On Functional Aggregate Queries with Additive Inequalities

M. Abo Khamis, R. R. Curtin, B. Moseley, H. Q. Ngo, X. Nguyen, D. Olteanu, M. Schleich

erc · University of Oxford · University of Michigan · Carnegie Mellon University · relationalAI (AI for the enterprise)

relational.ai fdbresearch.github.io

FAQ-AI: Functional Aggregate Queries with Additive Inequalities

$$\boxed{Q(\mathbf{x}_F) = \bigoplus_{\mathbf{x}_{\mathcal{V}\setminus F}} \left(\bigotimes_{K\in\mathcal{E}_s} R_K(\mathbf{x}_K) \right) \otimes \left(\bigotimes_{K\in\mathcal{E}_\ell} \mathbf{1}_{\sum_{v\in K}\theta_v^K(x_v)\le 0} \right)}$$

$$Q(a,b) = \sum_c R(a,b)\cdot S(b,c)\cdot T(c,d)\cdot \mathbf{1}_{a\le d}\cdot \mathbf{1}_{\frac{b}{2}\le c}\cdot \mathbf{1}_{a^2+\frac{b}{2}\le 5c}$$ Sum-Product: $(\mathbb{R},+,\times)$

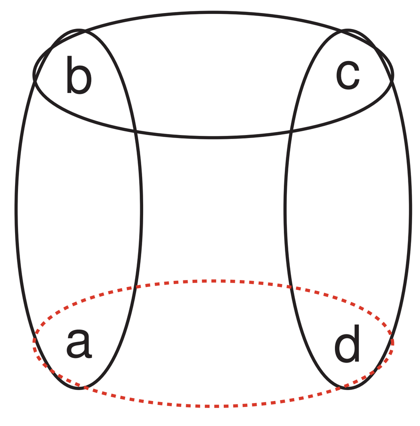
$$Q(a,b) = \bigvee_c R(a,b)\wedge S(b,c)\wedge T(c,d)\wedge \mathbf{1}_{a\le d}\wedge \mathbf{1}_{\frac{b}{2}\le c}\wedge \mathbf{1}_{a^2+\frac{b}{2}\le 5c}$$ Boolean: $(\{1,0\},\vee,\wedge)$

$$Q(a,b) = \bigotimes_c R(a,b)\oplus S(b,c)\oplus T(c,d)\oplus \mathbf{1}_{a\le d}\oplus \mathbf{1}_{\frac{b}{2}\le c}\oplus \mathbf{1}_{a^2+\frac{b}{2}\le 5c}$$ Arbitrary: $(\mathbf{D},\oplus,\otimes)$

Query Hypergraph for FAQ-AIs

Query Hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E} = \mathcal{E}_s \cup \mathcal{E}_\ell)$

- Set of variables $\mathcal{V} = \{X_1,\dots,X_n\}$
- Set of "skeleton" hyperedges $\mathcal{E}_s$
  - Each hyperedge in $\mathcal{E}_s$ is defined by a factor $R_K(\mathbf{x}_K)$
- Set of "ligament" hyperedges $\mathcal{E}_\ell$
  - Each hyperedge in $\mathcal{E}_\ell$ is defined by sum of univariate functions



$\mathcal{H}$ for Example II

Relaxed Tree Decompositions

A Generalized Hypertree Decomposition (TD) for $\mathcal{H} = (\mathcal{V}, \mathcal{E} = \mathcal{E}_s \cup \mathcal{E}_\ell)$:

1. Tree $T = (V(T), E(T))$
2. Bag $\chi(t) \subseteq \mathcal{V}$ for each tree-node $t \in V(T)$

A TD satisfies:

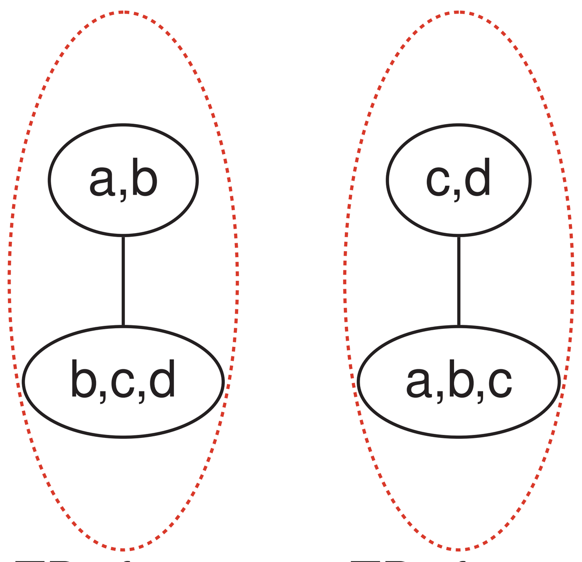
1. Running intersection property
2. Containment property

Containment for Tree Decompositions:

1. every hyperedge is covered by some bag

Containment for Relaxed Tree Decompositions:

