A Front-end, Hadoop-based Data Management Service for Efficient Federated Clouds

George Kousiouris[1], George Vafiadis[1] and Theodora Varvarigou[1]

***Abstract*— In the recent years, cloud computing has emerged as the new IT paradigm that promises elastic resources on a pay-per-use basis. The challenges of cloud computing are focused around massive data storage and efficient large scale distributed computation. Hadoop, a community driven Apache project has provided an efficient and cost effective platform for large scale computation using the map-reduce methodology, pioneered by Google. In this paper, the design of a Hadoop-based data management system as the front-end service for Cloud data management is investigated. This framework is enriched with RESTful APIs in front of Hadoop and a series of components that aim to extend Hadoop's functionality beyond its well known back-end, heavy data processing scope. These components are used to enrich security, logging and data analysis features and also data access compatibility between different but interconnected Cloud providers (federated Clouds). Hadoop capabilities are also extended in a quest for intelligent decision making regarding the choice of the fittest services for federation in a federated cloud scenario, in addition to legally compliant behaviour regarding the geographical location of data storage.**

***Index Terms*—Apache Hadoop, Cloud Computing, Data Management, Web Services**

1 INTRODUCTION

The emergence of a variety of Infrastructure-as-a-Service providers (such as [1],[2] and [3] among many) has made it possible for companies or individuals to outsource their IT infrastructures with higher flexibility and dynamic reservation of resources according to their expected usage.

One of the features offered by these providers is Storage as a Service, like the Amazon S3 offering. However, storage systems and data management are still considered as very important challenges in the years to come ([14], [15]), since current implementations show significant constraints (for example with regard to federation capabilities and data lock-in).

Furthermore, existing back-end solutions like Apache Hadoop ([4]) have helped forward the notion of parallel processing and distributed infrastructures through an open source contribution with high levels of performance, fault tolerance and ease of management. However their usage has been limited in internal infrastructures, for specific data-intensive processing tasks, mainly through the MapReduce framework.

The major aim of this paper is to merge these fields through an innovative mechanism for a distributed and flexible Data Management Service (OPTIMIS Distributed FileSystem-ODFS), that will be able to offer Storage as a Service in a more transparent and flexible way than before. ODFS is heavily based on Apache Hadoop (and its underlying Distributed File System-HDFS) and extends its functionalities from a back-end high performance parallel processing framework to a front-end, full-scale offering of Data as a Service. In order to do so, a number of functionalities have been implemented on top of Hadoop:

-Addition of RESTful interfaces in order to expose critical functionalities as services

-Extension of HDFS framework with regard to security, location management, data logging

-Ability to federate the Data Management Service to more than one IaaS providers, avoiding interoperability issues and data lock-in

-Ranking mechanisms for selection of services to federate from a data activity perspective

-Efficient management of data with regard to different scenarios, policies and infrastructure conditions

The remainder of the paper is structured as follows. Section 2 presents related work in the field of cloud data management and hadoop implementations, while Section 3 describes from a high level point of view the ODFS solution. Section 4 presents the key functionalities of added value provided by ODFS while Section 5 presents the status of the implementation at the time and preliminary results on the algorithms used. Finally Section 6 concludes the paper.

2 RELATED WORK

A study on distributed storage systems and architectures used in Clouds can be found in [5]. A benchmarking approach to identify strengths and weaknesses in different cloud-based data management implementations appears in [6]. This approach compares different approaches mainly in distributed databases (most of them are however based on HDFS) in a variety of read/write operations. An incorporation of the HDFS in current Grid infrastructures inside CERN is presented in [7]. In this work, the authors describe why HDFS can meet the needs of a specific experiment in the LHC in addition to a performance analysis of the framework.

