Learning Regression Models over Factorized Views

Maximilian Schleich, Dan Olteanu, Radu Ciucanu
Department of Computer Science, University of Oxford

DEPARTMENT OF COMPUTER SCIENCE · UNIVERSITY OF OXFORD · erc

Idea

- **Goal**: Build *least squares regression models* over training datasets defined by arbitrary join queries over databases.
- **Observation**: Joins entail a *high degree of redundancy* in both computation and data representation, which is not required for an end-to-end solution to learning over joins.
- **Solution**: **F** uses gradient descent to learn the model parameters in one pass over a factorized database view.

Recap on Linear Regression

The model:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \ldots \theta_n x_n = \sum_{i=0}^{n} \theta_i x_i, \text{ where } x_0 = 1$$

Least Squares Objective Function gives a measure of the error between the actual value $y^{(i)}$ and the model $h_\theta(x^{(i)})$:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2.$$

To minimise the error, we calculate the gradient and update the parameters iteratively as follows:

$$\forall 0 \le j \le n : \theta_j := \theta_j - \alpha \frac{\delta}{\delta \theta_j} J(\theta)$$
$$:= \theta_j - \alpha \sum_{i=1}^{m}\left(\sum_{k=0}^{n} \theta_k x_k^{(i)} - y^{(i)}\right) x_j^{(i)}.$$

By letting $\theta_y = -1$, the gradient becomes:

$$\sum_{i=1}^{m}\left(\sum_{k=0}^{n} \theta_k x_k^{(i)}\right) x_j^{(i)}$$

Highlights of Our Solution

- Decouple data-dependent (aggregate) computation from data-independent parameter convergence:

$$\forall 0 \le j \le n : S_j = \sum_{k=0}^{n} \theta_k \times \text{Cofactor}[k,j]$$
$$\text{where Cofactor}[k,j] = \sum_{i=1}^{m} x_k^{(i)} x_j^{(i)}$$

- Compute the cofactor matrix in one pass over the factorized join.
- Time and space complexity: $O(|D|^{fhtw(Q)})$, where $fhtw(Q)$ is the fractional hypertree width of the query hypergraph.
- Principles also applicable to polynomial regression, factorized machines, and various regularizers.

Factorized Join Example

- Natural join of relations House, Shop, TaxBand.
- Redundancy in flat join: z_1 occurs in 24 tuples, h_1 to h_3 occur in eight tuples each and are paired with the same combination of values for P, T and S.
- Avoid to explicitly materialise the local products.

