Computers and Electrical Engineering 000 (2018) 1–13

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Exact and heuristic MapReduce scheduling algorithms for cloud federation☆

Thouraya Gouasmi[a,b,*], Wajdi Louati[a,c], Ahmed Hadj Kacem[a,b]

[a] ReDCAD Lab, University of Sfax, Tunisia
[b] Faculty of Economics and Management of Sfax, Tunisia
[c] National Engineering School of Sfax, Tunisia

ARTICLE INFO

Article history:
Received 15 June 2017
Revised 16 January 2018
Accepted 16 January 2018
Available online xxx

Keywords:
MapReduce
Cloud federation
Distributed scheduling
Cost optimization

ABSTRACT

Geo-distributed big-data processing has recently received much attention since it ensures large-scale and geographically distributed data processing, using Hadoop or Spark, in an efficient, fault-tolerant and reliable manner. The objective of this work is to propose a new geo-distributed MapReduce-based framework and algorithm for federated cloud platforms. A distributed heuristic algorithm, called FDMR (Federated Distributed MapReduce), that takes advantage of data locality, inter-cloud data transfer and high availability of capacities offered by the federation is proposed. The aim of FDMR is to reduce job cost while respecting deadline constraint. The goal of this paper is also to propose an exact MapReduce scheduling model to serve as a baseline for benchmarking and to compare and discuss the heuristic algorithm results. The performance evaluation proves that the proposed algorithm FDMR can improve resource utilization of the cloud federation and consequently reduce cost and job response time while satisfying the deadline constraint.

1. Introduction

Currently, several big-data processing frameworks, e.g. Hadoop and Spark, are offered as a service by various private or public cloud computing platforms like Amazon Web Services, Microsoft Azure, Google App Engine and IBMs Blue Cloud. These frameworks ensure large-scale data processing in an efficient and fault-tolerant manner in several fields including log analysis, web indexing , pattern matching, scientific analysis and business intelligence. However, few applications require geo-distributed datasets processing (e.g. process data at different locations/sites) [1] like biological data processing, web crawling, stream and social network analysis. Existing frameworks do not consider geo-distributed data locations, e.g. the all raw data is moved from different locations to a single location before the computation which can be a drawback in terms of security, privacy, performance and network utilization (e.g. bottleneck in terms of data transfer). Several proposals, presented in this survey [2], tried to rethink the current big-data processing frameworks to process geographically distributed data at their locations without moving entire raw data to a single location. The novel frameworks consider several factors including task assignment, data locality (e.g. data processing at the same site where the data is located), data movement (e.g. moving only the desired data while reducing the need for high bandwidth), and security and privacy. Since our work only considers

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. L. Bittencourt.

* Corresponding author at : ReDCAD Lab, University of Sfax, Tunisia.
E-mail addresses: thouraya.gouasmi@redcad.org (T. Gouasmi), wajdi.louati@redcad.org (W. Louati), ahmed.hadjkacem@redcad.org (A.H. Kacem).

https://doi.org/10.1016/j.compeleceng.2018.01.021
0045-7906/

geo-distributed MapReduce-based systems with a special focus on data locality and data movement, existing frameworks and algorithms consider either data locality (like G-Hadoop [3], GMR [4], Nebula [5]) or data movement (like Rout [6], and Meta-MapReduce [7]). To the best of our knowledge, no work has jointly considered data locality and data movement. In fact, the existing proposals addressing data locality suffer from limited performance and offer higher cost due to the limited number of available resources per cluster or due to the slow inter-cluster connections. On the other hand, the works focusing on data transfer propose distributed and reliable algorithms for moving desired data among clusters while minimizing remote access and reducing the cost, but do not provide optimized allocation of inter-cluster bandwidth.

The objective of this work is to design a new geo-distributed MapReduce-based framework and algorithm called FDMR (Federated Distributed MapReduce) for federated Clouds, that basically rely on high speed inter-cloud networks. The proposed framework considers both data locality and movement where the objective is to reduce both the job response time and the cost by allocating idle resources from the cloud federation. A heuristic scheduling Algorithm, that takes advantage of data locality and extra-capacity offered by other clusters while reducing both VM cost and data transfer cost subject to Deadline constraint is proposed. Furthermore, an exact approach formulated and solved as a mixed integer program is proposed to provide a baseline for benchmarking and evaluation of the heuristic scheduling algorithm FDMR. Performance evaluation proves that the proposed algorithm FDMR can efficiently reduce MapReduce job cost while ensuring optimal resource allocation.