CSAL ([8]) introduces an abstraction layer in order to minimize application dependency from storage implementations and data lock-in. In this effort it introduces

[1]G. Kousiouris, G. Vafiadis and Theodora Varvarigou are with the National Technical University of Athens 9, Heroon Polytechniou Str 15773 Athens, Greece, Tel.+302107722546;e-mail:gkousiou@mail.ntua.gr,gvaf@iccs.gr, dora@telecom.ntua.gr.

common namespaces and metadata handling, after implementing the necessary interfaces towards standard Cloud offerings like Amazon S3 and Microsoft Azure. In our case, we minimize the lock-in due to the fact that the ODFS appears as a standard, local directory to the end user, regardless of where the data may be stored. This aids in either taking the data directly from the user location or adding federated Cloud providers on the fly through the storage VM concept that is described in Section 3.

Smartbox ([9]) is designed mainly for mobile devices, offering a traditional hierarchical namespace for data access and an attribute based namespace for semantic queries. It supports a Write-Once-Read-Many consistency scheme. BlobSeer ([10]) introduces a promising framework with the main intention to improve the efficiency and throughput of storage backends under heavy concurrency conditions. Its main focus is on the MapReduce framework, through the usage of versioning.

Nectar ([11]) is focused primarily on the intelligent interchange between data and computation and their relationships. The main focus of the latter is that data that are not been used frequently by other computational tasks can be replaced by the computation that produced them, in order to save storage space. On the other hand, computations that have been made recently can be cached and therefore save the time to reproduce them, if they are needed by many tasks. They are based on functional operators of LINQ, an extension to manipulate .NET objects. The need to have programs that support .NET and LINQ is the major drawback of this approach, however the benefits can be substantial..

In [12], UniHadoop is presented. Its main focus is to integrate Unicore, a well known Grid middleware tool (with limited however storage capabilities) with the Hadoop Distributed File System in order to leverage the advantages of the latter with regard to scalability, reliability and storage efficiency.

In [13], an automated framework is presented for choosing between different storage providers, based on user requirements and general characteristics of the providers such as cost or performance. This is an interesting approach that can be integrated with the framework presented in this paper, mainly in the area of resource management and cloud federation selection.

3 OPTIMIS Data Manager Generic Functionality

The OPTIMIS Data Manager and Distributed File System (ODFS) solution is based on the Apache Hadoop DFS. In that implementation, there are two types of nodes that form the backbone of the file system. The NameNode is the central component that holds all the information regarding the location of the files, the location of the blocks and in general performs the major management actions. The DataNodes are the slaves that hold the actual data and are used also for the parallel processing of tasks that act on these data (in the standard usage of Hadoop), in the MapReduce framework. In order to increase the capacity of the distributed file system, more DataNodes may be added on the fly. Even though Hadoop was mainly created for implementing the MapReduce framework, its data management features can be utilized in a generalized manner for offering Storage as a Service, which is also the usage in the scope of this paper. Even though this master/slave architecture is characterized by a single point of failure (the NameNode), there are efforts ([21],[22] and [23]) that aim to overcome this obstacle, for improvement of availability or scalability. Our main concern is not in this part of the framework.

In order to be able to offer Storage as a Service through HDFS, one needs to implement suitable RESTful interfaces that must be created on top of this component in order to offer the generic functionality. Applications or users may at any time access and process the data based on suitable interfaces (and a necessary key) such as the sshFS interface ([17]). These interfaces help HDFS appear as a simple mounted directory in the service VMs that are running inside or ourside of the cloud.

So in the OPTIMIS cloud, whenever a new user is registered in the system, an account is created on his/her behalf on this distributed filesystem. Through this storage space, service VMs (that have been contextualized in a previous phase with the necessary information like the ssh key) may share data and perform operations on them. Automatic fault tolerance is achieved, since more than one replicas may be created. In addition, the existence of the replicas plus the distribution of the file parts themselves creates an automated load balancing feature, especially for read operations on the data. More information on the general HDFS core architecture can be found in [16].

Furthermore, a key characteristic is the ability to be flexible. For this reason, we use the notion of storage VMs. A storage VM is a DataNode that is contained inside a VM. The ODFS consists of a number of such VMs, exploiting their storage space in order to distribute it among its users. This aids in toggling the amount of resources used by the Data Management system in a very flexible and generic way. What is more, it makes the ODFS solution available for usage also in the PaaS layer, since it can be directly used over an IaaS provider. For this reason, also the NameNode is encapsulated inside a VM.

