A new Data Placement approach for Scientific Workflows in Cloud Computing environments

Hamdi Kchaou[1], Zied Kechaou[1] and Adel M. Alimi[1]

[1] The University of Sfax, National School of Engineers (ENIS),
REsearch Groups in Intelligent Machines (REGIM),
BP 1173, Sfax 3038, Tunisia
{hamdi.kchaou, zied.kechaou, adel.alimi}@ieee.org

Abstract. The reach of Cloud Computing technologies approved distributing with massive data applications such as Scientific Workflows, which processing huge scientific data in dispersed computing infrastructures. Among the characteristics of Cloud Computing, we mention the elasticity that allows workflows to dynamically stipulate necessary resources for tasks execution. The processing of massive data with scientific workflows increase the data transmission, rise execution delay and it request huge bandwidth cost. So, to reduce the execution cost of workflows and the data movements, data placement optimization technics must be taken into consideration. While placing datasets during execution of tasks for a job in a workflow, there are dependencies between datasets and between tasks. In this paper, we propose a data placement approach based on heuristic genetic algorithm which takes into accounts control and data flow dependency, in order to reduce data movements and so the utilization of resources in cloud environments.

Keywords: Cloud Computing; Massive Data; Scientific Workflow

1 Introduction

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and re-leased with minimal management effort or service provider interaction [1]. The cloud model comprises five essential features (On-demand self-service, Broad network access, Resource pooling, Rapid elasticity, Measured service, Resource usage), three service models (SaaS: Software as a Service, IaaS: Infrastructure as a Service and PaaS: Platform as a Service), and four deployment models (private cloud, community cloud, public cloud and hybrid cloud). The SaaS includes software applications, the PaaS contains platform applications and the IaaS take account of hardware resources as CPU, Memory, Disk and Bandwidth. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers [2]. The cloud computing have the ability to exploit large amounts of computing

and storage resources used to execute complex applications such as scientific workflow.

A workflow is the automation of a process, during which data is processed by different logical data processing activities according to a set of rules [3]. Workflows are usually modeled as directed acyclic graphs (DAGs) such that workflow tasks are represented as graph vertices and the data flows among tasks are represented by graph edges. The direction of edges shows data flows among tasks [4]. The data required by scientific workflows are frequently big and distributed, that is why data placement becomes difficult. The adequate placement of data in data centers can reduce the data movements and equilibrium the load of data centers. It is ineffective to place all data sets in single data center and in order to achieve computing tasks; entirely requisite data sets must be placed in the same data center. So due to bandwidth limitations, we need to place datasets in one data center in the largest amount possible, to reduce data movements between storage nodes and compute nodes. Therefore, we can say as it is represented in figure 1, data placement is to place datasets in data centers in cloud computing environments. Each dataset has a size and each data center has a storage space. The data placement is a NP-hard problem. In order to solve this type of problems and as mentioned in [5], [6] and [7], genetic algorithm have proven a better performances as an optimization technics for data placement.

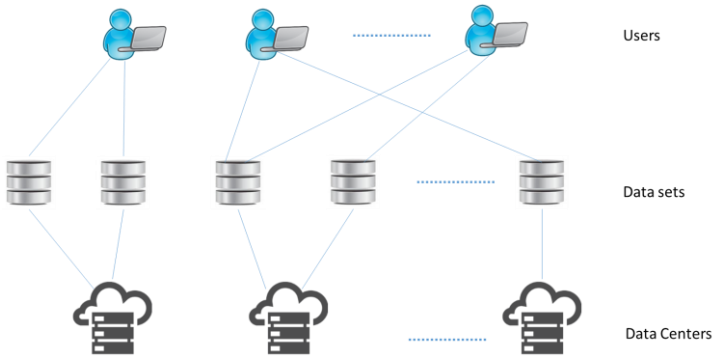

Fig. 1. Data placement on Data Centers under user's demands

To deal with massive data, MapReduce is one of the most way. Announced in 2004, MapReduce is a framework for large scale data processing using commodity clusters [8]. In the MapReduce programing model, a job is divided into two continuous phases which are named as the map phase and reduce phase, then the latter does not start until the former is complete [9]. The map and reduce tasks will be distributed to nodes and executed in parallel.

