MAIN DIFFERENCES BETWEEN MAP/REDUCE AND COLLECT/REPORT PARADIGMS

Krassimira Ivanova

Abstract*: This article presents main differences between Map/Reduce (MRP) and Collect/Report (CRP) paradigms. The most important difference is that in MRP the calculations and data must be all completely independent. In opposite, the CRP assumes that all data are interconnected and may be processed in common, taking in account all interconnections.*

Keywords*: Map/Reduce Paradigm, Collect/Report Paradigm, Big Data, Cloud computing*

ACM Keywords: *E.1 Data Structures; Distributed data structures.*

Introduction

Nowadays, *data-intensive* computing problems are emerging. In contrast to the traditional computing problems, data-intensive problems demonstrate the following features [Lockwood, 2015]:

- **Input data is far beyond gigabyte-scale**: datasets are commonly on the order of tens, hundreds, or thousands of terabytes;
- **They are I/O-bound**: it takes longer for the computer to get data from its permanent location to the CPU than it takes for the CPU to operate on that data.

The Map/Reduce Paradigm (MRP) is a way of solving a certain subset of parallelizable problems that gets around the bottleneck of ingesting input data from disk. Whereas traditional parallelism brings *the data to the compute*, map/reduce does the opposite, it brings *the compute to the data.* Below we will remember the main features of map/reduce paradigm following the work [Lockwood, 2015].

In Map/Reduce, the input data is not stored on a separate, high-capacity storage system. Rather, the data exists in little pieces and is permanently stored on the compute elements. This allows our parallel procedure to follow these steps:

1. We do not have to move any data since it is pre-divided and already exists on nodes capable of acting as computing elements;
2. All of the parallel worker functions are sent to the nodes where their respective pieces of the input data already exist and do their calculations;

3. All of the parallel workers communicate their results with each other move data if necessary, and then continue the next step of the calculation.

Thus, the only time data needs to be moved is when all of the parallel workers are communicating their results with each other in point 3 above. There is no more serial step where data is being loaded from a storage device before being distributed to the computing resources because the data already exists on the computing resources.

Hadoop is an actual implementation of Map/Reduce. Hadoop, perhaps the most widely used Map/Reduce framework, accomplishes this feat using the Hadoop Distributed File System (HDFS). HDFS is fundamental to Hadoop because it provides the data chunking and distribution across compute elements necessary for Map/Reduce applications to be efficient.

The main features of Map/Reduce and Hadoop are:

- Map/Reduce brings *compute to the data* in contrast to traditional parallelism, which brings data to the compute resources;
- Hadoop accomplishes this by storing data in a replicated and distributed fashion on HDFS;
- HDFS stores files in chunks which are physically stored on multiple compute nodes;
- HDFS still presents data to users and applications as single continuous files despite the above fact;
- Map/Reduce is ideal for operating on very large, flat (unstructured) datasets and perform trivially parallel operations on them;
- Hadoop jobs go through a Map stage and a Reduce stage where:
    - The Mapper transforms the raw input data into key-value pairs where multiple values for the same key may occur;
    - The Reducer transforms all of the key-value pairs sharing a common key into a single key with a single value.

Of course, for the compute elements to be able to do their calculations on these chunks of input data, the calculations and data must be all completely independent from the input data on other compute elements. This is the principal constraint in Map/Reduce jobs: *Map/Reduce is ideally suited for trivially parallel calculations on large quantities of data*, but if each worker's calculations depend on data that resides on other nodes, one will begin to encounter rapidly diminishing returns [Lockwood, 2015].

This constraint causes the need of principally other paradigm for storing and processing Big Data. Such paradigm is so called “Collect/Report Paradigm” [Markov & Ivanova, 2015]. The goal of this paper is to outline the main differences between these two paradigms as well as to propose a possible convergent paradigm which may unite the positive features of both paradigms.

Collect/Report Paradigm

The Collect/Report Paradigm is based on the possibility of so called "Natural Language Addressing" (NLA) [Markov et al, 2015].

CRP assumes that incoming information is coded in RDF format. In Collect/Report Paradigm, all nodes have to "listen" in parallel the incoming stream of RDF-data and to "collect" (to store) information only in the layers the nodes have to support. In the same time, nodes have to "listen" incoming stream of requests and only nodes, which have information corresponded to given request has to "report" (to send answer).

