Received July 27, 2020, accepted September 2, 2020, date of publication September 10, 2020, date of current version September 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023344

Data Processing Model to Perform Big Data Analytics in Hybrid Infrastructures

JULIO C. S. DOS ANJOS[1], KASSIANO J. MATTEUSSI[1], (Member, IEEE), PAULO R. R. DE SOUZA, JR.[1], GABRIEL J. A. GRABHER[1], GUILHERME A. BORGES[1], JORGE L. V. BARBOSA[2], GABRIEL V. GONZÁLEZ[3], VALDERI R. Q. LEITHARDT[4,5,6], (Member, IEEE), AND CLAUDIO F. R. GEYER[1]

[1]Federal University of Rio Grande do Sul, Institute of Informatics, UFRGS/PPGC, Porto Alegre 91501-970, Brazil
[2]University of Vale do Rio dos Sinos, UNISINOS/PPGCA, São Leopoldo 93022-750, Brazil
[3]Faculty of Science, Expert Systems and Applications Laboratory, University of Salamanca, 37008 Salamanca, Spain
[4]VALORIZA Research Center, Instituto Politécnico de Portalegre, 7300-110 Portalegre, Portugal
[5] Laboratory of Embedded and Distributed Systems, University of Vale do Itajai, Itajaí 88302-901, Brazil
[6]Universidade Lusófona de Humanidades e Tecnologias, COPELABS, 1700-097 Lisbon, Portugal

Corresponding author: Julio C. S. Dos Anjos (jcsanjos@inf.ufrgs.br)

This work was supported in part by "SmartSent" (#17/2551-0001 195-3), CAPES (Finance Code 001), PNPD program, CNPq, PROPESQ-UFRGS-Brasil and FAPERGS Project "GREEN-CLOUD - Computação em Cloud com Computação Sustentável" (#16/2551-0000 488-9), and in part by the Junta De Castilla y León—Consejería De Economía Y Empleo: System for simulation and training in advanced techniques for the occupational risk prevention through the design of hybrid-reality environments with ref J118 and by Fundação para a Ciência e a Tecnologia under Project UIDB/04111/2020.

ABSTRACT Big Data applications are present in many areas such as financial markets, search engines, stream services, health care, social networks, and so on. Data analysis provides value to information for organizations. Classical Cloud Computing represents a robust architecture to perform complex and large-scale computing for these areas. The main challenges are the user's unknowledge about Cloud infrastructure, the requirement needed for improving performance, and the resource management to maintain stable processing. In these difficulties, an inadequate solution can lead to users overestimate or underestimate the number of computational resources, which drives to the budget increases. One way to work around this problem is to make use of Volunteer Computing since it provides distributed computational resources at free monetary cost. However, a volatile machine behavior is a problem to address in Big Data data distributions. Thus, this work proposes a data distribution model composed of Cloud Computing and Volunteer Computing environments in a hybrid fashion for Big Data analytics. The contributions of this work are: i) the required evaluation to enable efficient deployment of Big Data in hybrid infrastructures; ii) the development of an HR_Alloc Algorithm for establishing the data placement to Big Data applications; iii) a model to resource allocation in hybrid infrastructures. The obtained results indicate the feasibility of using a hybrid infrastructure with up to 35% of unstable machines in the worst-case scenario, without losing performance and a monetary cost lower than 20% in comparison to Classical Cloud Computing. Also, communication costs decrease up to 57.14% in the best-case scenario due to load balancing.

INDEX TERMS Big data analytics, cloud computing, hybrid infrastructures, MapReduce, volunteer computing.

I. INTRODUCTION

The increasing use of electronic devices such as smartphones, tablets, and sensors on the Internet has led to the generation of massive volumes of data. The International Data Corporation (IDC) estimates an exponential growth in data production, and it is moving ahead from 33 Zettabytes in 2018 to 135 Zettabytes in 2025 [1]. In such context, the rise of real-time Big Data analysis has promoted the creation and adoption of many applications and solutions in the most varied domains such as social networks, stock market, education, astronomy, meteorology, bioinformatics, exact sciences, social sciences, and others.

In fact, the possibility of handling massive amounts of data in real-time has attracted the attention of many organizations that are seeking efficient, low-cost strategies for data

The associate editor coordinating the review of this manuscript and approving it for publication was Gianmaria Silvello.

processing and analysis in order to provide valuable solutions and services [2]. Along these lines, frameworks and libraries have been made to handle Big Data analysis in different contexts. It can cite, for instance, the adoption of Hadoop MapReduce (HM) [3] for batch processing, Spark [4], Flink [5] and Storm [6] for batch and real-time processing, the last processing most utilized in Internet of Things (IoT) scenarios.

Further, these frameworks need environments capable of supporting data-intensive processing, enable on-demand services, and varied workloads. This is where Classical Cloud Computing (CCC) fits with on-demand self-service, broad network access, resource pooling, and rapid elasticity on a pay-as-you-go model [7]. These characteristics revolutionized how data analysis has led in the last past decade, as well as, motivated both academic and industrial organizations to investigate and develop technologies and solutions for supporting a world in transformation [8].

CCC transformed the entire computing industry and business models [8] and currently has been shifting from local storage and processing to online processing at the network Edge. It does not change the fact that CCC represents a robust architecture with efficient management to perform large-scale and complex computing in a distributed and decentralized fashion [9]. Nevertheless, data-intensive processing at the cloud level presents a bulk of challenges such as data movement, and communication, as well as, the difficulty of service optimizing for supporting billions of end-devices in real-time [10]–[12].

Volunteer Computing (VC) represents a kind of improvement for the needs mentioned above and also is known as a heterogeneous platform that allows underutilized users' resources to allow perform a wide range of applications at a fraction of the cost of classical High-Performance Computing (HPC) clusters, or CCC providers. For instance, an application running on top of a VC infrastructure runs data task closer to users at the Edge of the network. Thus, this approach can be 56% more efficient than CCC environments as well as IoT applications can reach up to 5.3 times faster due to low levels of latency and lower data-movement [13].

At the same direction, our recent studies [16], [17] and [18] reinforce the fact that classical VC can be used to balance expenditures costs and processing between volunteers and CCC instances. In summary, the provided solutions to schedule tasks using a data locality algorithm that follows computers' capacities, i.e., the machines with more computational power will receive more data to process than others. Also, the machines join in groups and receive data replicas to reduce network traffic, data movement, and latency. In such context, it is possible that VC or Volunteer Cloud Computing (VCC) devices (as described in Darrous *et al.* [13]) are capable of supporting data-intensive processing at the Fog Computing (Fog) and Edge layers. It is because Fog resources have been representing a CCC extension for real-time data analytics, due to Cloud has high bandwidth, reliability, and locality-aware [19].

Besides, it is important to mention that the behavior of volatility in a VC environment - is quite similar to what is found in Fog scenarios. In such context, the Fog nodes are aware of their geographical distribution and logical location within the cluster context, operating in a distributed manner over varied network conditions [20]. Thus, if it is possible to abstract problems such as resource discovery, intermittent availability, and security, then Volunteer Desktop Cloud (VDC) can be used as well in Fog environments. VDC is a cloud environment composed of volunteer desktops with trusted relationships. VDC is not limited by battery charge time or low processing capacity like Edge devices. Then these environments can be eligible to run Big Data applications flowing determined requirements more efficiently.

The main motivation of this work is to investigate if it is possibility to adopt adjacent technologies such as CCC and VDC environments to work together to provide a hybrid infrastructure called Hybrid Cloud for Big Data Processing (HCBDP) for performing Big Data analytics. Thus, answering the following questions: i) How effective strategies for data splitting and distribution are for HCBDP? ii) What are the required resources?

Moreover, this work extends classical VC that does not support Directed Acyclic Graphs (DAG) based processing [15] by introducing a model to support this characteristic. Thus, the core of our solution is defined based on an algorithm called HR_Alloc which dispatches data over well-selected machines between Cloud Computing for Hybrid Environment (CCHE) and VDC with available resources to create a single HCBDP infrastructure.

The contributions of this work are summarized as follows:

- The paper proposes a model that extends CCC to HCBDP for performing Big Data analytics on top of CCC and VDC environments;
- The proposed model is evaluated in terms of the performance of real-world workloads on top of CCC and VDC environments. Further, although the design of our experiments does not cover Fog or Edge Computing, explicitly, it is important to mention our model takes into account their characteristics. Thus, this approach can be easily adapted for these environments too;
- The HR_Alloc Algorithm minimizes network latency and data movement because it takes decisions using a data-driven mechanism to provide resource allocation between hybrid infrastructures. As an outcome, the model helps to decrease hardware and computing expenditures due to the advantages of the use of VDC in the hybrid processing paradigm.

The paper is structured as follows. Section II presents the state-of-the-art of big data analytics in hybrid infrastructures. Section III shows the architecture of our solutions and describes the proposed model and the algorithm created to provide resource allocation. Section IV describes the methodology, experiments, and obtained results. The conclusions and future works are outlined in Section V.

II. RELATED WORK

This section presents the shortcomings in Big Data analytics in homogeneous, heterogeneous and hybrid environments such as Multi-Cloud, Hybrid Cloud (HC) and VC environments. HC is a Cloud composed of CCC added of one or more Cloud infrastructures such as Private or Public Cloud or a VCC.

A. BIG DATA IN HETEROGENEOUS ENVIRONMENTS

MapReduce (MR) was the first simplified data processing model for high-distributed clusters applied to batch workloads at Google, in 2004 [21]. MR has been motivating tons of solutions in the most varied fields. Our previous work [16], for instance, proposed *MRA++* strategy to distribute data according to the heterogeneity of the machines to prevent the application's slow down.