In this paper, we presented a novel approach for placement data with scientific workflows in cloud computing environments, which includes:

- Genetic algorithm for data placement
- Dependencies between tasks of a MapReduce job and which illustrate the control flow in scientific workflow.

- Dependencies between datasets consumed and produced by MapReduce jobs.

The remainder of this paper is organized as follows. Section 2 illustrates the problem, describes the proposed approach and suggest a formulation of problem. Section 3 discusses the related work. Section 4 contains experimental simulation results. Section 5 summarizes our conclusions and points out the future work.

2 Proposed approach

2.1 Description

A workflow is mainly imperative for applications in which data dependencies occur among the tasks. The dependencies represent the flow of data between workflow activities from data producer to data consumer. In control-driven workflows, or control flows, the connections between the activities in a workflow represent a transfer of control from the preceding task to the one that follows [10]. Control dependencies indicate a constraint on the relative order of execution of two tasks, without implying data transfer. They are only necessary for tasks with side effects, where data dependencies alone are not sufficient to determine the ordering [11]. Our pro-posed approach combines control flow and data flow, which called hybrid according to [10]. In this workflow, the tasks will be a MapReduce Jobs. Since MapReduce is data-driven, the hybrid workflow will be inclined on the road to data flow. In the other side, a job can be split into many map and reduce tasks, which are dependent to each other's. The map task is an input for the reduce task which can be an input for another map task and so on. This illustrates the control dependencies between tasks.

2.2 Problem formulation and proposed approach

MapReduce is known as one of the most powerful framework to processes massive datasets. In order to solve problems of MapRe-duce plateforms, researchers integrate MapReduce into scientific workflows like in [12]. We integrates MapReduce jobs in the mod-el formulation which can be denoted as:

$$W = (N,J,E,D) \quad (1)$$

Where, W denotes the workflow, N is a set of Nodes, as vertices in a DAG (Directed Acyclic Graph), which contains a number of jobs. J is a number of MapReduce jobs which can be split into map tasks and reduce tasks. D represents a set of datasets. E is a set of directed edges. The data flow is controlled by the dependencies between datasets used by jobs. Each job ji, can be composed by one or several map tasks and reduce tasks. The control flow is measured by the dependencies between jobs of nodes specified by directed edges between nodes. A directed edge between N1 and

N2 means that j4 depends on j1, j2 and j3. N1 produces d3 and j4 consumes it as an input.

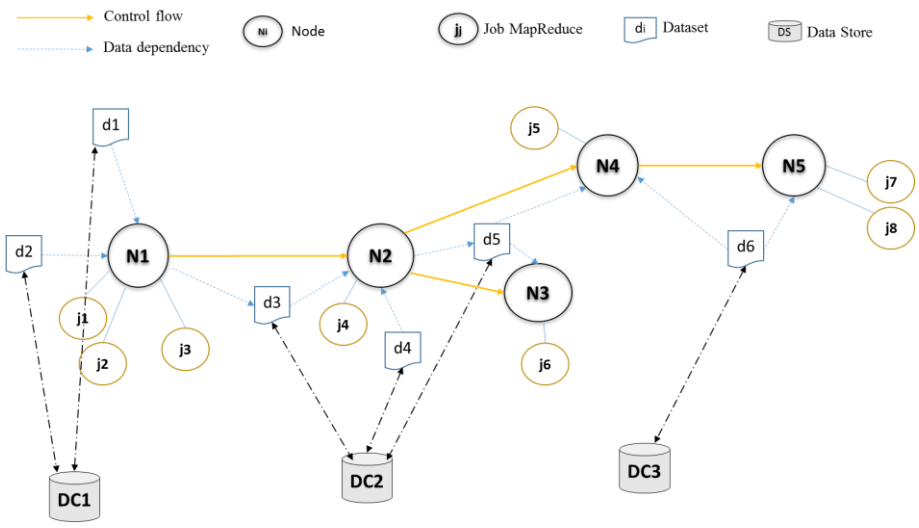

Fig. 2. Example of the proposed scientific application approach

In Figure 2, N = {N1, N2, N3, N4, N5}; d1, d2 are the input data of t1; d3 is produced after executing task t1; d3 and d4 are the input data of t2; d5 is the input data of t3; d6 is the input data of t4 and t5; j1, j2, j3 compounds t1; j4 compounds t2; j5 compounds t4; j6 compounds t3; j7, j8 compounds t5. d1 and d2 belongs to DC1; d3, d4 and d5 have its place on DC2; d6 belongs to DC3.We can classify datasets into two groups: initial datasets and generated datasets like d3 and d5. So, we are dealing with two categories of data placement offline due to the initial datasets and online due to the generated datasets.