Main advantages of Collect/Report Paradigm are:

- *Collecting information is done by all nodes independently in parallel. It is possible one node to send information to another;*
- *Reporting information is provided only by the nodes which really contain information related to the request; the rest nodes do not react, they remain silent;*
- *Input data as well as results are in RDF-triple or RDF-quadruple format.*

CRP is a good foundation for intelligent data processing based on multi-dimensional memory structures [Markov et al, 2013]. In addition, via CRP and natural language addressing, three main problems of storing Big Data may be solved [Markov et al, 2014]:

- *Volume – avoiding additional indexing, duplication of keywords, and corresponded pointers, leads to reducing additional memory needed for accessing information i.e. we may use addressing but not classical search engines;*
- *Velocity – avoiding recompilation of information base permits high speed of storing and immediately readiness of information to be accessed. This is very important possibility for stream data;*
- *Variety – natural language addressing permits creating a special kind of graph information bases which may operate both with structured as well as semi-structured information.*

Main Differences Between Map/Reduce and Collect/Report Paradigms

Map/Reduce Paradigm (MRP) and Collect/Report Paradigm (CRP) are two different approaches for operating on very large, flat (unstructured) datasets (Big Data). The main differences between these two paradigms may be systematizes as follow:

- MRP performs trivially parallel operations and results are couples (keyword, value). The CRP is designed to cover the case when the data are represented in RDF-format and the results are triples (subject, relation, object);
- *In CRP, reporting information is provided only by the nodes which really contain information related to the request; the rest nodes do not react, they remain silent. In MRP all nodes send resulting information assuming that all reported data is needed for end user;*
- *An important advantage of the CRP is reducing the traffic to and from the cloud structures for storing data in RDF-format and readiness to extract information within microseconds after it has been stored;*
- *The most important difference is that in MRP the calculations and data must be all completely independent. In opposite, the CRP assumes that all data are interconnected and may be processed in common, taking in account all interconnections.*

Conclusion

In this article we have outlined the main differences between the Map/Reduce and Collect/Report paradigms. Concluding, we may propose a convergent paradigm “Map/Collect/Report” (MCRP).

MRP expects that the data is pre-divided and already exists on nodes capable of acting as computing elements. After Mapping phase the data is formatted in format of couples (keyword, value).

CRP expects that the incoming data is in RDF format. How the data are prepared in RDF format is not commented. One possible variant is to use mapping, similar to MRP but to generate triples as intermediate result.

This way, in MCRP, we have the sequence:

- Mapping Big Data to RDF-triples or quadruples;
- Collecting RDF information in hyper-graph structures;
- Reporting only requested information.

Bibliography

[Lockwood, 2015] Glenn K. Lockwood, “Conceptual Overview of Map/Reduce and Hadoop”, http://www.glennklockwood.com/di/hadoop-overview.php#compare (accessed on 23.02.2015)

[Markov & Ivanova, 2015] Krassimir Markov, Krassimira Ivanova, “General Structure of Collect/Report Paradigm for Storing and Accessing Big Data”, International Journal “Information Theories and Applications”, Vol. 22, Number 3, 2015, ISSN 1310-0513 (printed), ISSN 1313-0463 (online), pp. 266 - 276.

[Markov et al, 2013] Krassimir Markov, Koen Vanhoof, Iliya Mitov, Benoit Depaire, Krassimira Ivanova, Vitalii Velychko and Victor Gladun, "Intelligent Data Processing Based on Multi-Dimensional Numbered Memory Structures", Diagnostic Test Approaches to Machine Learning and Commonsense Reasoning Systems, IGI Global, 2013, pp. 156-184, doi: 10.4018/978-1-4666-1900-5.ch007, ISBN: 978 1-4666-1900-5, EISBN: 978-1-4666-1901-2

[Markov et al, 2014] Krassimir Markov, Krassimira Ivanova, Koen Vanhoof, Benoit Depaire, Vitalii Velychko, Juan Castellanos, Levon Aslanyan, Stefan Karastanev, „Storing Big Data Using Natural Language Addressing", In: N. Lyutov (ed.), int. Sc. Conference "Informatics in the Scientific Knowledge", VFU, Varna, Bulgaria, 2014, ISSN: 1313-4345, pp. 147-164.

[Markov et al, 2015] Krassimir Markov, Krassimira Ivanova, Koen Vanhoof, Vitalii Velychko, Juan Castellanos, „Natural Language Addressing", ITHEA® Hasselt, Kyiv, Madrid, Sofia, IBS ISC No.: 33, 2015, ISBN: 978-954-16-0070-2 (printed), ISBN: 978-954-16-0071-9 (online), 315 p.

Authors' Information

Krassimira Ivanova *– University of National and World Economy, Sofia, Bulgaria;*

e-mail: krasy78@mail.bg

Major Fields of Scientific Research: Software Engineering, Business Informatics, Data Mining, Multidimensional multi-layer data structures in self-structured systems