The MOON project (***M****apReduce* ***O****n* ***O****pportunistic e****N****vironments*) [22], constituted both homogeneous cluster and VC. This approach was driven by the need to avoid data movement costs to clouds and across them by wide-area networks (WAN). The authors mentioned that one machine with an unavailability rate of 40% needs up to eleven replicas. Thus, the solution applying a LATE algorithm [23]. Besides, data loss due to volatility in machines is addressed by data replication in reliable machines. Also, the approach does not adapt the scheduling to the machines' heterogeneous nature.

BitDew-MapReduce (BitDew-MR) represents other MR implementation for volatile machines [24]. BitDew-MR reduces costs through the bag-of-tasks application with a synchronization schema of barrier-free computation to mitigate the host churn. However, BitDew-MR does not consider geographically distributed environments.

Muhammad *et al.* [25] propose a solution to handle vast volumes of data with high input rates that require low latency. The goal is to solve the unbalanced loads caused by skewed streams on heterogeneous clusters. Similarly, Aten [11] manages data aggregation and data streams within message queues, assuming different algorithms as strategies to partition data flow. It optimizes data communication in geo-distributed and heterogeneous environments. Nevertheless, these implementations do not note data transference between different platforms, such as CCC providers.

The work of Ji and Li [26] evaluates the adoption of algorithms for geo-distributed data analytics. It uses a centralized approach that considers significant bandwidth and leads to poor performance as well as to the privacy problem. The distributed execution is used to move computation between data centers by aggregating intermediate results for further processing.

B. CLOUD AND MULTI-CLOUD BASED SYSTEMS

In CCC scenarios, the task of obtaining application and infrastructure requirements is complicated and sometimes incurs higher costs than expected for the Classical Cloud Server Providers (CCSP). It occurs because an application can use more (overestimated) or less (underestimated) resources. The infrastructure might be modified to support the most varied requirements, such as workload variations, multi-tenant requests, scalability, security, and others.

Jayalath *et al.* [27] introduced G-MR, a Hadoop implementation based on a geo-distributed dataset across multiple data centers that can perform MR jobs across multiple paths (the performance can vary considerably). Moreover, open-source frameworks such as *Hadoop MapReduce*, do not support multi-paths. The G-MR has an algorithm called *Data Transformation Graph* (DTG), which determines an execution path for performing a job sequence for MR.

Big Data processing uses data replication mechanisms between different datacenters and CCSP. This type of communication requires high demand, and the variability of performance can lead to the network bottlenecks between CCC operations [28]. In this scenario, deploy strategies to reduce data transfers is much common. It is possible to cite, for instance, the study of Tudoran *et al.* [29] where there are two methods for modeling complex infrastructures: i) the *analytical models* use low-level details with workloads and are characterized by their ability to predict performance, where the details will determine the best modeling; ii) the *sampling method* is an active approach that does not require any previous knowledge of the infrastructure and the information about network topology.

HyMR [30] is a framework for enabling an autonomic *Cloud burst* for clusters of virtual machines that executes MR jobs over a *Multi-Cloud*. The authors implemented a hybrid infrastructure as a Service (HyIaaS) for Virtual Machine (VM) instance (partitions management) in *Multi-Cloud*. HyIaaS implements an OpenStack extension. This partitioning is transparent to the users since it allows access to all VMs in the same manner, regardless of their physical allocation.

Palanisamy *et al.* [31] argue that users tend to choose resources based on trace files of past executions of MR jobs in the CCC. In fact, CCC solutions can be improved by per-job means and per-customer optimization, allocating small slices of resources and leading to low usage in the CCC scenarios. The work utilized a framework for cost-effective resource management called Cura that is designed to create cluster configurations for the MR job automation. The aim is to optimize resource allocation to reduce the infrastructure costs in the CCC datacenters.

Matteussi *et al.* [32] propose a technique to minimize disk contention effects in shared virtualized systems (e.g. CCC and container-based Clouds) in order to improve applications' performance. The work provided a dynamic resource management strategy to adjust disk I/O utilization rates for MR Applications. Moreover, the authors mention the necessity of fair resource management due to the heterogeneity of machines and workloads.

Rathinaraja *et al.* [33] propose a dynamic ranking-based MapReduce job scheduler called DRMJS. It schedules map and reduces tasks based on the VM's performance ranking.

TABLE 1. Frameworks and techniques for big data analysis.

		Infrastructures							Strategies													Proc	
Area	**Author**	Geo-Distributed	Cloud Infrastructure	Desktop Grid	Cluster	Multi-Cloud	Hybrid Infrastructure	Lambda Architecture	Cost Evaluation	Failure Recovery	Network Overhead	I/O Throughput	Computation Capacity	Storage Evaluation	P/S or M_Queue	Data Replication	Data Movement	Orchestration System	Processing in Memory	Group Strategy	Barrier Free	Batch	Streaming
Cloud/Multi-Cloud	Jayalath *et al.* [27]	x	x			x			x		x											x	
	Tudoran *et al.* [29]	x				x			x		x	x	x									x	
	Loreti *et al.* [30]		x			x							x					x				x	
	Balaji *et al.* [31]		x						x				x					x				x	
	Matteussi *et al.* [32]		x									x		x								x	
	Rathinaraja *et al.* [33]		x								x	x	x				x					x	
	Chen *et al.* [34]	x	x			x		x	x		x	x				x	x					x	
Heterogeneous	Lin *et al.* [22]			x	x							x				x	x					x	
	Souza Jr. *et al.* [11]	x			x						x	x			x								x
	Anjos *et al.* [16]				x								x				x			x		x	
	Lu *et al.* [24]			x						x		x				x	x				x	x	
	Nasir *et al.* [25]				x						x	x				x							x
	Ji *et al.* [26]	x				x						x				x	x					x	
Hybrid	Coyne *et al.* [35]					x				x							x					x	
	Clement *et al.* [36]		x							x							x					x	
	Alexandrov *et al.* [39]				x			x		x	x				x				x	x		x	x
	Pham *et al.* [40]					x		x							x			x	x			x	x
	Rezgui *et al.* [37]		x				x					x	x				x					x	
	This proposal	x					x		x	x			x			x	x	x		x	x	x	

The main goal of this work is minimizing job latency and improve resource utilization. The DRMJS algorithm calculates each virtual machine's performance score based on hardware heterogeneity (CPU, disk I/O and Network I/O). The big challenge of this proposal is to obtain hardware information in real-time from CCSPs because they are hidden to users.

Chen *et al.* [34] introduce a QoS-Aware data placement to minimize communication costs and data transference between geo-distributed data centers. The proposed heuristic considers the traffic flows in the network topology of data center and replica distribution. The data centers are joined as a block-dependence tree (BDT), reducing the construction to a graph partition problem. The proposal formulates a cost function that optimizes the mapper costs through data replication strategies, and as a result, minimizes the data block transfer cost.

C. HYBRID SYSTEMS

Usually, hybrid systems are represented as a mix of public and private CCC [35]. HC represents a set of on-premises, private and public CCCs orchestrated between two platforms. A hybrid system gives greater flexibility and more data deployment options [35]. The data is moved from the private to public CCC when a new VM allocation is required to improve task performance. The data locality and data movement remain a challenge for accelerating iterative MR in HC once iterative applications reuse invariant input data.

Clement *et al.* [36] address iterative MapReduce issues in Hybrid IaaS CCC environments. The authors argue that it is essential to improve the ability to take advantage of the data locality in a HC environment. The strategy aims to extend the original fault-tolerance mechanism of HDFS and deploy data replicas from an on-premise VM in a private CCC to another VM. Also, the off-premise VM is allocated in a public CCC as an external rack in the HDFS..

Rezgui *et al.* [37] implement CloudFinder a VCC, where several Private Clouds would combine their resources to use as a single Cloud. The computational resources are deployed on top of GENI, an NSF-funded Cloud Federation. The owners of Private Clouds donate computing resources in a volunteer fashion. This proposal evaluates execution time, physical hardware (disk, memory, and energy consumption) to perform an optimal workload placement in an available machine, through a weighted average. Finally, heterogeneity evaluation and network latency are open issues.

Lambda Architecture [38] enables building Big Data systems as layers to satisfy properties such as internal code optimization and iterative algorithms that allow them to achieve immutability re-computation, as well as provide low latency without impairing the robustness of the system and other factors. The Apache Flink, previously called Stratosphere, is a data analytic framework that follows the *Lambda Architecture* and enables the extraction, analysis, and integration of heterogeneous datasets [39]. Flink has a flexible pipeline that enables several MR and extended functions like Map, MapPartion, Reduce, Aggregate, Join, and Iterative computing.

All application programming interfaces are translated into an intermediate representation of a compiled program via a cost-based optimizer [5].

Cirus [40] is a framework for *Ubilytics* solutions that provides a type of Big Data analytics applied for IoT scenarios. The deployment supports heterogeneous environments based on brokers (IoT Edge), and the sensors are implemented as a Platform as a Service (PaaS) for IoT real-time applications. The reconfiguration management is implemented by *Roboconf*, which dynamically adjusts the infrastructure.

D. RELATED WORK DISCUSSION

Table 1 summarizes the state-of-the-art in Big Data analytics for homogeneous, heterogeneous, and hybrid environments such as CCC, Multi-Cloud, Desktop Grid systems, and HC. Most studies (88,9%) investigated the batch workloads with the MR framework. It occurs because the MR model represents the platform for a plurality of Big Data technologies and solutions which have been used today, such as the Hadoop MR, Storm, Spark, Flink, and so on. However, only few solutions are close to VC, for instance BitDew-MR [24], Cirus [40], MRA++ [16] and CloudFinder [37].

A sampling assessment indicates that 61% of authors propose solutions to CCC or Multi-Cloud implementations, and only 27% propose geo-distributed approaches. The strategies for data and task distribution in the context of HCBDP are unexplored, being an opportunity for the design of new solutions. Furthermore, few solutions evaluate strategies like computational resources and deployment costs, in this scenario, 55% of the studies are using one or another strategy, but only 11% include both. In such a situation, there is gap in a literature gap regarding Big Data analytics in hybrid infrastructures. Nevertheless, some topics could be better explored, for instance, data distribution models, resource management strategies (CPU, memory, disk I/O, network and energy), data placement, and so on. Therefore, this study explores the hybrid infrastructures to find the alternatives for various free-resource allocation in Big Data systems and, also, to enable Big Data in Fog environments in the future.