DC = set of data centers

DS = set of datasets

$D_{ij} = |\, j_i \cap j_j \,| \quad (i, j < n)$

Which means the number of datasets required by both ji and jj forms a dependency matrix D. n is the total number of datasets.

A simple example, takes three datasets and two data centers. Job j1 requires two datasets ds1 and ds2, j2 requires the three datasets, and the dependency matrix is:

$$D = \begin{pmatrix} 2 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

2.3 Heuristic genetic placement algorithm

According to [5] and [13], in order to solve the data placement problems, many optimization and evolutionary algorithms can be used, but the GA (Genetic Algorithm) (steps in figure 3) has been proven to have a better performance in many cases. GAs were first described by John Holland in the 1960s and further developed [14].

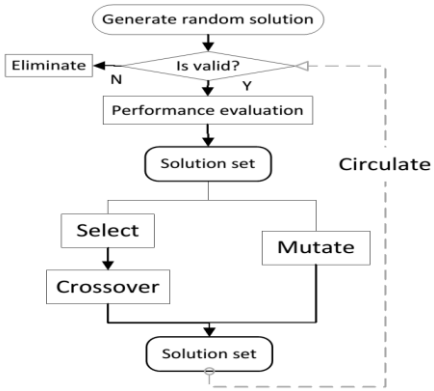

Fig. 3. Steps of the Genetic Algorithm [5]

A. Encoding Rules

To encode the problem in this paper to the genetic algorithm, we use integer-encoding rules and to each gene, there is a placement strategy. The placement of data in cloud is represented by two matrices. The first matrix is the data dependency and the second matrix is the MapReduce jobs control dependency matrix.

B. Individual and population

A gene of individual is the placement of datasets on data centers. It means that the j^{th} dataset placed in the gj^{th} data center. For example there three data centers and 6 datasets. The gene code (311232) means that dataset 1 is placed in data center 3 and so on.

C. Fitness Function

The fitness function reveal the degree of dependency between datasets and the degree of control dependency among MapReduce jobs. We denoted D_t as the total degree of dependency, which is defined as:

$$Dt = \sum dsi, dsj + \sum ji, jj$$

The fitness function is: $1/ D_t$

D. Optimization Placement Strategy

We propose below a heuristic to minimize data movement:

Steps for proposed heuristic algorithm

1. For i = 1, ….., n do

2. Calculate the dependencies of dataset ds_i with each DC,
   Calculate the dependencies of job j_j with each DC
3. Get the maximal dependency obtained in step 2
4. Place the dataset ds_i to the suitable data center
   Place the job j_j to the suitable node
5. If the size of ds_i exceeds the storage capacity of the selected data center, find the next maximal dependency until this dataset can be placed in the suitable data center and the job j_j to the suitable node;

3 Related works

In our proposed approach, we investigate the combination of three concepts, which are the placement of massive data, the scientific workflow and the MapReduce model. We conducted our state of the art through the combination of these concepts as denoted in table 1. To compare works in these domains we created three pairs that resulted in 'Placement in MapReduce', 'MapReduce with scientific workflows' and 'Placement of massive-data in scientific workflows'.