The rest of the paper is organized as follows. Section 2 outlines related work. Section 3 presents the FDMR Architecture and components. Section 4 describes the problem definition, the operational model and the analytical model. Exact and Heuristic Scheduling algorithms for Map tasks allocation in cloud federation are described in the Section 5. The Section 6 presents the implementation and evaluation results of both approaches.

2. Related work

Several geo-distributed big-data processing have been proposed in the literature. Authors in [2] presented a survey of recent advances in this field and tried to classsify geo-distributed big-data processing frameworks and algorithms into two categories: User-located geo-distributed data and Pre-located geo-distributed data.

In the first category, both the data as well as the related jobs are distributed over different clusters by the user site. Aggregation of outputs of all the sites is not required and depends on the job. Some MapReduce frameworks for user-located geo-distributed data like BStream [8], Hierarchical MapReduce (HMR) [9], HybridMR [10] are recently proposed.

In the second category, jobs are distributed over different clusters by the user site. Data is already geo-distributed before the computation. Outputs of all the sites are then aggregated at a specified site. Examples of MapReduce-based frameworks for pre-located geo-distributed data include G-Hadoop [3], GMR [4], Nebula [5], Medusa [11], Fed-MR [12]. Wang et al. proposed G-Hadoop [3] framework for processing geo-distributed data across multiple clusters. They designed a new Hadoop scheduler in which one datacenter presents a Master node responsible for dividing and distributing job computing across others datacenters acting as worker nodes. G-Hadoop may ensure fault-tolerance by duplicating map and reduce tasks across multiple cluster nodes. GMR in [4] tried to improve the MapReduce architecture by determining the best execution path of job sequence across datacenters. G-MR consists of a single component named GroupManager executing a data transformation graph (DTG) algorithm to determine an optimized path for performing MapReduce jobs, and JobManagers components deployed in each of the participating data centers. The JobManagers uses two additional components, a CopyManager that copies of data in data centers and an AggregationManager that aggregates results from these data centers. Medusa [11] proposes a framework to optimize resource allocation and WAN traffic movement for geo-distributed MapReduce processing. Medusa does not provide any new concept except handling new types of faults.

Our proposed FDMR Framework is related to the second category while taking into account data locality[13]. The existing MapReduce-based frameworks for pre-located geo-distributed data (e.g. G-Hadoop, GMR, Nebula) suffer from limited performance and offer higher cost due to the limited number of available resources per cluster or due to the slow inter-cluster connections. Since FDMR is conceived for federated clouds that basically rely on high speed inter-cloud networks, our work considers (in addition to data locality) the bandwidth availability to allow migration of data and jobs from a cloud to another. The objective is to reduce both the job response time and the cost by allocating idle resources from the cloud federation.

3. FDMR architecture and components

Fig. 1 shows the FDMR architecture and its components. The proposed architecture is partially inspired from the BStream one [8] and extends it to include cloud federation techniques including migration and resource pooling. First, the user sends the MapReduce job to the Controller (C) of cloud A and specifies its deadline (step 1). The Controller (C) distributes the Map (M) and Reduce (R) tasks (MapReduce sub-jobs) to all Controllers (C) in the Federation (step 2). Each Controller (C) relies on the Estimator (E) for resource allocation (step 3). The Estimator (E) refers a job database (DB) to estimate: the capacities of clusters, the input data size, the number of Map tasks, the network configuration, etc (step 4). If the estimated resource allocation returned to the controller (C) by the estimator (E) (step 5) exceeds the available resources (slots) in the cloud, then the controller (C) initializes the FDMR (step 6) to decide which Map tasks (M) should be moved and to which cloud they will be placed, and sends mapping configurations to the Zookeeper (Z) to store them. The FDMR uses Dispatcher (D) to

Please cite this article as: T. Gouasmi et al., Exact and heuristic MapReduce scheduling algorithms for cloud federation, Computers and Electrical Engineering (2018), https://doi.org/10.1016/j.compeleceng.2018.01.021