The Gamma Operator for Big Data Summarization on an Array DBMS

Yiqun Zhang *University of Houston (USA)*

Summary

SciDB is a parallel array DBMS that provides multidimensional arrays, a query language and basic ACID properties. We introduce a summarization matrix operator that computes sufficient statistics. This operator:

- requires only one pass over the input data set.
- doesn't require the data set to fit in RAM.
- exhibits linear time complexity and speedup.
- can work in parallel.
- can benefit a family of statistical and machine learning models. (e.g., PCA, linear regression and variable selection.)
- works an order of magnitude faster than SciDB built-in operators (SciDB calls LAPACK).
- works two orders of magnitude faster than SQL queries on a fast column DBMS.
- works faster than the R package even when the data set fits in RAM.

We also show PCA and linear regression computation is reduced to a few minutes for large data sets. On the other hand, a Gibbs sampler for variable selection can iterate much faster in the array DBMS than in R, exploiting the summarization matrix.

Definitions

Matrix	Size	Description
X	$d \times n$	Data set: independent variables, dimensions
U	$d \times d$	Principal components
V	$d \times d$	Squared eigenvalues; diagonal
Y	$1 \times n$	Dependent variable
$\boldsymbol{X}$	$(d+1) \times n$	Augmented matrix with 1s
β	$(d+1) \times 1$	Regression coefficients including Y intercept
γ	$d \times 1$	Selected variables; binary vector
$\mathbf{Z}$	$(d+2) \times n$	Augmented matrix with 1s and Y
Γ	$(d+2) \times (d+2)$	Generalized summarization matrix

Sufficient Statistics

Old:

$$n = |X|$$

$$L = \sum_{i=1}^{n} x_i$$

$$Q = XX^T = \sum_{i=1}^{n} x_i \cdot x_i^T$$

We introduce the Γ matrix as a comprehensive accurate summary of X, sufficient to compute all the models discussed in this paper:

New: $\mathbf{Z} = [1, X, Y]$; $\Gamma = \mathbf{Z}\mathbf{Z}^T$

$$\Gamma = \begin{bmatrix} \mathbf{Q} & \mathbf{X}Y^T \\ Y\mathbf{X}^T & YY^T \end{bmatrix}$$

$$= \begin{bmatrix} n & L^T & \sum y_i \\ L & Q & XY^T \\ \sum y_i & YX^T & YY^T \end{bmatrix}$$

$$= \begin{bmatrix} n & \sum x_i^T & \sum y_i \\ \sum x_i & \sum x_i x_i^T & \sum x_i y_i \\ \sum y_i & \sum y_i x_i^T & \sum y_i^2 \end{bmatrix}$$

Matrix Γ is comparatively much smaller than X for big data and is symmetric.

Based on Γ our algorithm to compute Θ works in two phases:

1. Compute Γ
2. Exploit Γ in intermediate matrix computations.

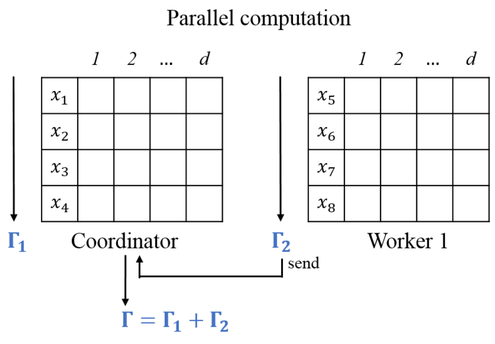
Algorithm

Sequential algorithm to compute Γ:

Data: $X = \{x_1, x_2, \ldots, x_n\}, Y = \{y_1, \ldots, y_n\}$
Output: Γ

$\mathbf{Z} \leftarrow [1, X, Y] = \{z_1, z_2, \ldots, z_n\}$
$\Gamma \leftarrow \mathbf{0}$
for $i = 1 \ldots n$ **do**
  for $a = 0 \ldots d+1$ **do**
    for $b = 0 \ldots d+1$ **do**
      $\Gamma_{ab} \leftarrow \Gamma_{ab} + z_{ia} * z_{ib}$
    end
  end
end

Given the additive properties of Γ, the same algorithm is applied on each node $I = 1 \ldots N$ to get Γ_I, then we combine all partial results to get the global $\Gamma = \sum_I \Gamma_I$ in the coordinator node. Our operator requires that the whole data point z_i fits in one chunk. In other words, z_i cannot be partitioned into several chunks.



COMPUTATION OF STATISTICAL MODELS USING Γ

PCA:
$$\rho_{ab} = \frac{nQ_{ab} - L_a L_b}{\sqrt{nQ_{aa} - L_a^2}\sqrt{nQ_{bb} - L_b^2}}$$

Obtain eingeinvalues and eingenvectors: SVD(ρ)

LR:
$$\hat{\beta} = (\mathbf{Q})^{-1}\mathbf{X}Y^T$$

VS:
$$\pi(\gamma|X,Y) \propto (c+1)^{-\frac{k+1}{2}} \left(YY^T - \frac{c}{c+1}(Y\mathbf{X}_\gamma{}^T)(\mathbf{Q}_\gamma)^{-1}(\mathbf{X}_\gamma Y^T) - (c+1)^{-1}\tilde{\beta}_\gamma^T \mathbf{Q}_\gamma \tilde{\beta}_\gamma\right)^{-n/2}$$

Experiments

- **In R**: load the data set into RAM then do the transposition and multiplication.
- **SQL:** do self join on the data set.
- **AQL/AFL in SciDB:** do self cross join on the data set.
- **SciDB operators:** work directly on the array.

Comparing only summarization; data set KDDnet; d=100; 2 nodes; time in secs.

n	R	SQL queries	Γ operator
10k	0.9	3.1	0.3
100k	6.5	27.1	1.3
1M	44.1	612.5	13.7
10M	fail	13286.5	150.2

Computing model Θ; data set KDDnet; d=100; 1 node; time in secs.

n	PCA		LR		VS	
	R	Γ operator + R	R	Γ operator + R	R	Γ operator + R
10k	0.7	0.8	0.7	0.9	13.1	13.0
100k	5.7	2.5	6.3	2.6	34.4	14.8
1M	61.2	16.8	60.5	16.9	113.0	29.1
10M	fail	194.9	fail	194.9	fail	207.2

Comparing SciDB programming mechanisms; data set KDDnet; d=100; 1 node; time in secs.

n	AQL/AFL	spgemm()	SciDB-R	Γ operator
10k	168.3	3.1	19.1	0.3
100k	stop	15.3	76.8	1.3
1M	stop	177.8	stop	13.7
10M	stop	fail	stop	150.2

All experiments are performed under default cluster configuration: 1 node running 2 instances. Each node has 4 cores, 4GB RAM and 3TB disk space.