Table 1. Comparative study of data placement approaches

Authors	Placement in MapReduce		MapReduce with Scientific workflows		Placement of massive data in scientific workflows	
	Jobs dependencies		With placement?		Total dependency	
	Yes	No	Yes	No	Yes	No
[Our approach]	*		*		*	
[Q. Zhao et al. 2016]						*
[K. Deng et al. 2015]					*	
[M. Ebrahimi et al. 2015]						*
[L. Cui et al. 2015]					*	
[Q. Zhao et al. 2015]						*
[L. Zeng et al. 2015]						*
[N. Mohamed et al. 2014]				*		
[M. Wang et al. 2014]						*
[J. Wang et al. 2014]		*				
[F. Ma et al. 2012]						*
[Z. Tang et al. 2012]				*		
[X. Fei et al. 2012]				*		
[N. Maheshwari et al. 2012]		*				
[E. Zhao et al. 2012]						*
[D. Yuan et al. 2011]						*
[Y. He et al. 2011]		*				
[P. Nguyen et al. 2011]				*		
[D. Yuan et al. 2010]						*
[X. Fei et al. 2009]				*		

3.1 Placement in MapReduce

In this section, we takes works that consider placement in MapReduce and then we classified them between two types to know which contains dependencies between MapReduce jobs and which not. Almost of papers [15], [16] and [17] mentioned in this section do not care about job dependency. We think as explained in section 2, that dependencies between jobs is an important factor that affects the placement of massive data in such environments. Some works considers dependencies on data, like in[15]. There is no link between jobs in the case of multiple jobs with multiple datasets. The placement of jobs is treated separately. Therefore, in this work, it is more interesting to place jobs that have common interest together. In [16], authors proposes an energy efficient data placement algorithm that conserve energy for large data centers running MapReduce jobs. In addition, it concerns also data dependencies between jobs. There are some works like [17], which ameliorate the placement of data in MapReduce by proposing a placement storage structure for Hadoop and how to better store data in HDFS blocs. Each bloc is stored independently.

3.2 MapReduce with Scientific Workflows

In this part and according to Table 1, we cover works deploying scientific workflows, which integrates the MapReduce model, and we classified them into two types to know which presents placement algorithms and which not. All works founded [18], [12], [19], [20] and [21] do not holds placement solution. In [19], authors proposes a dataflow based scientific workflow in which the jobs are Map and Reduce. In [19] and [20], authors considers dependencies between datasets. The authors in [12], proposes a MapReduce-enabled Scientific Workflow Framework with static optimization scheduling policy in heterogeneous clusters to get higher performance. However, the dependency between data is considered. In [21], authors proposes a workflow system for integrating structure, and orchestrating MapReduce jobs for scientific data intensive workflows. All cited works in this sector do not presents control dependencies in the workflow proposed.

3.3 Placement of Massive data in Scientific Workflows

In this part and refers to Table 1, we consider works that concerns the placement of massive data in scientific workflow. Then we classified them into two parts, the first one contains total dependencies that means control and data dependencies. The second one only one type of workflow dependency. The paper in [23], proposes a data placement method, which could effectively reduce the data movement between the data centers and there is a data dependency between the nodes of datasets. In [24], authors proposes a Security-Aware and Budget Aware workflow scheduling strategy (SABA), which holds an economical distribution of tasks among the available CSPs (Cloud Service Providers) in the market, to provide customers with shorter make span as well as security services. The proposed approaches to cluster tasks based on data dependency. A matrix bases k-Means clustering strategy for data placement in scien-

tific cloud workflows systems is done in [25]. The proposed strategy attempts to minimize the total data movement during the execution of workflows by placing and data dependency is reflect-ed. In [26], authors proposes an algorithm that can calculate the minimum cost for intermediate dataset storage in scientific cloud workflows systems. It builds an intermediate data dependency graph (IDG) from the data provenances in scientific workflows. Besides, another work proposes a data placement strategy based on data dependency clustering for scientific workflow in heterogeneous cloud in [27]. In [4], and [7], data dependency is taken into consideration. At the best of our knowledge, there are two works presenting a total dependency (control and data flow), one presents a K-cut graph-partitioning algorithm to minimize the volume of data transfer [22]. Besides, another propose a genetic algorithm based data replica placement strategy to reduce data transmissions in cloud [5].