4 Key Functionalities

In the previous section, the generic functionality of the ODFS was presented. In this section the key characteristics that provide added value for the ODFS are described (Figure 1). The major aims of the OPTIMIS project ([30]) are the following:

-create a management environment whose decision making will be defined after taking under consideration parameters such as Trust, Risk, Eco-efficiency and Cost (TREC).

-enable the usage and interconnection of resources in different Cloud providers through federation. This interconnection may be driven by various factors such as TREC policies, shortage of internal resources etc.

- analyze the legal requirements with regard to data management in clouds and implement an according framework.

4.1 Management Based on TREC factors

In this effort, the data management framework exposes necessary interfaces in order to define its policies based on the aforementioned parameters. This goal has significantly affected the design of the ODFS. The notion of storage VMs was based on the ease of management that is required in order to follow TREC-based metrics. The latter are expected to characterize each VM, so that in the end the Data Manager will be able to adapt to the infrastructure goal through enabling or disabling one or more storage VMs.

This may be based on a modeling framework, that will indicate the necessary storage VMs that must be decommissioned in order to save energy or cost, from the available array of VMs used by the ODFS. Alternatively, it can be based on predefined policies of the Data Manager (e.g. Eco mode, Performance mode etc.).In this effort, we utilize the graceful decommission process of HDFS, which guarantees that DataNodes are made inactive through a process that keeps data integrity and prevents file system corruption.

The selection of which storage VMs will be decommissioned can be based on the current energy consumption of each VM in conjunction with the amount of storage utilized by the latter and the replication overhead.

4.2 Cloud Federation

When instructed to do so, the ODFS enables suitable interfaces with the external providers in order to launch storage VMs in the latter and extend the distributed file system (Figure 1). In order to achieve interoperability the Apache Libcloud ([29]) may be used. This is an abstraction layer above the calls to external providers that resolves the differences in their respective interfaces. Main advantages of this approach are the following:

-One universal file system as seen from the services VMs

-Transparent federation to the end user

-Standard way of accessing the file system through regular interfaces and shared storage between different service VMs thus reducing data lock-in

-The contents of the federated DataNode are automatically replicated in other internal DataNodes as a standard HDFS process thus making the undeployment process faster

-The Data Management solution provided by OPTIMIS can work on any Cloud (with the transformation of the Data Management VMs in the according hypervisor), since it is abstracted on a VM level.

Main disadvantages include the need for plug-in clients for each different Cloud provider in order to launch the federated storage VMs (minimized through the Apache Libcloud project) and for processing power (storage VMs)

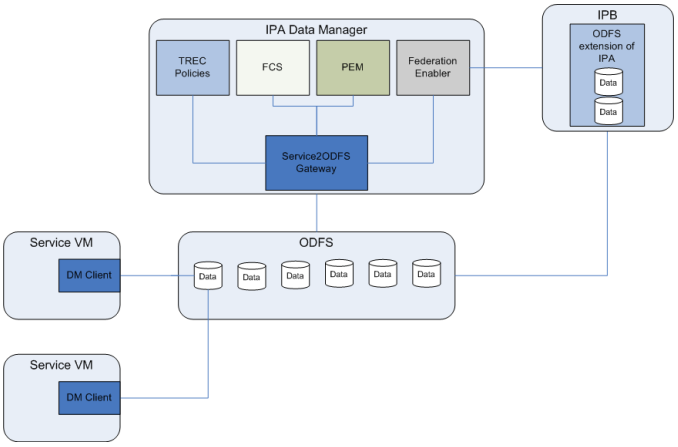

Figure 1: General Architecture and Federation of ODFS

4.3 Compliance to Legal Requirements (PEM)

The legal framework with regard to personal data processing is a hindering factor in the uptake of Cloud technologies. This is especially true due to the legal gaps in terms of the definition of personal data and the nature of Cloud computing, as is mainly expressed through the Cloud provider's lack of awareness for client usage and processing ([18]). In the context of the OPTIMIS project, an extensive analysis is made on these limitations ([19]). These pose a significant set of requirements in a legally compliant Cloud provider ([20]), namely data location monitoring/awareness, access control and security, replication and location-constrained placement.