III. THE ARCHITECTURE AND MODEL FOR HCBDP

CCSP infrastructures have heterogeneous hardware with varied specifications that require fair adjustments. Thus, an incoherent configuration of Cloud services may lead to overestimate or underestimate resource capacities [32]. In scenarios comprised of several heterogeneous CCC environments, an orchestrator should be used to manage Big Data pipelines for data analytics. Also, it must not be centralized to add interoperability for the data distribution in the network [12].

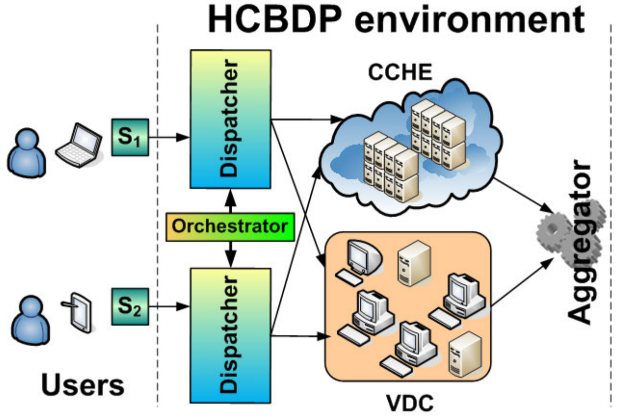


FIGURE 1. Data flow in a HCBDP.

Figure 1 presents the HCBDP and its data flow. The HCBDP environment is comprised of five main components: **Users** and their data sources, the **Dispatcher** receives data and redistributes them to VDC and CCHE infrastructures and also provides infrastructure reallocation with **Orchestrator** coordinating. Finally, an **Aggregator** enables consistent results for the system.

The volunteer machines form groups according to their computational capacity to compose a pool of volunteer resources available to users. The selected volunteer resources comprise a VDC, where the tasks are executed. Once the user creates a VDC, none can occupy the same resource until the owner releases this resource to the pool. Each volunteer resource must have cloud storage (i.e., a service like AWS S3, Google Drive, or Dropbox) where the data is stored. Further, as data is previously replicated from the moment that a volunteer resource setup has started it is necessary a data copy to the attached cloud storage.

The dispatcher handles the task assignments and input data from users or other input sources, for instance, a sensor in a Fog scenario. In this module is deployed the *HR_Alloc* Algorithm, which is detailed in Section III-A. The communication system uses a message queue in a publish/subscriber communication model, *i.e.*, the communication is uncoupled in time and space. It is a centralized data-driven strategies that manages remote data localization, and policies for the splitting and distribution of data, by the needs of each subsystem of the hybrid environment.

The CCHE and VDC environments have their own Big Data processing engine. In such a scenario, the data distribution model was designed to respect the computational capacities of each volunteer machine individually. Thus, a hybrid Big Data engine with two or more distributed file systems must deal with low bandwidth for the data distribution. In the particular case of the VDC environment, the user could assign several sensors closer to computational resources. Following, sensors send data, which will be pre-processed on Big Data processing in these environments. Thus, this is where volunteer machines could compose a Fog environment to improve data processing in the hybrid environment.

Finally, the data processed in each Big Data engine need to be integrated as in a single computation. The aggregator manages all outputs and combines them into a single result using a local aggregation of the keys, for instance, to avoid unnecessary data transfers. The key to this strategy is to provide an input data size for the resources to achieve a better load balance. However, it is not a trivial task, considering that the user might not have sufficient information for decision making regarding data distribution and computational resources.

Thus, an algorithm such as the *HR_Alloc* Algorithm becomes a requirement. The algorithm analyzes the data distribution and resource allocation. To confirm these hypotheses, it was used a simulator called the BIGhybrid simulator [18] with the *HR_Alloc* Algorithm built-in. BIGhybrid is an analysis tool for Big Data in hybrid environments to enable the deploying strategies for distribution and data placement, which supports a controlled environment with consistent evaluations.

TABLE 2. Notation adopted for the model description.

Symbol	Description
$\hat{R}_C$	Vector with all resources offers in a CCC.
r_C	An individual resource CCC allocated.
n	The total possible resources in CCC.
k	The total possible resources in VDC.
$\hat{R}_{\text{VDC}}$	Vector with all resources in a VDC.
R	All available resources in the system.
r_{VDC}	An individual resource in VDC allocated.
S	The selected resources.
τ	The number of simultaneous tasks in each worker.
β	The input data in a particular system.
Φ_C	The wave rate produced in a CCC.
Φ_{CCHE}	The wave rate produced in a CCHE.
Φ_{VDC}	The wave rate produced in a VDC.
$C_{ck\text{-size}}$	Chunk size in CCC.
$\text{CCHE}_{ck\text{-size}}$	Chunk size in CCHE.
$\text{VDC}_{ck\text{-size}}$	Chunk size in VDC.
W_C	Workload in CCC.
W_{CCHE}	Workload in CCHE.
W_{VDC}	Workload in VDC.
S_C	The selected resource in CCC.
S_{CCHE}	The selected resource in CCHE.
S_{VDC}	The selected resource in VDC.

A. THE MODEL FOR DATA AND TASK DISTRIBUTION

The main idea of this model is to evaluate resource availability by establishing task distribution and load balancing strategies for varied workloads as best as possible. The first assumption is that the processing occurs in waves, when the mean of tasks begin and finishes almost the same time, as in the MapReduce execution model. It means that the system fills all the available computational resources for task execution. Thus, the tasks must begin and finish at the same time until the job finishes as in execution waves.

This execution behavior in heterogeneous environments can be achieved through to able several adjustments about execution job and resource allocation algorithm. Table 2 summarizes the notation used throughout this Section, which sets out the model used in this HCBDP proposal.

A CCSP offers resources in the format of *VM* instances. Thus, each *VM* represents a set of heterogeneous resources comprised of CPU cores, memory, and storage. This model should decide how to balance workload between two environments, for instance, a set of VMs of a CCC and VDC machines for creating a HCBDP.

The Φ relations, shown in Equations 1, 2 and 3, represent how many execution rounds (waves) a job can have in a workload when all computational resources are occupied. Φ_C relation (Equation 1) is the ratio of the total workload in CCC (W_C) (equivalent to input data in chunk number) divided by the selected resources in CCC (S_C). Φ_C is a particular case where the job execution occurs only in the CCC without VDC resources. In a batch workload, such as MapReduce, for instance, the input data (called β) is split in chunks in accordance with a chunk size ($C_{ck\text{-size}}$) that determines the workload.

$$\Phi_C = \frac{W_C}{S_C}, \quad \text{where } S_C = \sum_{i=1}^{n} S_{C_i} * \tau \text{ and } W_C = \frac{\beta}{C_{ck\text{-size}}} \tag{1}$$

The job in the HCBDP model has two engines that work in parallel and, therefore, a Φ relation is defined to each environment. Refers to Equation 2 for CCHE and Equation 3 for VDC, W_{CCHE} and W_{VDC} are the workloads in CCHE and VDC respectively.

$$\Phi_{\text{CCHE}} = \frac{\text{CCHE}_{ck\text{-size}}}{C_{ck\text{-size}}} * \frac{W_{\text{CCHE}}}{\sum_{i=1}^{n} S_{\text{CCHE}_i} * \tau} \tag{2}$$

$$\Phi_{\text{VDC}} = \frac{\text{VDC}_{ck\text{-size}}}{C_{ck\text{-size}}} * \frac{W_{\text{VDC}}}{\sum_{k=1}^{k} S_{\text{VDC}_k} * \tau} \tag{3}$$

To achieve a good load balancing between Φ_{CCHE} and Φ_{VDC} relations, these relations must maintain a relationship with the Φ_C relation. Therefore, the HCBDP will achieve the best load balancing if the bi-conditional statement in Equation 4 is satisfied, as it will be demonstrated in Section IV.

$$\Phi_{\text{CCHE}} \leq \Phi_C \iff \Phi_{VDC} \leq \Phi_C \tag{4}$$

B. HR_Alloc ALGORITHM

HR_Alloc Algorithm, in Algorithm 1, implements the data distribution strategy about the dispatcher module to maximize the use of the VDC resources and minimizing the CCHE resource allocation in a HCBDP. This adjustment should not increase the execution time of the job when compared with a CCC implementation. The premise is that some volunteer resources are relatively stable in a hybrid environment, *i.e.*, not all computational resources have intermittent availability. Another assumption is concerning security. The VDC has a strong trust relationship in a hybrid environment.

The data split considers the execution waves in the Big Data engines to achieve this goal. The algorithm defines the resource allocation in CCHE and VDC environments based on the CCC execution. The workload for CCHE and VDC depends on the input data size in the batch and streaming workloads. The users must provide minimal information, such as, total workload size and the vectors with the available resources for CCHE and VDC.

