Execution of Large-Scale Scientific Workflows in Cloud Environment: A Comprehensive Survey

Vinay K[1], S M Dilip Kumar[1], Venugopal K R[1]
[1]*University Visvesvaraya College of Engineering*
(E-mail: ec.vinay@gmail.com)

Abstract— Scientific Workflows (SWf) in scientific applications has ever-growing data and computing requirements. Thus, demand a high-performance computing and large data centers respectively. Due to the rise of cloud and application technologies, researchers are designing more sophisticated and useful applications to store, manage and process large scientific data, and execute SWf applications on cloud resources. Therefore, many efforts have been made towards the development of scientific workflow management systems for cloud computing. This paper presents a comprehensive survey of scientific workflows in cloud computing. It addresses numerous key topics, namely, features of SWf, SWf classification, different SWf examples and results, SWf management system, SWf languages, workflow life cycle, SWf parallelism, factors influencing to run SWf on clouds and challenges imposed while running workflow applications. Further, a comparative analysis is presented on the existing SWf solutions and two state-of-art real world case studies are provided. In addition, a comparison of running SWf on clusters, grids and clouds are presented. Finally, SWf open issues and conclusions are provided. The topics covered in this survey could serve as a guidance for the researchers to focus and find solutions to the existing problems of scheduling in cloud computing.

Keywords— cloud computing, scientific workflows, resource provisioning, scheduling, scientific workflow management system

I. INTRODUCTION

Scientists are developing large-scale SWf containing millions of tasks and requiring thousand hours of aggregate computation time. Obtaining the computational resources to run these SWf pose many challenges for application developers [1]. Cloud computing [2] has brought tremendous changes for the scientific community to execute SWf on cloud. However, SWf are concerned with the automation of procedures, whereby files and data are passed between the participants according to a defined set of rules to achieve an overall goal [3]. Workflows are broadly classified into two types, namely scientific workflows and business workflows as shown in Figure 1. SWf are concerned with networks of analytical steps that manage huge and complex distributed computations and are used to contribute in emerging scientific technologies. Usually, SWf processes large amount of data in a high performance computing [4]. On the other side, business workflows (BWf) are mainly used to process the business data.

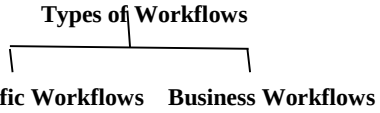

Figure 1: Workflows Classification

Deploying SWf applications on cloud offers several advantages as follows:

1. *Scale of scientific problems:* Cloud computing offers enormous amount of computing resources and storage space for scientific applications, allowing scientists to carry out research on a much larger scale.
2. *Application deployment is flexible due to virtualization technique:* Different environment platforms are pre-loaded or dynamically allocated on virtual machine instances.
3. *On-demand resource allocation leads to better responsiveness:* Workflows computing resources are scalable dynamically in cloud based environment which results in fast turnaround for end users.
4. *Trade-off between performance and cost***:** Reducing hardware cost, increasing in computing power and storage capacity; advent of multi-core architectures and modern supercomputers consisting of hundreds of thousands of cores boost the performance of the task, but at a cost.

Scientific Workflow Management Systems (SWfMS) have been chosen as the vital tool in scientific computing, as it provides several functionalities such as workflow specification, process coordination, task scheduling and execution, monitoring and provenance and fault tolerance. Workflow systems like Taverna, Pegasus, Swift, Kepler, etc., are extensively used in various disciplines, such as bio-informatics, astronomy, physics, earth science, neuroscience, etc. As we are entering into a big data era, it is important for SWfMS to integrate with cloud environment to deal with the ever growing large scale scientific data.

This paper presents a comprehensive survey of SWf on cloud that are gaining a lot of momentum particularly in scientific community. This survey enables one to understand the basic concepts, review of current landscape on SWf on clouds, classification, challenges, and SWf management system. In

addition, a comparative analysis on the existing SWf solutions and two state-of-art real world case studies are provided. A comparison of SWf on clusters and grids with clouds is also presented.

The rest of the paper is organised as follows. Section II provides the motivation about the work. Section III discusses the factors that influence SWf on cloud. In Section IV, we classify scientific workflows based on applications and resources. Section V briefly explains about an overview of generic SWfMS. Section VI present different SWfMS. The challenges imposed by migrating SWf on cloud are discussed in Section VII. Two real world case studies are discussed in Section VIII. Comparisons between cluster, grid and cloud with respect to SWf are presented in Section IX. Section X and XI highlights an open issues and the conclusions.