Shops

Z	H
z_1	h_1
z_1	h_2
z_1	h_3
z_2	h_4

House

Z	S	P
z_1	s_1	p_1
z_1	s_1	p_2
z_1	s_2	p_3
z_2	s_2	p_4

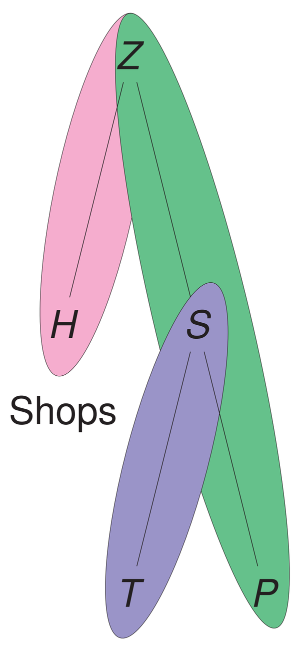
TaxBand

S	T
s_1	t_1
s_1	t_2
s_1	t_3
s_2	t_4
s_2	t_5

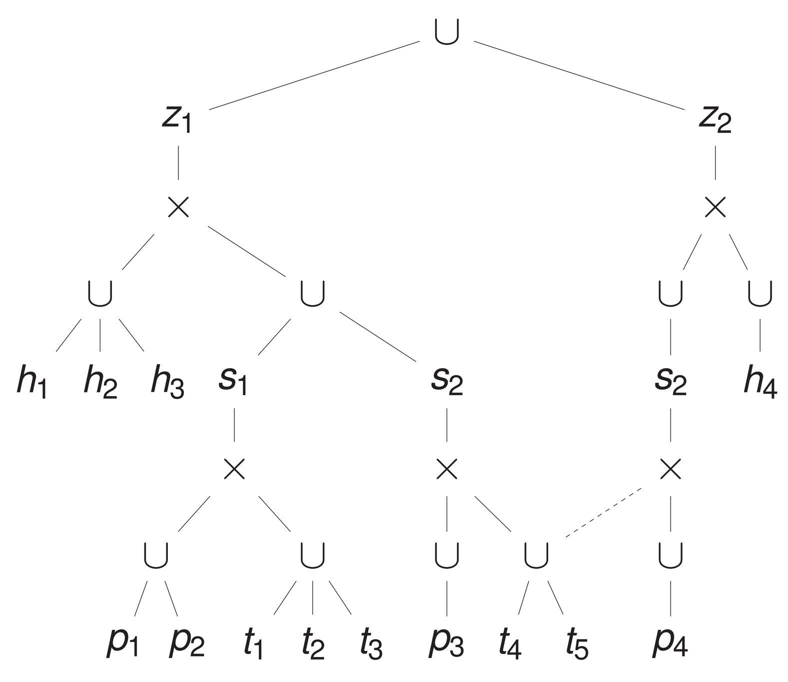
Shops ⋈ House ⋈ TaxBand

Z	H	S	P	T
z_1	h_1	s_1	p_1	t_1
z_1	h_1	s_1	p_1	t_2
z_1	h_1	s_1	p_1	t_3
z_1	h_1	s_1	p_2	t_1
z_1	h_1	s_1	p_2	t_2
z_1	h_1	s_1	p_2	t_3
z_1	h_1	s_2	p_3	t_4
z_1	h_1	s_2	p_3	t_5
....				
the above for h_2 and h_3				
....				
z_2	h_4	s_2	p_4	t_4
z_2	h_4	s_2	p_4	t_5

(a) The three relations of database **D** and natural join $Q(\mathbf{D})$.



(b) F-tree F.