Algorithm 1 HR_Alloc Algorithm

```
   input: data size, $C_{ck\text{-size}}$ , $\mathrm{CCHE}_{ck\text{-size}}$ , $\mathrm{VDC}_{ck\text{-size}}$ ,$\tau$
   Data: $\hat{R}_C$, $\hat{R}_{\mathrm{VDC}}$
1  begin
2      read $\hat{R}_C$
3      while within computational budget do
4          Select a $r_{C_i}$ resource from $\hat{R}_C$
5          if budget $<$ {$r_{C_i}$ cost } then
6              $S_C = \sum_{i=1}^{n} S_{C_i} * \tau$
7              budget = budget $-$\{$r_{\mathrm{CCHE}}$ cost\}
8          next $r_{C_i}$
9      $\Phi_C \leftarrow$ PhiCalc ($S_C$, $C_{ck\text{-size}}$ , $\beta$, $\tau$)
10     while ($\Phi_{\mathrm{CCHE}} > \Phi_C$) $\vee$ ($\Phi_{VDC} \leq \Phi_C$) do
11         {$W_{\mathrm{CCHE}}$, $W_{\mathrm{VDC}}$} $\leftarrow$ Split the input data
12         VDC_Alloc ($W_{\mathrm{VDC}}$, $S_{\mathrm{VDC}}$, $\mathrm{VDC}_{ck\text{-size}}$ , $\Phi_C$)
13         CCHE_Alloc ($W_{\mathrm{CCHE}}$, $S_{\mathrm{CCHE}}$, $HC_{ck\text{-size}}$ , $\Phi_C$)
14     Function VDC_Alloc (Split,$S_{\mathrm{VDC}}$, $\Phi_C$)
15         while  $\Phi_{\mathrm{VDC}} \neq \Phi_C$ do
16             Select a $r_{\mathrm{VDC}}$ resource from $\hat{R}_{\mathrm{VDC}}$
17             $S_{\mathrm{VDC}} = \sum_{i=1}^{k} S_{\mathrm{VDC}_i} * \tau$
18             $\Phi_{\mathrm{VDC}} \leftarrow$ PhiCalc
               ($S_{\mathrm{VDC}}$, $\mathrm{VDC}_{ck\text{-size}}$ , $W_{\mathrm{VDC}}$, $\tau$)
19             $S_{\mathrm{CCHE}} \leftarrow$ remove an equivalent resource
               of $r_{\mathrm{VDC}}$ from $S_C$
20         return $S_{\mathrm{VDC}}$
21     Function CCHE_Alloc (Split,$S_{\mathrm{CCHE}}$, $\Phi_C$)
22         while  $\Phi_{\mathrm{CCHE}} \neq \Phi_C$ do
23             $\Phi_{\mathrm{CCHE}} \leftarrow$ PhiCalc
               ($S_{\mathrm{CCHE}}$, $\mathrm{CCHE}_{ck\text{-size}}$ , $W_{\mathrm{CCHE}}$, $\tau$)
24             $S_{\mathrm{CCHE}}$ $\leftarrow$ adjusts resource from $S_C$
25         return $S_{\mathrm{CCHE}}$
26     Function PhiCalc
       (Resources, Chunk-size, Workload, $\tau$)
27         Calculates $\Phi$ relation in accordance with
           Equation 1 or Equation 2 or Equation 3
28         return
```

Three functions in the *HR_Alloc* Algorithm will select the adequate resources to maintain the Φ relation stable from a determine workload. Φ computation is key for each environment. The smaller number of CCHE resources compared to the CCC represents a saving budget cost, and maximizing the amount of VDC resources means not compromise workload execution time performance. *PhiCalc* function calculates the Φ_C relation in CCC to provide an upper bound in terms of selecting from resources considering the computational budget of the user. The *CCHE_Alloc* function determines the number of volatile machines that are selected in accordance with the workload size to preserve the Φ_{CCHE} relation. *VDC_Alloc* function selects the best VDC resources in the S_{VDC} set to perform the workload that matches with Φ_{VDC}.

The computational budget (line 3) determines the set of resources for CCC. However, only part of these resources are selected effectively (line 22) to compose the resources of CCHE. The effects of the computational budget is analyzed in Section IV in comparison with the hybrid environment cost.

The data in the hybrid environment is split by the processing capacities of resources for each Big Data engine (line 11) and adjusted for achieving the relation of Equation 4. The amount of input data is preserved, but it is redistributed between the environments to provide the best load balance possible.

Since one VDC resource is selected to form the S_{VDC} set, one equivalent resource can be removed from S_C, then the algorithm provides adjustments on the S_{CCHE} set to preserve Φ_{CCHE} relation. Observes that as the S_{VDC} set can provide some machines with lower computational capacities than the S_{CCHE} resource set, the total number of devices on each set can be different, but what is preserved is the Φ relation among them.

Further, in a VDC environment, the data is distributed in accordance with the computational capacities of the machines. Thus, the execution time in the heterogeneous environment is optimized and the data copy is minimized. This algorithm executes before each job execution to determine the optimal data distribution and resource allocations by the workload kind.

IV. EVALUATION

The development of new software for hybrid infrastructures, raises the following questions: (i) How effective strategies for data splitting and distribution are for HCBDP? (ii) What are the required resources? This section presents and evaluates stratagems for the deployment of hybrid environments, particularly the dispatcher module.

A. METHODOLOGY

This section describes the fully methodology of experiments. Thus, the experiments were evaluated through an analysis of the workloads produced in the work of Yanpei *et al.* [41]. This work adopts the use of simulation on top of BIGhybrid simulator to establish a relation close to the following synthetic workloads. These workloads had the outcomes obtained from real-world executions of Big Data applications of companies like Yahoo$^{\mathrm{TM}}$and Facebook$^{\mathrm{TM}}$. The YH and FB clusters have 2,000 and 3,000 machines respectively. However, although the type of jobs at Facebook changes significantly from one year to another, the purpose of this scenario is to cover a real-world environment. The number of experiments performed was around 1,800 tests, each one hosted into a single machine aiming to reduce the execution time required for the evaluations.

Along these lines, every evaluation performed represents discrete-event simulations performed on top of Grid'5000 [42] scenarios in varied sites (Sophia, Nancy, Rennes and Grenoble). Each cluster has 50 hosts with 2 Intel Xeon E5520 processors of 2.27 GHz, with 4 cores, 24 GB of RAM, 119 GB of local disk and 1 Gbps network.

The computational capacity of the processors in the simulated experiments is equivalent to an Intel Xeon E5506-2 Cores, 4M Cache, 2.13 GHz ≈ 5 GFlops and the computational capacity in a VDC environment represents a distributed value between 4 to 6 GFlops, for all of the experiments. This configuration is similar to what was found in Yahoo and Facebook according to the evaluation made by Yanpei [41]. For analytical purposes, the computational consumption is defined by workloads of 64, 32, and 16 MB chunk size. The network, workload and the machines numbers vary in each experiment. The number of *Reduce* tasks is equal to twice the amount of machines. The experiments are conducted with machines with low, medium, and high-scales.

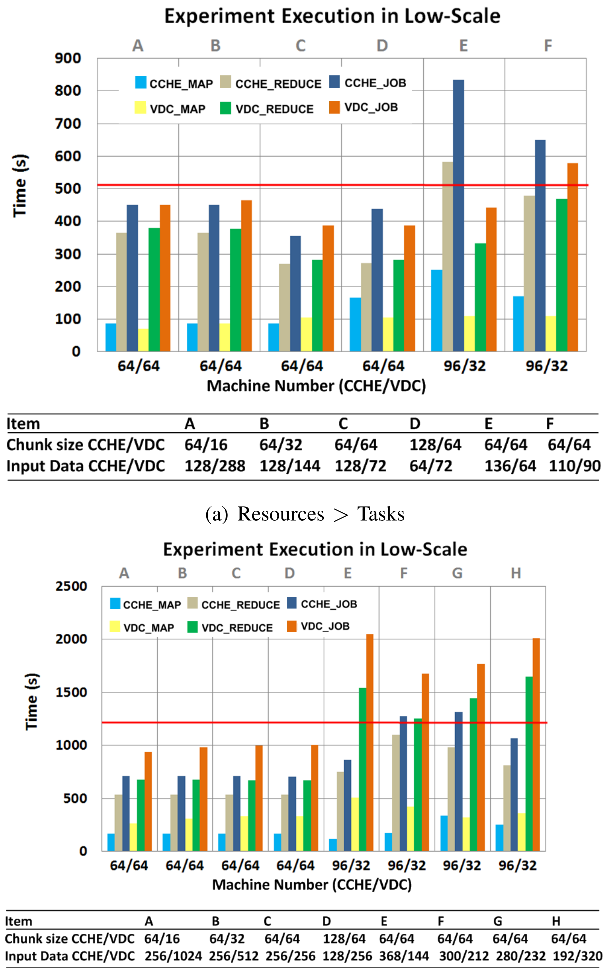
B. EXPERIMENTS WITH LOW AND LARGE-SCALE IN HCBDP

The first experiment is a sequence of two cases in low-scale, with 128 machines in a HCBDP, where the aim is to verify the impact of Φ relation in the data distributing and the amount machines for CCHE and VDC. Figure 2 shows these two cases. The first case, in Figure 2.(a), studies the behavior where the number of resources available for processing is higher than the number of tasks, and the second case, in Figure 2.(b), studies the behavior where the number of tasks is greater than available resources. Each machine processes two tasks per core. The concurrent execution task in the y-axis is measured in seconds, and the number of machines for CCHE and VDC in the x-axis is measured in units. The red line indicates the execution time in a CCC deployment, with 128 machines. Different executions are expresses by **A** to **H** letters.

In the first case, the execution time is equivalent to 503 seconds in Figure 2.(a) with 200 tasks. In the second case, in Figure 2.(b) the red line indicates an execution time equivalent to 1,232 seconds and 512 tasks. The network bandwidth in both experiments is 1 Gbps for CCHE and 10 Mbps for VDC.

From **A** to **D**, there is the best load balance for tasks and data distribution in both environments. In the first case, the number of waves is between 1.5 to 2; thus, if the Φ relation is not observed when the data is distributed as in **E** and **F**, the job execution time increases in comparison with the CCC deployment. The same phenomena occurs in the second case, with the executions from **E** to **H**. If it is only evaluated the chunk size and the input data from Figures 2 (a) and (b) is not possible to reach any conclusion.

Table 3 shows that the Φ_C relation in the CCC environment is equal to 4 for two evaluations. Thus, when there is the same value for Φ_{CCHE} and Φ_{VDC}, the execution achieves the right load balance. The Φ_C relation in comparison with the Φ_{CCHE} and Φ_{VDC} relations demonstrate that the job execution time is lower when the relationship $\Phi_{\text{CCHE}} \leq \Phi_C \iff \Phi_{\text{VDC}} \leq \Phi_C$ is verified. Thus, in this way, machine numbers can be determined to achieve this relationship.