While some of them are met rather easily through the standard functionality of HDFS (like replication), others pose more significant challenges. The most critical is location management, since the geographical area of data storage is one of the crucial factors from a legal point of view. These limitations are stated in the Service Level Agreement (SLA) between the client and the Cloud Provider. For instance, in OPTIMIS the SLA has specific sections making distinct the countries allowed for the user's data to be transferred and stored ([20]).

4.4 Federation Candidate Selection (FCS)

Federation to external IPs may occur for a variety of reasons, like temporary TREC violation, storage space depletion etc. However, when deciding which services to federate, one should take under consideration factors like forthcoming access to data and if this happens over a remote location. For example, one should federate a service VM that is anticipated to have the least interactivity with the data located on the internal shared storage. So, in a nutshell, the way the services will treat their centralized data (on the ODFS) in the forthcoming time intervals is critical for a successful federation selection. The framework should be able to predict this access pattern in order to implement good practice rules for optimizing the infrastructure setup.

However in order to have such a selection mechanism a framework is needed that will perform the following actions:

-Log user accesses on the ODFS through extending the FUSE interface.

-Preprocess these logs in order to reduce their size and extract meaningful information. In order to be utilized in the next

step, they are merged and transformed into average read or write accesses over a specified time interval (e.g. 15 minutes).
-Apply them as a dataset to a prediction algorithm that will utilize them in order to extract anticipated future accesses

In Table 1, the main interfaces specified for the Data Manager are contained. These are derived from the main description in Section 3 plus the key functionalities that are described in Section 4.

Table 1: Reduced List of REST interfaces for Data Manager

GET account/create/{sid}	Create a new account to the data manager and ODFS for a specific service
GET /storage/usage/{sid}	Returns the amount of disk space used by the service in the ODFS
GET /storage/locations	Returns the available storage locations of this IP (necessary so that the SP can filter the IPs that do not meet the location requirements)
POST /storage/locale/{sid}	Give location constraints to the Data Manager (for preventing a service's data to be transferred in a specific federated location)
POST /federation/new	Order the Data Manager to establish a storage federation with an external provider
POST /policy/{PolicyID}	Specify Policy mode based on TREC tools
GET /ranking/	Sort service VMs based on their anticipated I/O operations in the forthcoming intervals

5 PRELIMINARY RESULTS

At the moment, the implementation of the ODFS is undergoing. In this section the available features and preliminary results at this stage are described.

5.1 Federation and Legal Requirements

The federated prototype of the OPTIMIS Data Manager at this stage consists of a central ODFS installation that is located in the OPTIMIS testbed in Spain. In order to check the inclusion of a federated DataNode, a DataNode VM was initiated and successfully included in the secondary testbed in Umea, Sweden. The only difference was the increase in network delays for the transfers. For the future, our main intention is to utilize the exposed APIs from commercial IPs and launch automatically DataNode VMs in their resources.

In order to implement the location monitoring and placement feature, the rack awareness features of HDFS were utilized, in conjunction with geolocation capabilities in order to be able to extract the geographic regions for the data storage locations. The OPTIMIS data manager validates the placement of user data in specific geographical locations and actively enforces the location of the DataNodes to specific countries and data centers in order to fully comply with the SLA. Furthermore, a GUI has been implemented (Figure 2) that is linked with ODFS and that provides real-time monitoring information to the end user regarding the location of their data. This is a critical legal requirement.