4 Simulation Results

In the experiment, we run the simulation with test workflows on twenty data centers and we evaluate the number of data movements. We compare our proposed strategy to the random strategy where datasets and jobs are distributed to data centres and nodes randomly. The random strategy is the random data placement strategy used in Hadoop Distributed File System [28]. In figure 4, the number of data movements is indicated with diverse number of datasets. We assume that there is one job MapReduce per node in the proposed approach and in a future work we will adopt more than one job in each node. We execute a mapreduce job on the Ambari console TechSandBox lunched on my private cloud space offered by IBM demo cloud, as shown in figure 5.
The experiment is taken on a laptop with Intel Core i3 380M 2.53GHz, RAM 6GB, DISK 500GB, Bandwidth 100MB.

Fig. 4. Data movements

From figure 4, with the increase of the number of data sets, the number of data movements is also increasing. The results shown in figure 4 proves that our proposed

placement strategy can reduce the number of data movements better than the random strategy.

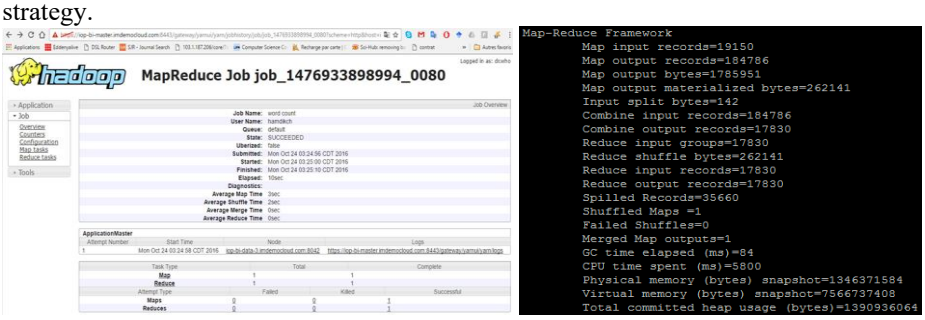

Fig. 5. Running a MapReduce job with Ambari

5 Conclusion

This paper outlines the current issues for data placement in scientific applications. It suggests an approach that can reduce the number of data movements. The proposed approach in this paper integrates the MapReduce model on the scientific workflow in Cloud Computing environments. Precisely, it proposes a placement of multiple MapReduce jobs on scientific workflow with control and data dependencies between jobs. In the future, we will use an online-shared environment for data science like the Ambari Console for testing multiple Hadoop jobs per node on real cloud environment.

Acknowledgments. The authors would like to acknowledge the financial support of this work by grants from General Direction of Scientific Research (DGRST), Tunisia, under the ARUB program.

References

1. Mell, P., Grance, T.: The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology. Nist Spec. Publ. 145, 7 (2011).
2. Haghighat, M., Zonouz, S., Abdel-mottaleb, M.: Expert Systems with Applications Cloud-ID : Trustworthy cloud-based and cross-enterprise biometric identification. Expert Syst. Appl. 42, 7905–7916 (2015).
3. W. M. Coalition: Workflow management coalition terminology and glossary.
4. Ebrahimi, M., Mohan, A., Kashlev, A., Lu, S.: BDAP: A Big Data Placement Strategy for Cloud-Based Scientific Workflows. 2015 IEEE First Int. Conf. Big Data Comput. Serv. Appl. 105–114 (2015).
5. Cui, L., Zhang, J., Yue, L., Shi, Y., Li, H., Yuan, D.: A Genetic Algorithm Based Data Replica Placement Strategy for Scientific Applications in Clouds. Trans. Serv. Comput. 1374, 1–13 (2015).
6. Er-Dun, Z., Yong-Qiang, Q., Xing-Xing, X., Yi, C.: A Data Placement Strategy Based on Genetic Algorithm for Scientific Workflows. 2012 Eighth Int. Conf. Comput. Intell. Secur. 146–149 (2012).