Item	A	B	C	D	E	F
Chunk size CCHE/VDC	64/16	64/32	64/64	128/64	64/64	64/64
Input Data CCHE/VDC	128/288	128/144	128/72	64/72	136/64	110/90

(a) Resources > Tasks

Item	A	B	C	D	E	F	G	H
Chunk size CCHE/VDC	64/16	64/32	64/64	128/64	64/64	64/64	64/64	64/64
Input Data CCHE/VDC	256/1024	256/512	256/256	128/256	368/144	300/212	280/232	192/320

(b) Tasks > Resources

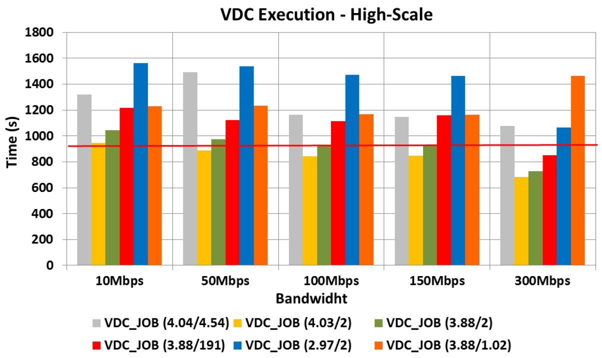
FIGURE 2. Impact of Φ relation in low scale - 128 machines.

TABLE 3. The Φ relationship for data distribution in hybrid environments.

Experiment	Φ_C	Φ_{CCHE}	Φ_{VDC}
A	**4**	**4**	**4**
B	**4**	**4**	**4**
C	**4**	**4**	**4**
D	**4**	**4**	**4**
E	**4**	**3.83**	**4.5**
F	**4**	**3.13**	**6.63**
G	**4**	**2.91**	**7.25**
H	**4**	**2**	**10**

The second experiment represents job executions in a large-scale scenario, with 2,000 machines and a workload of 9,088 chunks, ≈ a half Terabytes of input data. The network bandwidth varies in each execution to determine if the execution behavior follows the previous observations for this scenario. Figure 3 shows 30 different experiments. The objective is to seek the number of computers to achieve the relationship $\Phi_{\text{CCHE}} \leq \Phi_C \iff \Phi_{\text{VDC}} \leq \Phi_C$.

Figure 3.(a) shows the job executions. The red line represents the execution time for a CCC environment equivalent



(a) Simulation in Large-Scale

Relation ($\Phi_{\text{CCHE}}/\Phi_{\text{VDC}}$)	#Chunks (CCHE/VDC)	# Machines (CCHE/VDC)
VDC_JOB (4.04/4.54)	7270/1818	1800/400
VDC_JOB (4.03/2)	7270/1818	1800/909
VDC_JOB (3.88/2)	7000/2088	1800/1044
VDC_JOB (3.88/1.91)	7000/2088	1800/1800
VDC_JOB (3.88/1.02)	7000/2088	1800/2044
VDC_JOB (2.97/2)	5000/4088	1680/2044

(b) Relation workload vs. number of machines

FIGURE 3. **Simulation in large-scale.**

to 901 seconds, with $\Phi_C = 4$. Figure 3.(b) shows the number of machines and workloads for this experiment. The different jobs are represented with colors as indicated in the caption of the figure, where each job shows its Φ relation to CCHE / VDC respectively. The chunk size is 64 MB for all executions. The y-axis measures the parallel tasks in a VDC environment in seconds for each execution and the x-axis measures the bandwidth.

The experiment in large-scale operations shows that the behavior is similar in comparison with low-scale ones. However, some differences can be explained by a large number of machines in the network and the administrative overhead needed to manage the data transference on the Internet, as with low bandwidth. For instance, in the case with 300 Mbps bandwidth, more experiments are observed with lower execution time than CCC, and the Φ relationship maintenance is preserved more easily.

The relation $\Phi_{\text{CCHE}} \leq \Phi_C \iff \Phi_{\text{VDC}} \leq \Phi_C$ is also verified according to the earlier estimate. In some cases, the execution is possible from 10 Mpbs to 300 Mpbs bandwidth, for instance, with VDC_job (4.03/2) where $\Phi_{\text{VDC}} = 2$ and there are 909 machines. Nevertheless, the best performance occurs with 300 Mbps bandwidth. A thorough cost analysis can determine other data distributions where the borderline for the VDC execution may be exceeded without any loss of quality in the solution.

In summary, this scenario demonstrates that a slight variation in workload can lead to behavioral changes in the VDC environment. Nevertheless, the Φ relationship can be considered to be consistent, not only in low-scale but also in large-scale operations, as it was demonstrated in these experiments.

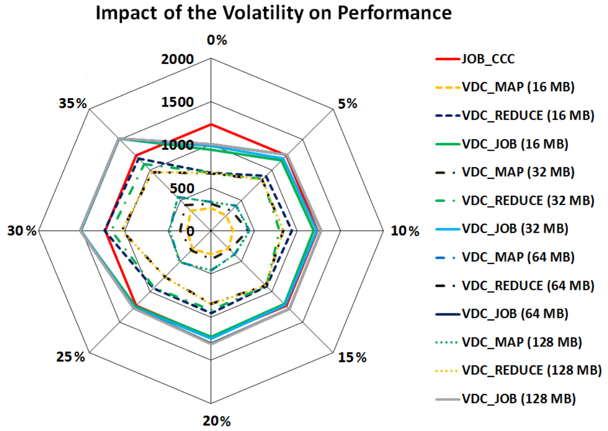


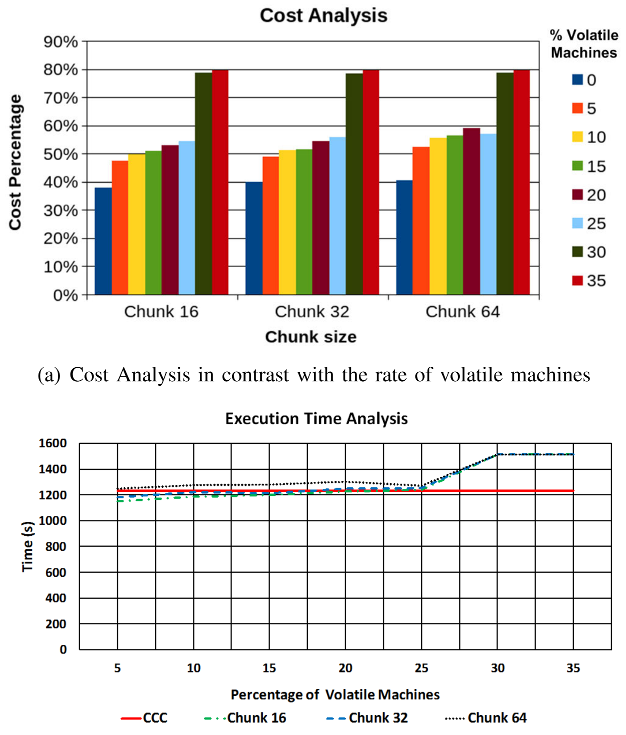
FIGURE 4. **Impact of volatility on performance - 10 Mbps.**

C. VOLATILITY IMPACTS

The third experiment investigates the impact of volatility on performance. The initial assumption is that in a 10 Mbps bandwidth network, the volatility environments are hard for fault recovering due to management overhead. Thus, it is needed to investigate until how much machines can be volatile in the environment to support Big Data applications with acceptable costs.

Figure 4 shows the impact of volatility related to performance. The execution profile is similar to Figure 2.(b) with a network of 10 Mbps bandwidth. The red line in the chart represents the execution time of 1,232 seconds in a CCC job. The external value in the radar graph indicates 0% to 35% of volatile machines in the cluster. The continuous lines indicate the job execution time for MapReduce. Dashed lines indicate Map and Reduce executions. The chunk sizes for this scenario are from 16 MB to 128 MB. The vertical values in the zero-point indicate the execution time in seconds. Data has three replicas for each chunk.

The performance impact is higher in the Reduce phase when there is more data movement and because the scheduler needs first to relaunch Map tasks for another machine with the data replica to the task re-execution in the failure case. On the other hand, the Map phase can benefit from data replication to minimize these initial overhead. When the intermediate data must be copied over the network to execute the reduction function again, there is a low-cost rate between 5% to 25% of unstable machines, in contrast with a high-cost rate from 26% to 35% unstable machines. Therefore, the experiments suggest there is operational flexibility in volatile environments when 5% to 25% of machines have a shutdown without any serious degradation of performance. The volatility from 26% to 35% depending on monetary costs to turns feasible to use in Big Data, but this can be questionable due to imposed administrative overhead. The chart demonstrates that chunk size seems to have a low influence on the results, or at least it is unnoticed.



(a) Cost Analysis in contrast with the rate of volatile machines

(b) Volatile Machines vs Time Execution

FIGURE 5. **Cost analysis vs time execution.**

Indeed, losing more than 1/4 of the machines in VDC could produce a high latency in very slow links on the Internet. The VDC environments can remain relatively stable in some scenarios but do not have a behavior that is easily predictable. In a volatile environment, the machines may have an overhead with a data copy to rebuild replicas, since they might experience long timeout periods. Thus, the fault-tolerance mechanisms (FTM) could achieve false-negatives in the failure detection execution. The FTM was detailed in previous work [18].

D. COST EVALUATION

This evaluation considers a hypothetical traditional computational budget for a CCC operation. Thus, the evaluation compares the volatile behavior with the overhead of data copy to rebuild replicas and analyses the number of volatile machines present in the experiment.