II. MOTIVATION

Cloud computing has gained a huge impetus in recent times in the field of enterprises, social networks, healthcare, e-governance, engineering, etc. However, despite significant growth in scientific domain, scientific applications often require massive number of computing resources to perform large-scale experiments and acquiring these resources dynamically for computation is a great challenge. Traditionally, computing resources for scientific applications are realised by cluster computing, grid computing and super computers that are difficult to set-up and maintain. Cloud computing is an alternative option that has on-demand resource provisioning capability and pay-as-you-go model. Several scientific applications have already migrated to cloud platform that has provided research opportunities in solving complex scientific problems with regard to resource provisioning and scheduling.

Moreover, these applications are compute and data intense and require mass storage to store scientific data and high computing resources to process.

There are many cloud service providers available such as Amazon web services (EC2, S3, EBS), Windows Azure, Google cloud platform, IBM Cloud, Nimbus cloud, etc. that facilitates running scientific workflow applications in a cost effective manner.

Scientific applications like Montage from astronomy, Epigenome from bio-chemistry, Broadband and Cybershake from seismology are serving the society by regularly analyzing and reporting the events in scientific fields. Henceforth, scientific users will be greatly benefited by adopting cloud environments in solving complex scientific problems. Few works have focused towards providing a comprehensive survey that provides a broad understanding of the SWf and migrating SWf on clouds. Motivated by these factors, an extensive review on current state-of-the-art approaches to empower the scientific community is presented in this paper.

III. FACTORS INFLUENCING SWF ON CLOUD

In this section, we illustrate the factors that influence SWf to run in cloud environments. The features of cloud computing that supports workflows are discussed in this section.

A. Resource Provisioning and Scheduling

Resource provisioning and scheduling for SWf on cloud occurs in two stages. In the first stage, the computing resources that are required to run workflows are dynamically provisioned and in the second stage, a schedule is generated and each workflow is mapped onto a best-suited resource. In the earlier works, especially those developed for grids or clusters concentrated only on scheduling processes, reason being that these environments provide a static pool of resources that are readily available to execute workflows and whose configuration is known in advance. Cloud environments present a different paradigm, on completion of the execution; all the resources provided are de-provisioned resulting in efficient resource utilization, cost effectiveness and improve workflow performance. In addition, there is a need for dynamic allocation of resources as per the requirement

B. Load Balancing

Load balancing in SWf is the process of distributing the tasks among various resources in any system for execution. Thus tasks need to be distributed to the resources in cloud computing, so that individual resource does approximately the same amount of task at any given point of time. SWf applications are mostly compute-intensive, effective balancing of virtual machines with respect to tasks are handled in cloud based environments. The primary objective of load balancing is to increase performance, prioritize SWf tasks and ensure cost effectiveness.

C. Elasticity

Elasticity is an important feature of cloud computing and is suitable for running scientific workflow applications because the resources required are dynamically increased. Elasticity in cloud computing aims at matching the amount of resources required for workflows with the amount of resources it actually requires. Instead of targeting physical size of the system in advance, cloud environment dynamically reacts to actual load by adding new virtual resources. Current research in execution of SWf in clouds either try to reduce the workflow execution time neglecting deadlines and budgets or concentrate on minimizing the cost while trying to meet the application deadline.

D. Availability

Cloud computing offers the availability of a massive number of computers for execution of scientific workflow applications. Traditionally, these requirements are fulfilled by using High Performance Computing (HPC) and fixed facilities such as super computers, clusters and grid, that are difficult to install, operate and maintain. Scientific workflow users can concentrate on workflow execution and need not be concerned

about the availability of resources as these resources are dynamically provisioned.

E. Legacy Code

SWf which are designed and developed using traditional programming languages are considered to be legacy applications. These applications are sometimes challenging to change codes that are developed and tested long back in worry of bugs. Which appear and in turn affect the results of scientific experiments. These kind of applications may be platform dependent and such applications can be installed and executed in cloud environments due to the use of virtualization techniques.

