values with incoming values.
- Overall: linear computation time - This is the best in the worst case.

Evaluation strategy know for decades under different names:

- Message passing [4] (in AI literatures)
- Semi-Join reduction [5] (in DB literatures)

The Bad Case: Cyclic FAQs (1/2)



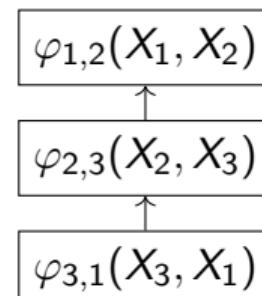
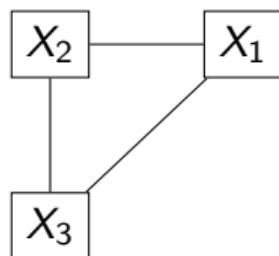
$$\varphi' = \bigvee_{\mathbf{x}_{[3]} \in \prod_{i \in [3]} \text{Dom}(X_i)} \psi_{1,2}(x_1, x_2) \wedge \psi_{2,3}(x_2, x_3) \wedge \psi_{3,1}(x_3, x_1)$$

@ $\varphi_{3,1}$ Send up (x_1, x_3) -values: $V_{(3,1) \rightarrow (2,3)}(x_1, x_3) = \psi_{3,1}(x_3, x_1)$

@ $\varphi_{2,3}$ Send up (x_1, x_2) -values: $V_{(2,3) \rightarrow (1,2)}(x_1, x_2) = \bigvee_{x_3} \psi_{2,3}(x_2, x_3) \wedge V_{(3,1) \rightarrow (2,3)}(x_1, x_3)$

@ $\varphi_{1,2}$ Sum up: $\varphi'() = \bigvee_{x_1, x_2} \psi_{1,2}(x_1, x_2) \wedge V_{(2,3) \rightarrow (1,2)}(x_1, x_2)$

The Bad Case: Cyclic FAQs (2/2)



$$\varphi' = \bigvee_{\mathbf{x}_{[3]} \in \prod_{i \in [3]} \text{Dom}(X_i)} \psi_{1,2}(x_1, x_2) \wedge \psi_{2,3}(x_2, x_3) \wedge \psi_{3,1}(x_3, x_1)$$

$V_{(2,3) \rightarrow (1,2)} = \bigvee_{x_3} \psi_{2,3}(x_2, x_3) \wedge V_{(3,1) \rightarrow (2,3)}(x_1, x_3)$ introduces $O(N^2)$ cost.

- It's a join instead of a semi-join.

A Roadmap for Further Study

- 1 Can we distinguish syntactically the acyclic from the cyclic hypergraphs?
 - α -acyclic, β -acyclic, free-connex, etc.
- 2 How to transform cyclic hypergraphs to acyclic ones?
 - Hypertree decomposition.
- 3 How to measure the goodness of such transformations?
 - Width measures: hypertree width, fractional hypertree width, etc.
- 4 How to design efficient algorithms for FAQs over (commutative) semirings?
 - InsideOut algorithm [1].

Problems Expressible in FAQ over Boolean Semiring

$(\{F, T\}, \vee, \wedge, F, T)$

- Constraint satisfaction problems (CSP) FAQ [1]
- Boolean conjunctive query evaluation (BCQ) FAQ [1]
- Conjunctive query evaluation (CQ)⁴ FAQ [1]
- Join evaluation FAQ [1]
- Satisfiability (SAT) FAQ [1]
- k -colorability FAQ [1]
- List recovery problem (coding theory) FAQ [1]

⁴it's also expressible using the set semiring.

Problems Expressible in FAQ over Set Sum-Product Semiring

$$(2^{\mathcal{U}}, \cup, \cap, \emptyset, \mathcal{U})$$

- Conjunctive query evaluation (CQ)⁵ FAQ [1]
- Join evaluation FAQ [1]

⁵It's also expressible using the Boolean semiring.

Problems Expressible in FAQ over Natural Sum-Product Semiring

$(\mathbb{N}, +, \times, 0, 1)$

- Complex network analysis FAQ [1]
- Count constraint satisfaction problems (#CSP) FAQ [1]
- Count satisfiability (#SAT) FAQ [1]

Problems Expressible in FAQ over Real Sum-Product Semiring

$(\mathbb{R}, +, \times, 0, 1)$

- Permanent FAQ [1]
- Discrete Fourier transform FAQ [1], AjiMcEl [2]
- Hadamard transform AjiMcEl [2]
- Inference in probabilistic graphical models FAQ [1]
- Probability propagation in AI AjiMcEl [2]
- Matrix chain multiplication FAQ [1], AjiMcEl [2]
- Graph homomorphism FAQ [1]
- BCJR decoding (Bahl, Cocke, Jelinek, Raviv) AjiMcEl [2]
- Holant problem FAQ [1]

Problems Expressible in FAQ over Max-Product Semiring

$([0, \infty), \max, \times, 0, 1)$

- MAP queries in probabilistic graphical models FAQ [1]
- Quantified conjunctive query evaluation (QCQ)⁶ FAQ [1]

⁶It's also expressible using the max-product, min-product semirings.

Problems Expressible in FAQ over Min-Sum Semiring

$((-\infty, \infty], \min, +, \infty, 0)$

- Gallager-Tanner-Wiberg decoding AjiMcEl [2]
- Viterbi decoding AjiMcEl [2]
- Trellis path problem AjiMcEl [2]
- Graph optimization KohlWils [3]
- Queuing systems KohlWils [3]
- Discrete event systems KohlWils [3]
- Optimization for weighted CSPs KohlWils [3]

Problems Expressible in FAQ over Two Semiring

$([0, \infty), \max, \times, 0, 1), ((0, \infty], \min, \times, \infty, 1)$

- Quantified conjunctive query evaluation (QCQ)⁷ FAQ [1]

$(\mathbb{N}, \max, \times, 0, 1), (\mathbb{N}, +, \times, 0, 1)$

- Count conjunctive query evaluation (#CQ) FAQ [1]
- Count quantified conjunctive query evaluation (#QCQ) FAQ [1]

⁷It's also expressible using the max-product semiring

References I

-  Mahmoud Abo Khamis, Hung Q. Ngo, and Atri Rudra.
Faq: Questions asked frequently.
In Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS '16, page 13–28, 2016.
-  S.M. Aji and R.J. McEliece.
The generalized distributive law.
IEEE Transactions on Information Theory, 46(2):325–343, 2000.
-  Jürg Kohlas and Nic Wilson.
Semiring induced valuation algebras: Exact and approximate local computation algorithms.
Artif. Intell., 172(11):1360–1399, 2008.

References II

-  Judea Pearl.
Fusion, propagation, and structuring in belief networks.
In Probabilistic and Causal Inference: The Works of Judea Pearl, pages 139–188.
2022.
-  Mihalis Yannakakis.
Algorithms for acyclic database schemes.
In VLDB, volume 81, pages 82–94, 1981.