Figure 5 shows this cost analysis of the HCBDP in contrast with the number of volatile machines and execution time. Figure 5.(a) compares the cost with traditional budget for a CCC operation, related to chunk size and the number of volatile machines. The first measure, the dark blue box, represents the cost without volatile machines where there are only stable machines. The x-axis presents the chunk size, and the y-axis measures the cost percentage related to CCC cost. Figure 5.(b) shows the execution time related to this cost analysis. The x-axis shows the percentage of volatile machines and y-axis measures the execution time in seconds.

The cost analysis adds an administrative penalty of 30% for each unstable machine added to the volatile environment. The penalty is related to data replication and overhead with relaunching tasks. The execution time has a similar profile with 5% and 25% of volatile machines and with a cost 60% lower in comparison with CCC. The costs with 64MB chunk size are slightly higher than others due to network latency. On the other hand, it can have lower administrative overhead to the management of fewer tasks, mainly in high-scale environments, as seen in Figure 3.

The use of 30% to 35% of unstable machines in VDC environments represents a cost decrease close to 20% in comparison with a CCC allocation. In contrast, the execution time is 23% higher in comparison with CCC, as Figure 5.(b) demonstrates. Nevertheless, the lower cost with environments composed of between 26% and 35% of volatile machines, as demonstrated in Figure 5.(b), might not be reasonable to some Big Data applications due to an increase 23% execution time in comparison with CCC. Thus, the execution time analysis also indicates that the HCBDP is feasible with up to 25% of volatile machines for Big Data environment.

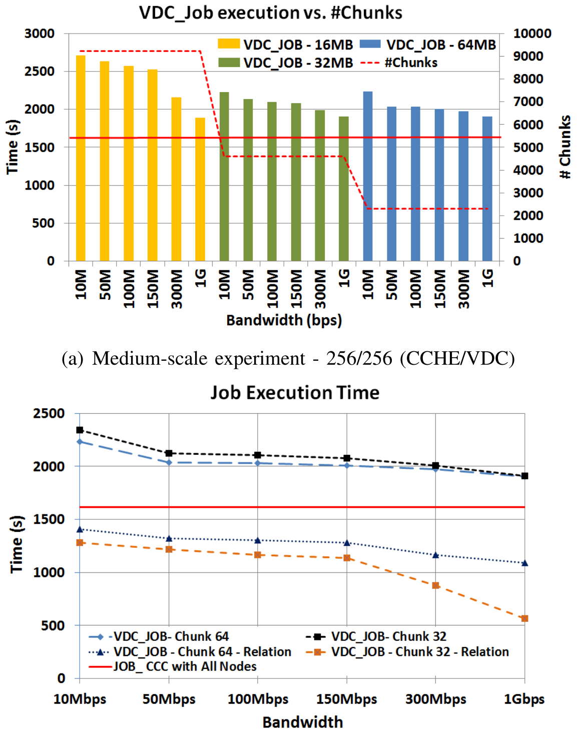
E. BANDWIDTH IMPACT IN DATA DISTRIBUTION VERSUS Φ RELATION

The next experiment is executed in medium-scale. The aim is to analyze the effectiveness of data distribution relations and determine the impact of bandwidth on the whole environment of the HCBDP in comparison with adoption of Φ relation.

The experiment is composed of two charts in Figure 6. These charts represent the analyses about experiments that process 4,608 chunks of 64 MB with 512 machines. The job execution time in CCC is 1,618 seconds, assented in the red line. The bandwidth ranges from 10 Mbps, 50 Mbps, 100 Mbps, 150 Mbps, 300 Mbps and 1 Gbps with latencies captured from the real-world environment. It should be noted that at this stage, a 1 Gbps bandwidth for volatile machines is possible with Optic Fiber links or in the 5G networks in Fog environments.

In the first experiment, in Figure 6.(a), the number of machines is divided by half, 256 machines for CCHE, and 256 machines for VDC. The total number of tasks is also equally divided into half 2,304 for the CCHE, and VDC. The blue, green, and yellow colors represent the job executions time in VDC with a chunk size of 64 MB, 32 MB, and 16 MB respectively. The other y-axis, on the right, measures the VDC workload in the chunk number for each execution with a different size for each executes sequence. The workload consists of 2,304, 4,608 and 9,216 chunks and has a chunk size of 64 MB, 32 MB, and 16 MB respectively. The x-axis measures the bandwidth.

The goal is to check if only a data division with the increased bandwidth will be sufficient to achieve load balance for the data split without using the *HR_Alloc* Algorithm, i.e. without the Φ relation. As can be seen, all the executions overcome the minimum expected time (1,618 seconds - in the red line). The performance is worse than an implementation in



(a) Medium-scale experiment - 256/256 (CCHE/VDC)

(b) Data Distribution with the Φ Relation analysis

FIGURE 6. Bandwidth impact in data distribution vs. Φ relation.

CCC with all the hosts. Thus, dividing the data and machines in half and increase the bandwidth is not a good strategy for distribution data in a HCBDP, regardless of bandwidth. Moreover, the experiment demonstrates that splitting the input into chunk sizes lower than 64 MB, such as 16 MB, can result in a poor performance in this scenario.

The sharp increase in the job execution time, rather than the reduction, is based on the false assumption that the division of machines and data by half (distributed in CCHE and VDC) could represent a reduction of half the time needed in hybrid environments. In fact, this likelihood is incorrect because, in a HCBDP, factors such as heterogeneity and volatility must also be taken into account. However, this scenario can be evaluated in another manner for an understanding of what in fact occurs with chunk sizes in relation to variations in bandwidth.

The subsequent analysis, in Figure 6.(b), is similar to the experiment of Figure 6.(a), however, with 64 MB and 32 MB chunk sizes and where is evaluated the use of *HR_Alloc* Algorithm. The purpose of this experiment is to consolidate the previous observations and to demonstrate that the use of the Φ relation is a feasible strategy for the setup of machines and data distribution in HCBDPs. The CCC job has a runtime of 1,618 seconds assented in red line. The x-axis measures the bandwidth. The application processes a workload of 2304 and 4608 chunks with 64 MB and 32 MB chunk sizes, respectively. The y-axis measures the concurrent tasks in a VDC environment in seconds for each execution with different chunk size.

Figure 6.(b) shows the execution time for a job in an HCBDP with two distinct analyses, above the red line there are job executions without the use of the *HR_Alloc* Algorithm and below it the job executions with its use. In the first analysis, above the red line, the impact is linear and has a slope close to 10% in the execution time, while the bandwidth increases from 10 Mbps to 150 Mbps (15 times). The range from 10 Mbps to 300 Mbps (an increase of 30 times in bandwidth) has similar behavior with a reduction from 15% to 30% for a chunk size of 64 and 32 MB, respectively. Unfortunately, this is not sufficient to promote a load balance between the CCHE and VDC environments.

The second case, below the red line, shows that the Φ relation has a beneficial effect by reducing the data transfers between machines from 39.1% to 57.14% in the worst and best-case scenarios, respectively. These gains are ≈ 2 to 3 times higher, in comparison with the bandwidth results, due to the effect of Φ relation use than without it. This reduction in data transfers added to the impact of the bandwidth produces the result needed to provide a proper load balancing and make the use of an HCBDP feasible. These results provide evidence that the VDC executes a larger number of local tasks with the Φ relation than without it and, thus, reduces the data transfers in the whole system.

F. DISCUSSION

This section examines the deployment of the HCBDP for Big Data analytics through synthetic applications from Yahoo Cluster with the use of a dispatcher module. The scenario represents Big Data applications in geographically distributed environments.

The studies of Jayalath *et al.* [27] and Tudoran *et al.* [29] are scenarios of CCC-to-CCC deployments; both provide support for data transfers in decreasing execution time for MapReduce jobs. The former focuses on the cost of performance while the latter observes I/O throughput and the computational environment capacity. In contrast, our work proposes a solution which can be used in a hybrid approach for real-time and batch applications. Moreover, our study implements mechanisms to avoid unnecessary data movement.

A recommendation set for VDC deployment can help in the setup of HCBDP used in geographically distributed environments. The evaluations, in subsections IV-C and IV-D, indicate that the relations between volatility impacts and cost analysis can determine the VDC resource level accepted in HCBDP use. The Φ relation establishes a method for determining resource allocation to CCHE and VDC in HCBDP. Also, it can produce load balancing when the data distribution is related to the number of CCHE and VDC resources.

The results suggest that HCBDP confers an operational continuity in an environment with up to 25% of unstable machines in the best-case scenario without a loss of performance and low-cost with three replicas for each chunk. Also, depending of cost evaluations can achieve 35% with volatile machines in worst-case. In contrast, the work of Lin *et al.* [22] argues that a machine with an unavailability rate of 40%, must have eleven replicas to achieve an availability rate of 99.99% for a single data block in HDFS. Therefore, our solution can still preserve storage resources.

The Φ_{CCHE} and Φ_{VDC} parameters used to find CCHE and VDC resources, establish a suitable number of machines to achieve an acceptable performance and good approximation. These settings also help inexperienced users to locate the number of CCHE and VDC machines without the need for previous knowledge of the CCSP infrastructure, which can be considered to be one of the benefits of this study.

Several authors like Tudoran *et al.* [29], Balaji *et al.* [31] and Clement *et al.* [36] argue that the users tend to choose resources based on their workload peak, and the systems must find the optimal chunk placement that correspond to the user needs. In contrast, the Φ relation between CCHE and VDC can help users to find the resources adapted to their workloads. In addition, the recommendation of a chunk size in a communication channel can help prevent excessive data movements in Big Data applications in hybrid infrastructures.

The correlation among the workload, number of machines, and load balancing in the Φ relation are behind the performance improvement to provide the best data load balancing possible and reducing the data transfers between nodes from 39.1% to 57.14% in the worst and best-case scenarios, respectively. These values are compatible with the work of Tudoran *et al.* [29], which achieved a reduction of 50% with a relative error of 10% to 15%.

