Computation of PDFs on Big Spatial Data: Problem & Architecture

Ji Liu, Noel Lemus, Esther Pacitti, Fábio Porto, Patrick Valduriez

Computation of PDFs on Big Spatial Data: Problem & Architecture

Ji Liu[1], **Noel M. Lemus**[2], **Esther Pacitti**[1], **Fabio Porto**[2], **Patrick Valduriez**[1]

[1]Inria and LIRMM, Univ. of Montpelier, France

{ji.liu, patrick.valduriez}@inria.fr, esther.pacitti@lirmm.fr

[2]LNCC Petrópolis, Brazil

nmlemus@gmail.com, fporto@lncc.br

Abstract. *Big spatial data can be produced by observation or numerical simulation programs and correspond to points that represent a 3D soil cube area. However, errors in signal processing and modeling create some uncertainty, and thus a lack of accuracy in identifying geological or seismic phenomenons. To analyze uncertainty, the main solution is to compute a Probability Density Function (PDF) of each point in the spatial cube area, which can be very time consuming. In this paper, we analyze the problem and discuss the use of Spark to efficiently compute PDFs.*

1. Introduction

Big spatial data is now routinely produced and used in scientific areas such as geological or seismic interpretation [Campisano et al. 2016]. The spatial data are produced by observation, using sensors [Chen et al. 2014], or numerical simulation [Cressie 2015]. These spatial data allow identifying some phenomenon over a spatial reference. For instance, the spatial reference may be a three dimensional soil cube area and the phenomenon a seismic fault, represented as quantities of interest (QOIs) of sampled points (or points for short) in the cube space. The cube area is composed of multiple horizontal slices, each slice having multiple lines and each line having multiple points. A single simulation produces a spatial data set whose points represent a 3D soil cube area.

However, errors in signal processing and modeling create some uncertainty, and thus a lack of accuracy when identifying phenomenons. In order to understand uncertainty, several simulation runs with different input parameters are usually conducted, thus generating multiple spatial data sets that can be very big, *e.g.* hundreds of GB or TB. Within multiple spatial data sets, each point in the cube area is associated to a set of different observation values in the spatial data sets. The observation values are those observed by sensors, or generated from simulation, at a specific point of the spatial area.

Uncertainty quantification of spatial data is of much importance for geological or seismic scientists [Trajcevski 2011]. It is the process of quantifying the uncertainty error of each point in the spatial cube space, which requires computing a Probability Density Function (PDF) of each point [Kathryn et al. 2015]. The PDF is composed of the distribution type (*e.g.* normal, exponential) and necessary statistical parameters (*e.g.* the mean and standard deviation values for normal and rate for exponential). However, calculating the PDF at each point can be time consuming, *e.g.* several days or months to process 2.4 TB data for an area of 10km (distance) * 10km (depth) * 5km (width).

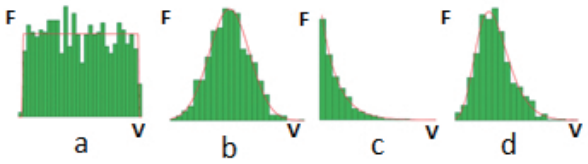

Figure 1. The distribution of a point.

In this paper, we take advantage of Spark [Zaharia et al. 2010], a popular in-memory big data processing framework for computer clusters, to efficiently compute PDFs in parallel. In the paper, we detail the problem, *i.e.* how to efficiently compute PDFs, and propose an architecture using Spark to compute PDFs.

2. Problem Definition

Figure 1 shows that the set of observation values at a point may have four distribution types, *i.e.* uniform (a), normal (b), exponential (c), and log-normal (d). The horizontal axis represents the values (V) and the vertical axis represents the frequency (F). The green bars represent the frequency of the observation values in value intervals and the red outline represents the calculated PDF. During the calculation of the PDF at a point, there may be some error between the distribution of the observation values and the calculated PDF. We denote this error by PDF error (or error for short in the rest of the paper). In order to precisely fit a PDF based on observation values, we need to reduce this error. Once we have the PDF of a point, we can calculate the QOI value that has the highest possibility, with which we can quantify the uncertainty each spatial data set.

Let DS be a set of spatial data sets, d_k be a spatial data set in DS and N be the number of points in a region. Each point $p_{x,y}$, where x and y are spatial dimensions in the slice, has a set of values $V = \{v_1, v_2, ..., v_n\}$ while v_k is the value corresponding to the point $p_{x,y}$ in $d_k \in DS$. Based on these notations, we define Equations 1 - 4, which are based on the formulas in [Dixon and Massey 1968]. The mean ($\mu_{x,y}$) and standard deviation ($\sigma_{x,y}$) values of a point can be calculated according to Equations 1 and 2, respectively. The error $e_{x,y,i}$ between the PDF F and the set of observation values V can be calculated according to Equation 3, which compares the probability of the values in different intervals in V and the probability computed according to the PDF. The intervals are obtained by evenly splitting the space between the maximum value and the minimum value in V. min is the minimum value in V, max is the maximum value in V and L represents the number of all considered intervals, which can be configured. $Freq_k$ represents the number of values in V that are in the kth interval. The integral of $PDF(x)$ computes the probability according to the PDF in the kth interval. Equation 3 is inspired by the Kolmogorov-Smirnov Test [Lopes 2011], which tests whether a PDF is adequate for a data set. In addition, we assume that the probability of the values outside the space between the maximum value and the minimum value is negligible for this equation. Then, the average error E of Slice i can be calculated according to Equation 4.