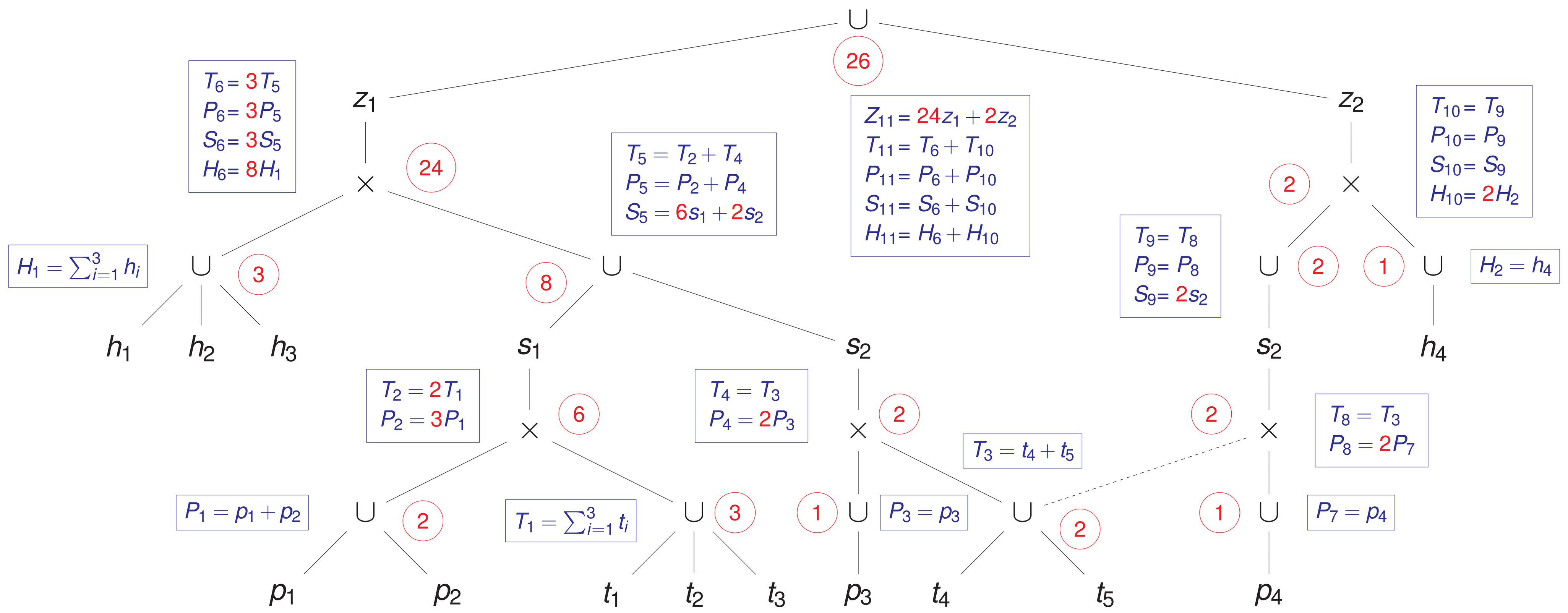


(c) Factorization $F(\mathbf{D})$ of $Q(\mathbf{D})$ over F.

F Computes the Cofactors of Model Parameters in One Pass over the Factorized Join

Reformulate the sum-aggregates defining cofactors to save computation. **This rewriting is already performed by the factorization of the join!**

1. Compute occurrence count of each value and weighted sums of each attribute at each node.
2. Complete incrementally the cofactor matrix.



	θ_0	θ_Z	θ_S	θ_P	θ_T	θ_H
Σ_0	26	$24z_1+2z_2$	$3(6s_1+2s_2)+2s_2$	$9(p_1+p_2)+6p_3+2p_4$	$6(t_1+t_2+t_3)+3(t_4+t_5)$	$8(h_1+h_2+h_3)+2h_4$
Σ_Z	Σ_0/θ_Z	$24z_1^2+2z_2^2$	$z_1S_6+z_2S_{10}$	$z_1P_6+z_2P_{10}$	$z_1T_6+z_2T_{10}$	$z_1H_6+z_2H_{10}$
Σ_S	Σ_0/θ_S	Σ_Z/θ_S	$3(6s_1^2+2s_2^2)+2s_2^2$	$3(s_1P_2+s_2P_4)+s_2P_8$	$3s_1T_2+3s_2T_4+s_2T_8$	$S_5H_1+S_9H_2$
Σ_P	Σ_0/θ_P	Σ_Z/θ_P	Σ_S/θ_P	$9(p_1^2+p_2^2)+6p_3^2+2p_4^2$	$3P_1T_1+3P_3T_3+P_7T_3$	$P_5H_1+P_9H_2$
Σ_T	Σ_0/θ_T	Σ_Z/θ_T	Σ_S/θ_T	Σ_P/θ_T	$6(t_1^2+t_2^2+t_3^2)+3(t_4^2+t_5^2)$	$T_5H_1+T_9H_2$
Σ_H	Σ_0/θ_H	Σ_Z/θ_H	Σ_S/θ_H	Σ_P/θ_H	Σ_T/θ_H	$8(h_1^2+h_2^2+h_3^2)+2h_4^2$