5.2 Security additions

Hadoop and HDFS have a weak security model, in particular the communication between Datanodes and between clients and datanodes is not encrypted. All data exchanges are performed in plaintext. The user authentication is performed by checking the user id without avoiding malicious users or distributed processes from impersonate any user and accessing their data. In newer versions of Hadoop the Kerberos authentication model was planned to be used. The current situation is that most of the installations in production environments are using the stable edition and the security features are disabled. In OPTIMIS we enhanced the security model of Hadoop using virtual private networks. The communication between the data nodes and the remote procedure calls (RPC) of the distributed file system are encrypted. The virtual machine hosting the DataNodes and the master NameNode are fully protected against eavesdropping and network attacks using a Virtual Private Network based on OpenVPN. Every DataNode is authenticated using an SSH key to the VPN server. If the authentication is successful the DataNode becomes member of the ODFS cluster. The authentication is established using the Data Manager as a safe gateway-proxy for user's interaction. This is especially critical in the case of federation of DataNodes across different IPs.

For the interaction of services with the ODFS, the service VM communicates with the Data Manager in order to perform operations on the distributed storage. The Data Manager verifies the user identity using a secret key, generated during the account creation and analyzes the user's intentions. The intentions are matched and checked if they comply with the SLA, the intention is digitally signed by the Data Manager and can be verified by any distributed node in the cluster. Every node has the certificate of the Data Manager and can verify the authenticity of the requested action. The Data Manager is responsible for approving an action and filtering malicious intentions.

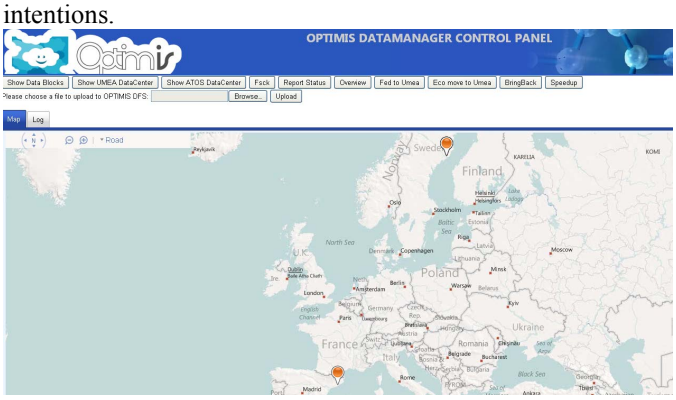

Figure 2: GUI for end user to monitor location of their data (legal requirement)

5.3 Federation Candidate Selection

In order to implement the time series prediction described in 4.4, we used the dataset that is provided from [28]. The logs provided from this have registered all the actions along with a timestamp, the process ID, the size and type of the transaction.

In order to be able to utilize them in the envisaged way (to create a time series of data activity metrics in predefined time intervals) we implemented a java module that performs automatically this task. This program took as arguments the needed time interval over which the samples would be compressed (e.g. 5 minutes, 15 minutes etc.) and produced the time series of average read/write operations. It is anticipated that this will be done in the near future through a MapReduce task, in order to exploit the parallelism of HDFS and reduce preprocessing time. We chose the 15-minute interval so that a sufficient length of the time series would be gathered from the available dataset (around 670 values).

This time series was split into two sets, a training and a validation one. The former was used to train a NARX network ([26]). The Non-Linear Autoregressive ANN utilizes the formula depicted in Figure 3 in order to predict the future accesses and has been utilized with success in the time series prediction field (e.g. [27]). It mainly utilizes past values of the original signal (y(t)) in addition to a potential exogenous input (u(t)). In our case y=u, so that we can predict the future accesses based only on the pattern of the past accesses. Furthermore, the non-linear functions of the neurons of the ANN help in adapting to sudden changes in the pattern, like peaks in activity.

$$y(t) = f(y(t-1), y(t-2, \ldots, y(t-n_y),$$
$$u(t-1), u(t-2), \ldots u(t-n_u))$$

Figure 3: Formula of the NARX ANN

Through the training process, the network learns how the previous values affect the following by identifying patterns in the past and arranges the coefficients of each factor (weights and biases). Afterwards, the validation set is used in order to extract the error of the approach. The network is given the initial feed of time series values (the last of the training set) and starts predicting the values of the validation set. It takes as input the time series of the average values of data management actions in the previous 15-minute points and produces a prediction regarding the anticipated activity in the following 15-minute intervals. The look-ahead can be set to whatever desired interval.