$$\mu_{x,y} = \frac{\sum_{i=1}^{n} v_i}{n} \tag{1}$$

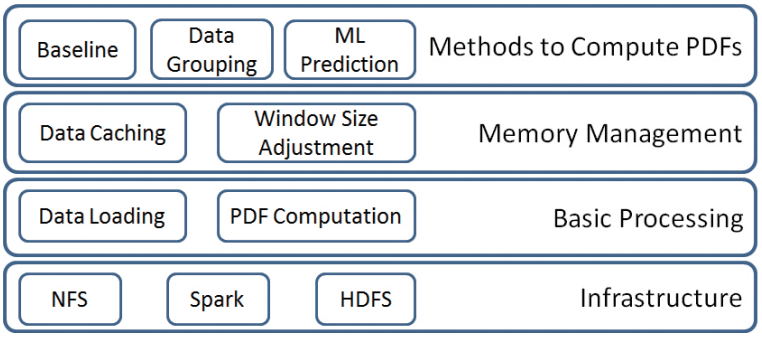

Figure 2. Architecture for Computing PDFs.

$$\sigma_{x,y} = \sqrt{\frac{\sum_{i=1}^{n}(v_i - \mu)^2}{n-1}} \tag{2}$$

$$e_{x,y,i} = \sum_{k=1}^{N} \mid \frac{Freq_k}{N} - \int_{min+(max-min)*\frac{k-1}{L}}^{min+(max-min)*\frac{k}{L}} PDF(x)dx \mid \tag{3}$$

$$E = \frac{\sum_{p_{x,y} \in slice_i} e_{x,y,i}}{N} \tag{4}$$

We can now express the problem as follows: given a set of spatial data sets DS corresponding to the same spatial cube area $C = \{slice_1, slice_2, ..., slice_j\}$, how to efficiently calculate the mean, standard deviation values and the PDF F at each point in $slice_i \in C$ with a small average error E not higher than a predefined average error ε.

3. Architecture

The architecture (see Figure 2) has four layers, *i.e.* infrastructure, basic process to compute PDFs, memory management and methods to compute PDFs. In the infrastructure layer, the big spatial data is stored in NFS [Sandberg et al. 1985]. Spark and HDFS [Shvachko et al. 2010] are deployed over the nodes of the computer cluster. The intermediate and the output data are stored in HDFS.

The basic processing of PDFs consists of data loading, from NFS to Spark RDDs, followed by PDF computation using Spark. The data loading process treats the data corresponding to a slice and pre-processes it in parallel, *i.e.* calculates statistical parameters of observation values of each point. Then, the PDF computation groups the data and calculates the PDFs and errors of all the points in a slice in parallel.

In order to efficiently compute PDFs, we use two memory management techniques: data caching and window size adjustment. We use data caching to reduce disk accesses with the Spark $Cache$ operation and a memory-based file system [Snyder 1990]. We test the Scala program on a small workload with different window sizes to find an optimal window size, which is used for the PDF computation of all the points in the slice.

We use two methods to compute PDFs efficiently, *i.e.* data grouping and ML prediction. The data grouping method groups the points with exactly the same mean and

standard deviation values into a group, using the *aggregation* operation in Spark. Then, one point is selected for each group to compute the PDFs. The ML prediction method is based on a decision tree to predict the distribution type to compute PDFs for each point.

4. Experimental Evaluation

To validate our solution, we implement the two methods in a Spark cluster and performed extensive experiments using three big spatial data sets of from 235 GB to 2.4 TB, generated based on the seismic benchmark of the HPC4e project [HPC4E]. The experimental results show that our solution is efficient and scales up very well compared with Baseline, *i.e.* brute-force method without using data grouping and ML prediction. Grouping outperforms Baseline by up to 92% (more than 10 times) without introducing extra error. ML can be up to 91% (more than 9 times) better than Baseline with very slight acceptable extra error (up to 0.017). The combination of Grouping and ML can be up to 97% (more than 33 times) better than Baseline with an acceptable extra error (up to 0.017).

References

Campisano, R., Porto, F., Pacitti, E., Masseglia, F., and Ogasawara, E. S. (2016). Spatial sequential pattern mining for seismic data. In *Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 241–246.

Chen, M., Mao, S., and Liu, Y. (2014). Big data: A survey. *Mobile Networks and Applications*, 19(2):171–209.

Cressie, N. (2015). *Statistics for spatial data*. John Wiley & Sons.

Dixon, W. J. and Massey, F. J. (1968). *Introduction to statistical analysis*.

HPC4E. HPC Geophysical Simulation Test Suite. `https://hpc4e.eu/downloads/hpc-geophysical-simulation-test-suite`.

Kathryn, F., Oden, J. T., and Faghihi, D. (2015). A bayesian framework for adaptive selection, calibration, and validation of coarse-grained models of atomistic systems. *Journal of Computational Physics*, 295:189 – 208.

Lopes, R. H. C. (2011). Kolmogorov-smirnov test. In *Int. Encyclopedia of Statistical Science*, pages 718–720.

Sandberg, R., Goldberg, D., Kleiman, S., Walsh, D., and Lyon, B. (1985). Design and implementation of the sun network file system. In *the Summer USENIX conf.*, pages 119–130.

Shvachko, K., Kuang, H., Radia, S., and Chansler, R. (2010). The hadoop distributed file system. In *IEEE Symp. on Mass Storage Systems and Technologies (MSST)*, pages 1–10.

Snyder, P. (1990). tmpfs: A virtual memory file system. In *European UNIX Users Group Conf.*, pages 241–248.

Trajcevski, G. (2011). Uncertainty in spatial trajectories. In *Computing with Spatial Trajectories*, pages 63–107.

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., and Stoica, I. (2010). Spark: Cluster computing with working sets. In *USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*.