V. CONCLUSION

Cloud has changed the way applications are developed and ported in geographically distributed infrastructures. HCBDP can offer services providing new features as well as being a suitable scenario for building Big Data applications and their range of components. However, it can be hard to maintain these systems if the users do not manage their application resources appropriately, which can lead to issues of cost-effectiveness.

This work provides Big Data analytics in a hybrid infrastructure called HCBDP. In contrast with other frameworks, this environment uses CCHE and VDC as its basic infrastructure for Big Data processing. The deployment approach uses a geographically distributed system;

The evaluations found behavioral patterns which enable their deployment in HCBDP (in low, medium and large-scale) and established the relationship among workload, number of machines, and load balancing through the relationship Φ between CCHE and VDC.

The *HR_Alloc* Algorithm for HCBDP establishes an operational continuity in an environment with up to 25% of unstable machines in the best-case scenario without a loss of performance, maintaining three replicas data and with a cost 60% lower than in comparison with CCC. The Φ_{HC} and Φ_{VC} parameters designed to find CCHE and VDC resources established that there were a number of suitable machines that could achieve an acceptable performance. Thus, the Φ relationship between CCHE and VDC minimizes network latency and data movement. Also, it can help users to find quickly resources adapted to their workloads.

Thus, the proposed model demonstrated to be viable for the decrease in computing expenditures due to the use of VDC in the hybrid processing paradigm.

Furthermore, the recommendation of chunk sizes in the communication channel can mitigate an excessive data movement in Big Data applications within hybrid infrastructures. The relationship between the workload, number of machines, and load balancing can be regarded as a significant contribution to improve data load balancing and reducing data transfers between machines from 39.1% to 57.14% in the worst and best case scenario, respectively. These values are compatible with those found in the literature.

In the future, this hybrid environment can enable the use of Fog in Big Data applications. It also reduces data transfers between IoT devices and CCC, with the use of VDC for pre-processing of Big Data in the Fog environment composed of these volunteer machines.

Other future works are needed to build a platform in a real-world environment. In particular, one possible strategy to the dispatcher module that could give priority to execution and thus avoiding the delay in the task flow. Moreover, the storage mechanism could be evaluated to include I/O interference and evaluation of container techniques, or to examine the use of accelerators (such as virtual GPGPUs as rCUDA or GVirtuS) and shared FPGAs added to HCBDP.

ACKNOWLEDGMENT

The experiments in this paper were carried out using the Grid’5000 experimental testbed, and conducted under the INRIA ALADDIN development scheme with the support of CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] D. Reinsel, J. Gantz, and J. Rydning, *The Digitalization The World: From Edge to Core*, vol. 1. Framingham, MA, USA: Seagate, 2018. [Online]. Available: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf

[2] M. D. Assunção, R. N. Calheiros, S. Bianchi, M. A. S. Netto, and R. Buyya, “Big data computing and clouds: Trends and future directions,” *J. Parallel Distrib. Comput.*, vols. 79–80, pp. 3–15, May 2015, doi: 10.1016/j.jpdc.2014.08.003.

[3] T. White, *Hadoop—The Definitive Guide*, vol. 1. Newton, MA, USA: O’Reilly Media, 2012.

[4] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica, “Apache spark: A unified engine for big data processing,” *Commun. ACM*, vol. 59, no. 11, pp. 56–65, Oct. 2016, doi: 10.1145/2934664.

[5] V. Markl, “Mosaics in big data: Stratosphere, apache Flink, and beyond,” *Proc. 12th ACM Int. Conf. Distrib. Event-Based Syst.*, New York, USA, 2018, pp. 7–13, doi: 10.1145/3210284.3214344.

[6] A. Toshniwal, J. Donham, N. Bhagat, S. Mittal, D. Ryaboy, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, and M. Fu, ''Storm Twitter,'' in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, p. 147, doi: 10.1145/2588555.2595641.

[7] L. Badger, T. Grance, R. Patt-Corner, and J. Voas, ''Cloud computing synopsis and recommendations,'' NIST, Gaithersburg, MD, USA, Tech. Rep. NIST 800-146, May 2012, doi: 10.6028/NIST.SP.800-146.

[8] R. Buyya, J. Broberg, and A. Goscinski, *Cloud Computing*, vol. 1. Hoboken, NJ, USA: Wiley, 2011.

[9] H. IAT, I. Yaqoob, A. NB, S. Mokhtar, A. Gani, and K. S. Ullah, ''The rise of big data on cloud computing,'' *Inf. Syst.*, vol. 15, pp. 98–115, Jan. 2015, doi: 10.1016/j.is.2014.07.006.

[10] E. G. Renart, J. Diaz-Montes, and M. Parashar, ''Data-driven stream processing at the edge,'' in *Proc. IEEE 1st Int. Conf. Fog Edge Comput. (ICFEC)*, May 2017, p. 31, doi: 10.1109/ICFEC.2017.18.

[11] P. R. R. de Souza, K. J. Matteussi, J. C. S. dos Anjos, J. D. D. dos Santos, C. F. R. Geyer, and A. da Silva Veith, ''Aten: A dispatcher for big data applications in heterogeneous systems,'' in *Proc. Int. Conf. High Perform. Comput. Simul. (HPCS)*, Jul. 2018, p. 585, doi: 10.1109/HPCS.2018.00098.

[12] J. C. S. Anjos, K. J. Matteussi, P. R. R. De Souza, A. da Silva Veith, G. Fedak, J. L. V. Barbosa, and C. R. Geyer, ''Enabling strategies for big data analytics in hybrid infrastructures,'' in *Proc. Int. Conf. High Perform. Comput. Simul. (HPCS)*, Jul. 2018, p. 869, doi: 10.1109/HPCS.2018.00140.

[13] J. Darrous, T. Lambert, and S. Ibrahim, ''On the importance of container image placement for service provisioning in the edge,'' in *Proc. Int. Conf. Comput. Commun. Netw.*, Valencia, Spain, 2019, pp. 1–9.

[14] M. Satyanarayanan, ''The emergence of edge computing,'' *Computer*, vol. 50, no. 1, pp. 30–39, Jan. 2017, doi: 10.1109/MC.2017.9.

[15] D. P. Anderson, ''BOINC: A platform for volunteer computing,'' 2019, *arXiv:1903.01699*. [Online]. Available: http://arxiv.org/abs/1903.01699

[16] A. JC, I. Carrera, W. Kolberg, T. AL, A. LB, and G. CR, ''MRA++: Scheduling and data placement on mapReduce for heterogeneous environments,'' *J. Future Gener. Comput. Syst.* vol. 42, pp. 22–35, Jan. 2015, doi: 10.1016/j.future.2014.09.001.

[17] J. C. S. dos Anjos, M. D. Assuncao, J. Bez, C. Geyer, E. P. de Freitas, A. Carissimi, J. P. C. L. Costa, G. Fedak, F. Freitag, V. Markl, P. Fergus, and R. Pereira, ''SMART: An application framework for real time big data analysis on heterogeneous cloud environments,'' in *Proc. IEEE Int. Conf. Comput. Inf. Technol., Ubiquitous Comput. Commun.*, Oct. 2015, p. 199, doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.29.

[18] J. C. S. Anjos, G. Fedak, and C. F. R. Geyer, ''BIGhybrid: A simulator for MapReduce applications in hybrid distributed infrastructures validated with the Grid5000 experimental platform,'' *Concurrency Comput., Pract. Exper.*, vol. 28, no. 8, pp. 2416–2439, Jun. 2016, doi: 10.1002/cpe.3665.

[19] T. M. Mengistu and D. Che, ''Survey and taxonomy of volunteer computing,'' *ACM Comput. Surveys*, vol. 52, no. 3, pp. 1–35, Jul. 2019, doi: 10.1145/3320073.

[20] M. Iorga, L. Feldman, R. Barton, M. MJ, N. Goren, and C. Mahmoudi, ''Fog computing conceptual model,'' Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. NIST 500-325, Mar. 2018, doi: 10.6028/NIST.SP.500-325.

[21] J. Dean and S. Ghemawat, ''MapReduce: A flexible data processing tool,'' *Commun. ACM*, vol. 53, no. 1, pp. 72–77, Jan. 2010, doi: 10.1145/1629175.1629198.

[22] H. Lin, X. Ma, J. Archuleta, F. Wc, M. Gardner, and Z. Zhang, ''MOON: MapReduce on opportunistic eNvironments,'' in *Proc. 19th ACM Int. Symp. High Perform. Distrib. Comput.*, New York, NY, USA, 2010, pp. 95–106, doi: 10.1145/1851476.1851489.

[23] M. Zaharia, A. Konwinski, J. AD, R. Katz, and I. Stoica, ''Improving mapReduce performance in heterogeneous environments,'' in *Proc. OSDI*, Dec. 2008, pp. 29–42, doi: 10.5555/1855741.1855744.

[24] L. Lu, H. Jin, X. Shi, and G. Fedak, ''Assessing MapReduce for Internet computing: A comparison of Hadoop and BitDew-MapReduce,'' in *Proc. ACM/IEEE 13th Int. Conf. Grid Comput.*, Sep. 2012, pp. 76–84, doi: 10.1109/Grid.2012.31.

[25] M. Anis Uddin Nasir, H. Horii, M. Serafini, N. Kourtellis, R. Raymond, S. Girdzijauskas, and T. Osogami, ''Load balancing for skewed streams on heterogeneous cluster,'' 2017, *arXiv:1705.09073*. [Online]. Available: http://arxiv.org/abs/1705.09073

[26] S. Ji and B. Li, ''Wide area analytics for geographically distributed datacenters,'' *Tsinghua Sci. Technol.*, vol. 21, no. 2, pp. 125–135, Apr. 2016, doi: 10.1109/TST.2016.7442496.

[27] C. Jayalath, J. Stephen, and P. Eugster, ''From the cloud to the atmosphere: Running MapReduce across data centers,'' *IEEE Trans. Comput.*, vol. 63, no. 1, pp. 74–87, Jan. 2014, doi: 10.1109/TC.2013.121.