7. Ebrahimi, M., Mohan, A., Lu, S., Reynolds, R.: TPS : A Task Placement Strategy for Big Data Workflows. 523–530 (2015).
8. Dean, J., Ghemawat, S.: MapReduce : Simplified Data Processing on Large Clusters. OSDI'04 Proc. 6th Conf. Symp. Opearting Syst. Des. Implement. 6, 1–13 (2004).
9. Song, J., He, H., Wang, Z., Yu, G., Pierson, J.-M.: Modulo Based Data Placement Algorithm for Energy Consumption Optimization of MapReduce System. J. Grid Comput. (2016).
10. Deelman, E., Gannon, D., Shields, M., Taylor, I.: Workflows and e-Science: An overview of workflow system features and capabilities. Futur. Gener. Comput. Syst. 25, 528–540 (2009).
11. Kelly, P.M.: Applying Functional Programming Theory to the Design of Workflow Engines. Science (80-.). (2011).
12. Tang, Z., Liu, M., Li, K., Xu, Y.: A MapReduce-enabled scientific workflow framework with optimization scheduling algorithm. Parallel Distrib. Comput. Appl. Technol. PDCAT Proc. 599–604 (2012).
13. Mitchell, M.: Genetic algorithms: An overview. Complexity. 1, 31–39 (1995).
14. Atay, Y., Kodaz, H.: Intelligent and Evolutionary Systems. 5, 43–55 (2016).
15. Wang, J., Shang, P., Yin, J.: DRAW: A New Data-gRouping-AWare Data Placement Scheme for Data Intensive Applications with Interest Locality. In: Cloud Computing for Data-Intensive Applications. pp. 149–174. Springer New York, New York, NY (2014).
16. Maheshwari, N., Nanduri, R., Varma, V.: Dynamic energy efficient data placement and cluster reconfiguration algorithm for MapReduce framework. Futur. Gener. Comp. Syst. 28, 119–127 (2012).
17. He, Y., Lee, R., Huai, Y., Shao, Z., Jain, N., Zhang, X., Xu, Z.: RCFile: A fast and space-efficient data placement structure in MapReduce-based warehouse systems. Proc. - Int. Conf. Data Eng. 1199–1208 (2011).
18. Mohamed, N., Maji, N., Zhang, J., Timoshevskaya, N., Feng, W.C.: Aeromancer: A workflow manager for large-scale MapReduce-based scientific workflows. Proc. - 2014 IEEE 13th Int. Conf. Trust. Secur. Priv. Comput. Commun. Trust. 2014. 739–746 (2015).
19. Fei, X., Lu, S.: A dataflow-based scientific workflow composition framework. IEEE Trans. Serv. Comput. 5, 45–58 (2012).
20. Fei, X.F.X., Lu, S.L.S., Lin, C.L.C.: A MapReduce-Enabled Scientific Workflow Composition Framework. 2009 IEEE Int. Conf. Web Serv. 663–670 (2009).
21. Nguyen, P., Halem, M.: A {MapReduce} Workflow System for Architecting Scientific Data Intensive Applications. Proc. 2Nd Int. Work. Softw. Eng. Cloud Comput. 57–63 (2011).
22. Deng, K., Ren, K., Zhu, M., Song, J.: A Data and Task Co-scheduling Algorithm for Scientific Cloud Workflows. IEEE Trans. Cloud Comput. 7161, 1–1 (2015).
23. Ma, F., Yang, Y., Li, T.: A Data Placement Method Based on Bayesian Network for Data-Intensive Scientific Workflows. 2012 Int. Conf. Comput. Sci. Serv. Syst. 1811–1814 (2012).
24. Zeng, L., Veeravalli, B., Li, X.: SABA: A security-aware and budget-aware workflow scheduling strategy in clouds. J. Parallel Distrib. Comput. 75, 141–151 (2015).
25. Yuan, D., Yang, Y., Liu, X., Chen, J.: A data placement strategy in scientific cloud workflows. Futur. Gener. Comput. Syst. 26, 1200–1214 (2010).
26. Yuan, D., Yang, Y., Liu, X., Chen, J.: On-demand minimum cost benchmarking for intermediate dataset storage in scientific cloud workflow systems. J. Parallel Distrib. Comput. 71, 316–332 (2011).
27. Zhao, Q., Xiong, C., Zhao, X., Yu, C., Xiao, J.: A data placement strategy for data-intensive scientific workflows in cloud. Proc. - 2015 IEEE/ACM 15th Int. Symp. Clust. Cloud, Grid Comput. CCGrid 2015. 928–934 (2015).
28. Hadoop, http://hadoop.apache.org/, accessed on 10 October 2016