In order to optimize the structure of the NARX ANN and improve its performance, we utilized a Genetic Algorithm (GA) that defined the most critical parameters of the network through an iterative evolutionary process (similar to the approach followed in [31]). This is critical since in an automated framework, this process must be performed without any human intervention for deciding the fittest configuration of the ANN. For each different setup, the performance of the network on the validation set was returned, in order to characterize the fitness of a specific network configuration. As the iterations (generations) of the GA proceed, the network's characteristics are refined so that the error of the latter is minimized. For the GA we altered 3 parameters, the number of generations (50 to 200 with a step of 50) , the elite members (2 to 10 with a step of 2) that were passed unchangeable to the next iteration and the crossover fraction (0.2 to 1 with a step of 0.2). The best setup of the GA was the one with 50 generations, 10 elite members and 0.8 crossover fraction. The prediction result for the optimized ANN appears in Figure 4 while the parameters of the ANN decided by the GA along with their optimized values appear in Table 2. The implementation was based on Matlab R2007b.

Table 2: Structure of the NARX ANN

Transfer Functions	Tansig
Number of Neurons (hidden)	3-1
Training Function	Trainbr
Initial Feed	35 previous values
Number of Training Epochs	665

From Figure 4, it can be observed that at the moment the algorithm lacks accuracy with regard to the exact time interval in which the peak will occur. However it can achieve significant accuracy in the area around this interval. So the fittest usage is answering to a question of the sort "which is the service that will read the most data in the following two hours". Given that the federation of VMs is not something that will occur for a very small period of time but in a range of hours or days, this limitation is not considered critical. Furthermore, even though the time series presents significant anomalies, with the most prominent being the sudden peaks in operations, the NARX network is able to identify them in advance. The prediction is made at interval 0 and looks at 200 15-minute intervals forward.

At this point it is necessary to stress that this prediction of future activity can be utilized in many other ways in order to optimize the management of the storage infrastructure. For example, one of the identified as more interesting research issues in the Hadoop community[24] is the dynamic replication of data in order to meet peaks in demand and based on the access patterns. Through the aforementioned framework, this can be easily achieved through linking the predictions with the change in the replication ratio.

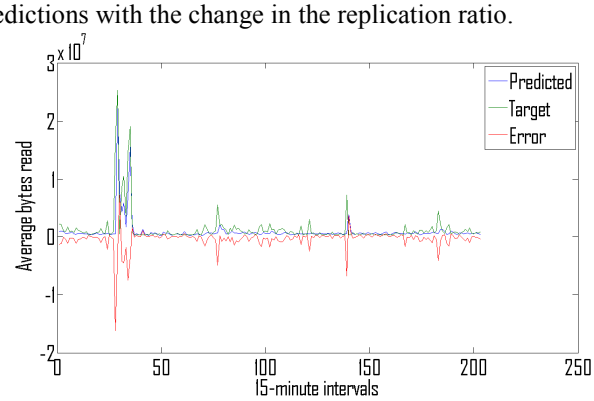

Figure 4: Prediction of read accesses in future 15-minute intervals. The prediction is made at interval 0 and looks at 200 15-minute intervals forward. It utilizes an initial feed of past values (period before 0) and then it feeds itself with the predicted values for the continuation of the process.

6 CONCLUSIONS AND FUTURE WORK

. In this paper, an innovative mechanism for managing data in Cloud infrastructures has been described. The implementation is still in progress but with substantial level of fulfillment at this stage. The design followed had the goal of increasing flexibility and create a legally compliant data management framework that can be handled also based on different optimization aspects.