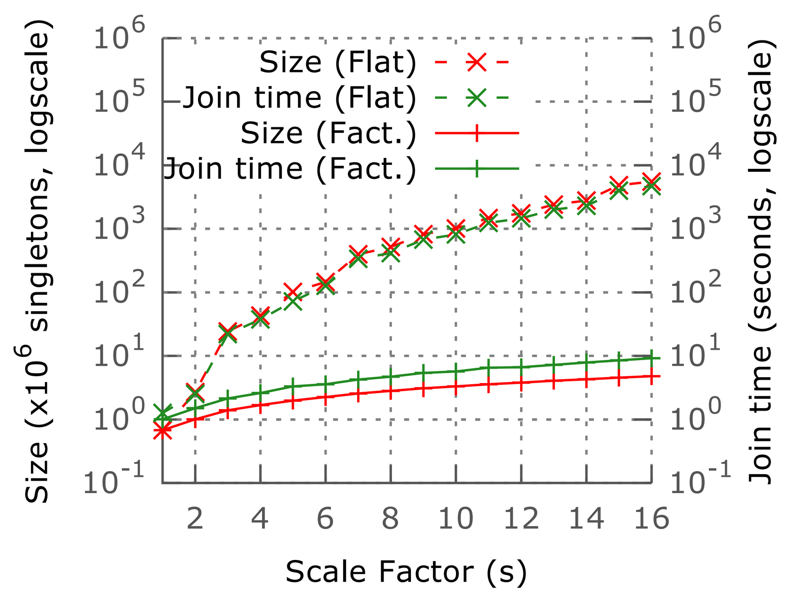
F: Flavors and Competitors

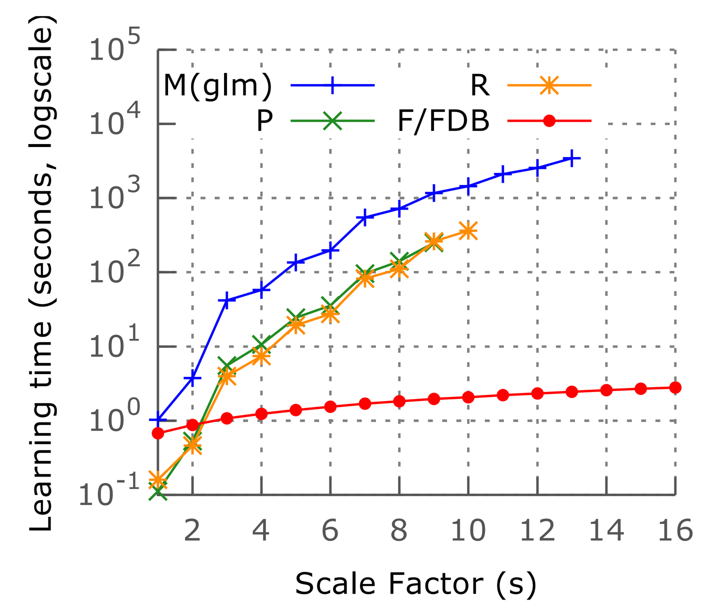
- **F/FDB**: Cofactors computed in one pass over the materialized factorized join.
- **F**: Factorized join and cofactor computation intermixed.
- **F/SQL**: SQL-encoding of **F**, intertwining joins and cofactors in one query.

Competitors: R (QR decomp.), Python StatsModels (ols), MADlib (ols, glm).

Experiments on Synthetic Data

- (left) Factorized join: Compression ratio and speedup.
- (right) Learning: Speedup.





Experiments on Real Data

		US retailer	LastFM (1)	LastFM (2)	MovieLens
# parameters		31	6	10	27
Join Size	Factorized	97,134,675	376,402	315,818	2,115,610
	Flat	2,585,046,352	369,986,292	590,793,800	27,005,643
	Compression	26.61×	982.86×	1870.68×	12.76×
Join Time	Factorized	36.03	4.79	9.94	12.28
	Flat	249.41	54.25	61.33	1.30
Import	**F** and M	0	0	0	0
	R	1189.12*	155.91	276.77	10.86
	P	1164.40*	179.16	328.97	11.33
Learn Time	**F**	9.69	0.53	0.89	3.87
	M (glm)	2671.88	572.88	746.50	31.91
	R	810.66*	268.04	466.52	6.96
	P	1199.50*	35.74	148.84	10.92
Total Time	**F**	16.29	0.11	0.25	2.12
	F/FDB	45.72	5.32	10.83	16.15
	F/SQL	108.81	0.58	2.00	14.26
	M (ols)	680.60	152.37	196.60	7.08
	M (glm)	2921.29	627.13	807.83	33.21
	R	2249.19*	478.20	804.62	19.12
	P	2613.31*	269.15	539.14	23.55
Speedup	**F** vs. M (ols)	41.78×	1385.18×	786.40×	3.34×
	F vs. M (glm)	179.33×	5701.18×	3231.32×	15.67×
	F vs. R	138.07×	4347.27×	3218.48×	9.02×
	F vs. P	57.16×	50.59×	49.78×	1.46×

Learning Linear Regression Models over Factorized Joins (SIGMOD'16); F: Regression Models over Factorized Views (PVLDB'16); Factorized Databases (SIGMOD Record'16)

WWW: http://www.cs.ox.ac.uk/projects/FDB/