[28] N. Grozev and R. Buyya, ''Performance modelling and simulation of three-tier applications in cloud and multi-cloud environments,'' *Comput. J.*, vol. 58, no. 1, pp. 1–22, Jan. 2015, doi: 10.1093/comjnl/bxt107.

[29] R. Tudoran, A. Costan, R. Wang, L. Bouge, and G. Antoniu, ''Bridging data in the clouds: An environment-aware system for geographically distributed data transfers,'' in *Proc. 14th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, May 2014, pp. 1–8, doi: 10.1109/CCGrid.2014.86.

[30] D. Loreti and A. Ciampolini, ''A hybrid cloud infrastructure for big data applications,'' in *Proc. IEEE 17th Int. Conf. High Perform. Comput. Commun.*, Aug. 2015, p. 1713, doi: 10.1109/HPCC-CSS-ICESS.2015.140.

[31] B. Palanisamy, A. Singh, and L. Liu, ''Cost-effective resource provisioning for MapReduce in a cloud,'' *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1265–1279, May 2015, doi: 10.1109/TPDS.2014.2320498.

[32] K. J. Matteussi, C. F. R. Geyer, M. G. Xavier, and C. A. F. De Rose, ''Understanding and minimizing disk contention effects for data-intensive processing in virtualized systems,'' in *Proc. Int. Conf. High Perform. Comput. Simul. (HPCS)*, Jul. 2018, pp. 901–908, doi: 10.1109/HPCS.2018.00144.

[33] J. Rathinaraja, A. VS, and A. Pau, ''Dynamic ranking-based MapReduce job scheduler to exploit heterogeneous performance in a virtualized environment,'' *J. Supercomput.*, vol. 1, pp. 1–28, Aug. 2019, doi: 10.1007/s11227-019-02960-0.

[34] W. Chen, B. Liu, I. Paik, Z. Li, and Z. Zheng, ''QoS-aware data placement for MapReduce applications in geo-distributed data centers,'' *IEEE Trans. Eng. Manag.*, early access, Feb. 21, 2020, doi: 10.1109/TEM.2020.2971717.

[35] L. Coyne, J. Dain, P. Gilmer, P. Guaitani, I. Hancock, A. Maille, T. Pearson, B. Sherman, and C. Vollmar, *IBM Private, Public, Hybrid Cloud Storage Solutions*, vol. 1. Poughkeepsie, NY, USA: IBM Red Books, Apr. 2018.

[36] F. J. Clemente-Castello, B. Nicolae, K. Katrinis, M. Mustafa Rafique, R. Mayo, J. C. Fernandez, and D. Loreti, ''Enabling big data analytics in the hybrid cloud using iterative MapReduce,'' in *Proc. IEEE/ACM 8th Int. Conf. Utility Cloud Comput. (UCC)*, Dec. 2015, pp. 290–299, doi: 10.1109/UCC.2015.47.

[37] A. Rezgui, N. Davis, Z. Malik, B. Medjahed, and H. S. Soliman, ''CloudFinder: A system for processing big data workloads on volunteered federated clouds,'' *IEEE Trans. Big Data*, vol. 6, no. 2, pp. 347–358, Jun. 2020, doi: 10.1109/TBDATA.2017.2703830.

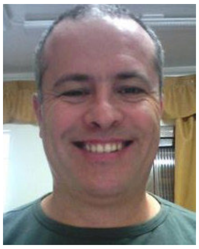
[38] C. Wu, R. Buyya, and K. Ramamohanarao, ''Big data analytics = machine learning+cloud computing,'' in *Big Data: Principles Paradigms*, R. Buyya, R. Calheiros, and A. Dastjerdi, Eds. San Mateo, CA, USA: Morgan Kaufmann, 2016, pp. 3–38, doi: 10.1016/B978-0-12-805394-2.00001-5.

[39] A. Alexandrov, ''The stratosphere platform for big data analytics,'' *Int. J. Very Large Data Bases*, vol. 23, no. 6, pp. 639–664, Dec. 2014, doi: 10.1007/s00778-014-0357-y.

[40] L. M. Pham, A. El-Rheddane, D. Donsez, and N. de Palma, ''CIRUS: An elastic cloud-based framework for ubilytics,'' *Ann. Telecommun.*, vol. 71, nos. 3–4, pp. 133–140, Apr. 2016, doi: 10.1007/s12243-015-0489-0.

[41] Y. Chen, A. Ganapathi, R. Griffith, and R. Katz, ''The case for evaluating MapReduce performance using workload suites,'' in *Proc. IEEE 19th Annu. Int. Symp. Model., Anal., Simul. Comput. Telecommun. Syst.*, Jul. 2011, 390–399 doi: 10.1109/MASCOTS.2011.12.

[42] INRIA. (Mar. 2020). *Grid5000*. [Online]. Available: https://www.grid5000.fr

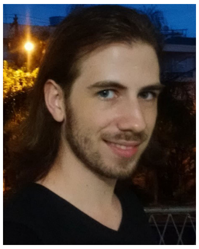


JULIO C. S. DOS ANJOS received the bachelor's degree in electrical engineering with an emphasis on electronic from Pontifical Catholic University of Rio Grande do Sul (PUC/RS), in 1991, and the Ph.D. degree in computer science from the Federal University of Rio Grande do Sul (UFRGS/RS), in 2017, and the master's degree in computer science from the Federal University of Rio Grande do Sul, in 2012. He has been holding a postdoctoral position in computer science with UFRGS since 2017. He is a Researcher in the areas of big data, MapReduce and streaming processing, distributed and heterogeneous systems, hybrid infrastructures, and big data analytics with machine learning and deep learning.

KASSIANO J. MATTEUSSI (Member, IEEE) received the bachelor's degree in computer science from the Regional Community University of Chapecó - Unochapecó, and the master's degree in computer science from the Pontifical Catholic University of Rio Grande do Sul - PUCRS, in 2015. He is currently pursuing the Ph.D. degree in computer science with the Federal University of Rio Grande do Sul - UFRGS. He is certified in software engineering. His areas of research interests include big data, cloud computing, virtualization, distributed systems, high-performance computing, blockchain, as well as FOG and EDGE computing.

PAULO R. R. DE SOUZA, JR., received the bachelor's degree in computer science from the University of Passo Fundo - UPF, and the master's degree in computer science from the Federal University of Rio Grande do Sul - UFRGS, in 2018. He is currently pursuing the Ph.D. degree in computer science with the Univ Rennes, Inria, CNRS. His areas of interests include big data, stream processing systems, distributed and parallel programming, load balance, and job scheduling.

GABRIEL J. A. GRABHER is currently pursuing the bachelor's degree in computer science with the Federal University of Rio Grande do Sul - UFRGS. His areas of interests include big data, cloud computing, and distributed systems.

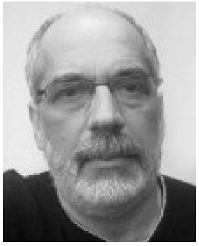


GUILHERME A. BORGES received the bachelor's degree in technology to Internet systems from the Federal Institute Sul-Rio-Grandense - IFSUL, and the master's degree in computer science from the Federal University of Rio Grande do Sul - UFRGS, in 2015. His areas of interests include autonomic computing, distributed systems, self-adaptive systems, smart cities, mobile computing, ubiquitous computing, and edge computing.

JORGE L. V. BARBOSA received the M.Sc. and Ph.D. degrees in computer science from the Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil, in 1996 and 2020, respectively. He conducted his postdoctoral research projects first at Sungkyunkwan University (SKKU), Suwon, South Korea, in 2016, and then at the University of California Irvine (UCI), Irvine, CA, USA, in 2020. He is currently a Full Professor of the Applied Computing Graduate Program (PPGCA) with the University of Vale do Rio dos Sinos (UNISINOS), São Leopoldo, Brazil, also the Head of the Mobile Computing Lab (Mobilab), UNISINOS, and a Researcher of Productivity at the Brazilian Council for Scientific and Technological Development (CNPq), Brazil. He is also a researcher in the areas of ubiquitous computing, context histories, context prediction using context histories mainly through similarity and pattern analysis. His main areas of application are health (u-health), accessibility (u-accessibility), learning (u-learning), and agriculture (u-agriculture).

GABRIEL V. GONZÁLEZ received the master's degree in intelligent systems from the University of Salamanca, in 2012, the master's degree in Internet Security, in 2014, the master's degree in information systems management, in 2015, and the Ph.D. degree from the Department of Computer Science and Automation, University of Salamanca. He is a Computer Engineer at the Pontifical University of Salamanca in 2011. He is currently a Research Professor with the Department of Informatics. Throughout his training, he has followed a well-defined line of research, focused on the application of multi-agent systems to ambient intelligence environments, with special attention to the definition of intelligent architectures and the fusion of information.

VALDERI R. Q. LEITHARDT (Member, IEEE) received the Ph.D. degree in computer science from the Federal University of Rio Grande do Sul UFRGS/RS, in 2015. He is currently an Adjunct Professor with the Polytechnic Institute of Portalegre and a Researcher at the VALORIZA Research Group, School of Technology and Management. He is also a collaborating researcher at COPELABS, Universidade Lusófona de Lisboa, Portugal. He collaborates as a Researcher with the Department of Informatics Engineering, University of Coimbra, and also with the Telecommunications Institute of Portugal - IT Branch Covilhã. He maintains cooperation with other research groups in Brazil, France, Belgium, Morocco, Italy, and Spain. His main line of research is in distributed systems with a focus on data privacy, communication protocols, and programming.



CLAUDIO F. R. GEYER received the Ph.D. degree in informatics from the Université de Grenoble I, in 1991, and the M.Sc. degree in computer science from UFRGS/RS, in 1986. He was a Mechanics Engineer at UFRGS/RS in 1978, where he is also a Tenured Professor in computer science. His areas of interests include pervasive and ubiquitous computing, grid and volunteer computing, scheduling and data-intensive computing, and big data.