The baseline technology used (Apache Hadoop project) has

allowed to exploit its advanced design aspects and at the same time extend it in order to enrich its purposes. The implemented interfaces and security additions as well as the increased functionalities provided by this work may extend its role from a back-end internal processing system to a full scale front-end data management framework for Cloud infrastructures.

For the future, the aim is to complete the implementation based on the proposed design regarding all the key characteristics of the system, like the TREC parameters. Furthermore, the usage of the Apache Mahout project ([25]) will be pursued so that even the core algorithm described in Section 5.3 can be executed in a parallel fashion.

ACKNOWLEDGMENT

This research is partially funded by the European Commission as part of the European IST 7th Framework Program through the project OPTIMIS under contract number 257115.

REFERENCES

[1] http://aws.amazon.com/ec2/
[2] http://www.flexiant.com/products/flexiscale/
[3] http://www.rackspace.com/cloud/
[4] http://hadoop.apache.org/
[5] Qinlu He; Zhanhuai Li; Xiao Zhang; , "Study on Cloud Storage System Based on Distributed Storage Systems," Computational and Information Sciences (ICCIS), 2010 International Conference on , vol., no., pp.1332-1335, 17-19 Dec. 2010, doi: 10.1109/ICCIS.2010.351
[6] Yingjie Shi, Xiaofeng Meng, Jing Zhao, Xiangmei Hu, Bingbing Liu, and Haiping Wang. 2010. Benchmarking cloud-based data management systems. In Proceedings of the second international workshop on Cloud data management (CloudDB '10). ACM, New York, NY, USA, 47-54. DOI=10.1145/1871929.1871938 http://doi.acm.org/10.1145/1871929.1871938
[7] Brian Bockelman, Using Hadoop as a Grid Storage Element, Journal of Physics, Conference Series 180 (2009) 012047
[8] Hill, Z.; Humphrey, M.; , "CSAL: A Cloud Storage Abstraction Layer to Enable Portable Cloud Applications," Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on , vol., no., pp.504-511, Nov. 30 2010-Dec. 3 2010 doi: 10.1109/CloudCom.2010.88
[9] Weimin Zheng, Pengzhi Xu, Xiaomeng Huang, and Nuo Wu. 2010. Design a cloud storage platform for pervasive computing environments. Cluster Computing 13, 2 (June 2010), 141-151. DOI=10.1007/s10586-009-0111-1 http://dx.doi.org/10.1007/s10586-009-0111-1
[10] Nicolae, B., Antoniu, G., Boug´e, L., Moise, D. and Carpen-Amarie, A. (2010). BlobSeer: Next generation data management for large scale infrastructures, Journal of Parallel and Distributed Computing 71(2): 168–184.
[11] GUNDA, P. K., RAVINDRANATH, L., THEKKATH, C. A., YU, Y., AND ZHUANG, L. Nectar: Automatic management of data and computation in data centers. In Proceedings of OSDI (2010).
[12] Wasim Bari, Ahmed Shiraz Memon, and Bernd Schuller,"Enhancing UNICORE Storage Management Using Hadoop Distributed File System",EURO-PAR 2009 – PARALLEL PROCESSING WORKSHOPS,pp. 345~352, 2009
[13] Arkaitz Ruiz-Alvarez and Marty Humphrey. 2011. An automated approach to cloud storage service selection. In Proceedings of the 2nd international workshop on Scientific cloud computing (ScienceCloud '11). ACM, New York, NY, USA, 39-48. DOI=10.1145/1996109.1996117 http://doi.acm.org/10.1145/1996109.1996117
[14] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: stateof- the-art and research challenges," Journal of Internet Services and Applications, vol. 1, pp. 7–18, 2010. [Online]. Available: http://dx.doi.org/10.1007/s13174-010-0007-6