1. every skeleton hyperedge is covered by some bag
2. every ligament hyperedge is covered by two adjacent bags



TD_1 for $\mathcal{H}$ TD_2 for $\mathcal{H}$

Example I

Given: Relations R, S of size $O(N)$

Task: $Q() = \sum_{a,b,c} R(a,b)\cdot S(b,c)\cdot \mathbf{1}_{a\le c}$

Existing approaches take $O(N^2)$ time

Our approach takes $O(N\log N)$ time

R a	b
1	1
1	2
2	1
2	2
3	2
3	3

lookup $b = 2$

count $= 3$

S b	c	#
1	1	2
1	2	1
2	0	4
2	2	3
2	3	2
2	4	1

find smallest $c \ge a$

Step 1: Pre-count S for each b
Step 2: Foreach $R(a,b)$, locate first $S(b,c)$ with $a \le c$
Step 3: Return pre-aggregated count

Example II

Given: Relations R, S, T of size $O(N)$

Task: $Q() = \sum_{a,b,c,d} R(a,b)\cdot S(b,c)\cdot T(c,d)\cdot \mathbf{1}_{a\le d}$

Existing approaches take $O(N^2)$ time

Our approach takes $O(N^{1.5}\log N)$ time

$S^{\text{light}} = \{t\in S : |\sigma_{b=t.b}S| \le \sqrt{N}\}$ $S^{\text{heavy}} = S\setminus S^{\text{light}}$ Partition S

$$\sum_{a,b,c,d} R\cdot(S^{\text{light}} + S^{\text{heavy}})\cdot T\cdot \mathbf{1}_{a\le d}$$

Pull out +

$$\sum_{a,b,c,d} \underbrace{R\cdot S^{\text{light}}}_{U(a,b,c)}\cdot T\cdot \mathbf{1}_{a\le d} + \sum_{a,b,c,d} R\cdot \underbrace{S^{\text{heavy}}\cdot T}_{W(b,c,d)}\cdot \mathbf{1}_{a\le d}$$

$O(N^{1.5})$ $O(N^{1.5})$ Compute U,W

$$\sum_{a,b,c,d} U\cdot T\cdot \mathbf{1}_{a\le d} + \sum_{a,b,c,d} R\cdot W\cdot \mathbf{1}_{a\le d}$$

Solve as in Example I

Example III: Learning SVM over Databases

Task: Compute $J(\beta)$ over dataset **D** defined by query Q over database I

$$J(\beta) = \sum_{(\mathbf{x},y)\in\mathbf{D}} \underbrace{\max\{0, 1-y\cdot f_\beta(\mathbf{x})\}}_{\text{Hinge Loss}}$$
$$= \underbrace{\sum_{(\mathbf{x},y)\in\mathbf{D}} (1-y\cdot f_\beta(\mathbf{x}))\cdot \mathbf{1}_{y\cdot f_\beta(\mathbf{x})\le 1}}_{\text{FAQ-AI}}$$

Existing approaches:

1. materialize **D**
2. learn model using favorite ML tool

Our approach:

1. avoid materialization of **D**
2. express learning using FAQ-AIs

SVM models can be learned in time **sublinear** in $|\mathbf{D}|$.

Further ML models that can benefit from FAQ-AI:

- k-Means Clustering
- Robust Regression with Huber Loss
- Boolean Principle Component Analysis
- Other models trained with non-polynomial loss

Width Measures for FAQs and FAQ-AIs without Free Variables

unbounded gap → ; polynomial gap ↓

	Single TD		Disjoint Partitioning Multiple TDs		General Partitioning Multiple TDs
FAQ	fhtw (InsideOut)	$\ge$	#subw (#PANDA)	$\ge$	subw (PANDA)
	VI		VI		VI
FAQ-AI	$\text{fhtw}_{\text{relaxed}}$ (InsideOut+GDS)	$\ge$	$\#\text{subw}_{\text{relaxed}}$ (#PANDA+GDS)	$\ge$	$\text{subw}_{\text{relaxed}}$ (PANDA+GDS)
	Arbitrary semiring				Boolean semiring

fhtw= fractional hypertree width
subw= submodular width
GDS = Chazelle's geometric data structure
■ = new result

Width Measures for FAQs and FAQ-AIs with Free Variables

unbounded gap → ; polynomial gap ↓

	Single TD		Disjoint Partitioning Multiple TDs		General Partitioning Multiple TDs
FAQ	faqw (InsideOut)	$\ge$	#smfw (#PANDA)	$\ge$	smfw (PANDA)
	VI		VI		VI
FAQ-AI	$\text{faqw}_{\text{relaxed}}$ (InsideOut+GDS)	$\ge$	$\#\text{smfw}_{\text{relaxed}}$ (#PANDA+GDS)	$\ge$	$\text{smfw}_{\text{relaxed}}$ (PANDA+GDS)
	Arbitrary semiring				Boolean semiring

faqw= FAQ width
smfw= submodular FAQ width
GDS = Chazelle's geometric data structure
■ = new result