[15] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia. Above the Clouds: A Berkeley View of Cloud computing. Technical Report No. UCB/EECS-2009-28, University of California at Berkley, USA, Feb. 10, 2009.
[16] http://hadoop.apache.org/common/docs/current/hdfs_design.html
[17] http://fuse.sourceforge.net/sshfs.html
[18] Hon, W. Kuan, Millard, Christopher and Walden, Ian, The Problem of 'Personal Data' in Cloud Computing - What Information is Regulated? The Cloud of Unknowing, Part 1 (March 10, 2011). Queen Mary School of Law Legal Studies Research Paper No. 75/2011. Available at SSRN: http://ssrn.com/abstract=1783577
[19] OPTIMIS Project Deliverable D7.2.1.2: "Cloud Legal Guidelines: Data Security, Ownership Rights and Domestic Green Legislation", LUH and other partners of the OPTIMIS Consortium, May 2011, available at http://www.optimis-project.eu/
[20] Benno Barnitzke, Wolfgang Ziegler, George Vafiadis, Srijith Nair, George Kousiouris, Marcelo Corrales, Oliver Wäldrich, Nikolaus Forgó and Theodora Varvarigou, "Legal Restraints and Security Requirements on Personal Data and Their Technical Implementation in Clouds", to appear in eChallenges 2011, 26-28 Oct. 2011, Florence, Italy
[21] Feng Wang, Jie Qiu, Jie Yang, Bo Dong, Xinhui Li, and Ying Li. 2009. Hadoop high availability through metadata replication. In Proceeding of the first international workshop on Cloud data management (CloudDB '09). ACM, New York, NY, USA, 37-44. DOI=10.1145/1651263.1651271 http://doi.acm.org/10.1145/1651263.1651271
[22] AS Talwalkar, "HadoopT-Breaking the Scalability Limits of Hadoop", Master Thesis, January 2011, Available at: https://ritdml.rit.edu/bitstream/handle/1850/13321/ATalwalkarThesis1-2011.pdf?sequence=1
[23] F. Marozzo, D. Talia, P. Trunfio, "A Peer-to-Peer Framework for Supporting MapReduce Applications in Dynamic Cloud Environments". In: Cloud Computing: Principles, Systems and Applications, N. Antonopoulos, L. Gillam (Editors), Springer, chapt. 7, pp. 113--125, 2010. Note: ISBN 978-1-84996-240-7.
[24] http://hadoopblog.blogspot.com/
[25] http://mahout.apache.org/
[26] http://www.mathworks.com/help/toolbox/nnet/ug/bss36dv-1.html
[27] Maria P. Menezes, Jr. and Guilherme A. Barreto. 2008. Long-term time series prediction with the NARX network: An empirical evaluation. Neurocomput. 71, 16-18 (October 2008), 3335-3343. DOI=10.1016/j.neucom.2008.01.030 http://dx.doi.org/10.1016/j.neucom.2008.01.030
[28] Dushyanth Narayanan, Austin Donnelly, and Antony Rowstron. 2008. Write off-loading: Practical power management for enterprise storage. Trans. Storage 4, 3, Article 10 (November 2008), 23 pages. DOI=10.1145/1416944.1416949 http://doi.acm.org/10.1145/1416944.1416949
[29] http://libcloud.apache.org/
[30] A.J. Ferrer, F. Hernández, J. Tordsson, E. Elmroth, A. Ali-Eldin, C. Zsigri, R. Sirvent, J. Guitart, R.M. Badia, K. Djemame, W. Ziegler, T. Dimitrakos, S.K. Nair, G. Kousiouris, K. Konstanteli, T. Varvarigou, B. Hudzia, A. Kipp, S. Wesner, M. Corrales, N. Forgó, T. Sharif, and C. Sheridan. OPTIMIS: a Holistic Approach to Cloud Service Provisioning, Future Generation Computer Systems, Elsevier, Vol. 28, No. 1, pp. 66-77, 2012.
[31] George Kousiouris, Tommaso Cucinotta, Theodora Varvarigou, "The Effects of Scheduling, Workload Type and Consolidation Scenarios on Virtual Machine Performance and their Prediction through Optimized Artificial Neural Networks , The Journal of Systems and Software (2011),Volume 84, Issue 8, August 2011, pp. 1270-1291, Elsevier, doi:10.1016/j.jss.2011.04.013.