F. Fault Tolerance

Fault tolerance in SWf is crucial, it guarantees reliability and availability of services until the end of SWf execution. Current SWfMSs exception handling, fault tolerance, and recovery issues are grouped and handled together. SWf fault tolerance happens at task-level, workflow-level, during migration and retrying. However, different SWfMS have different recovery mechanisms and checkpoints to avoid unnecessary mechanisms.

G. Security

Despite of the fact that cloud computing has several advantages but security is a major area of concern to deal with. The use of SWfMSs on cloud for scientific applications raises major security concerns regarding the threats against integrity, authorization, authentication, availability, etc. The conception of running secure workflow instances on public cloud platforms is still in its infancy Cloud service providers such as Amazon Web Services, Microsoft Windows Azure, Rackspace hosting, etc. provide tools and guidelines for enhanced security for scientific applications on cloud environments. Adopting the third party cloud service providers and executing the workflows, a trust required and should meet the scientific workflow users. A trust is required based on the selected cloud service provider, as the governance of information and applications are outsourced and it's delegated from owner's control.

IV. SCIENTIFIC WORKFLOWS CLASSIFICATION

A. I/O Intensive

Some of the scientific workflow applications are I/O-intensive that reads and/or writes large amount of data and their performance depends on the computing resources being used. I/O intensive applications produce large amount of output due to which a deliberation is required between transfer and storage cost. Scientific users can transfer input data to individual SWf for execution, or transfer data at once, and store them in the cloud for multiple runs. Normally, storage is cost-effective for input data that is reused often and accessed frequently, and transfer is cost-effective only if data is used once. The Montage7 application from astronomy which is an I/O bound application has over 10,429 tasks and reads 4.2 GB of input data and produces 7.9 GB of output data. Therefore, these SWf applications are cost effective to store the input data rather than transfer the data for each workflow [5].

B. Compute Intensive

Workflows whose job compute times dominate file transfer times are termed as compute-intensive [6]. Compute-intensive workflow utilizes CPU resources of cloud extensively as compared to other resources such as I/O, memory, etc. Dynamic allocation of CPU resources are very much needed for these kind of applications to continuously execute. To illustrate compute-intensive workflow, let us consider the Epigenome from biochemistry which has 81 tasks and reads 1.8 GB of input data and produce 300 MB of output data. Compute-intensive SWf is CPU-bound because it spends 99% of its runtime in the CPU and only 1% on I/O and other activities.

C. Data Intensive

Data-intensive SWf involves accessing data, processing it and transfer large amount of datasets that are replicated on different hosts. In order to minimize the time to transfer these datasets for execution requires suitable data and computational resources. Due to emergence of data-intensive SWf in scientific and enterprise scenarios, there is a need for scheduling these workflows efficiently and it comprises of jobs that are data dependent and these are executed as part of a functional unit with other jobs. The data volumes of these workflows can increase rapidly, with the accumulated data size of each application and is expected to reach 1 million Terabytes [21]. Amazon Web Services provides different storage services like Amazon S3, Amazon EBS and Amazon Glacier. Amazon S3 provides cost-effective object storage for a number of use cases for data-intensive workflow applications.

D. Memory Intensive

Memory-intensive workflows are those applications in which primary memory usage is extensive. Results of a task might be required for next task as input. Therefore, the output of one task is preserved in cache memory and fed as an input for another task in succession. Consider the Broadband from seismology has 320 tasks, which reads 6 GB of input data, and produce 160 MB of output data every day. The task is memory-limited because more than 75% of its runtime is utilized by tasks requiring more than 1 GB of physical memory.

E. Instance Intensive

Instance-intensive SWf are not adequately researched in comparison with other types of workflows. Instance-intensive SWf consists of large number of concurrent workflow instances, usually much simpler than those complex SWf enabled on a cloud computing environment [7]. These workflows throughput is important rather than decreasing execution time of a single instance. However, scheduling these instance-intensive workflows in cloud computing is a challenging issue that needs to be addressed.

II. OVERVIEW OF GENERIC SWFMS ON CLOUD

Figure 2 shows an overview of generic SWfMS on cloud, and also illustrates the entire process of SWf applications from an abstract representation to an actual execution in the cloud environment. SWfMS provides platform for scientists to model, design, execute and re-run SWf. The application components in cloud environments consists of workflow mapper, clustering engine, workflow engine, workflow scheduler, monitoring and provenance and resource manager. These components are discussed below:

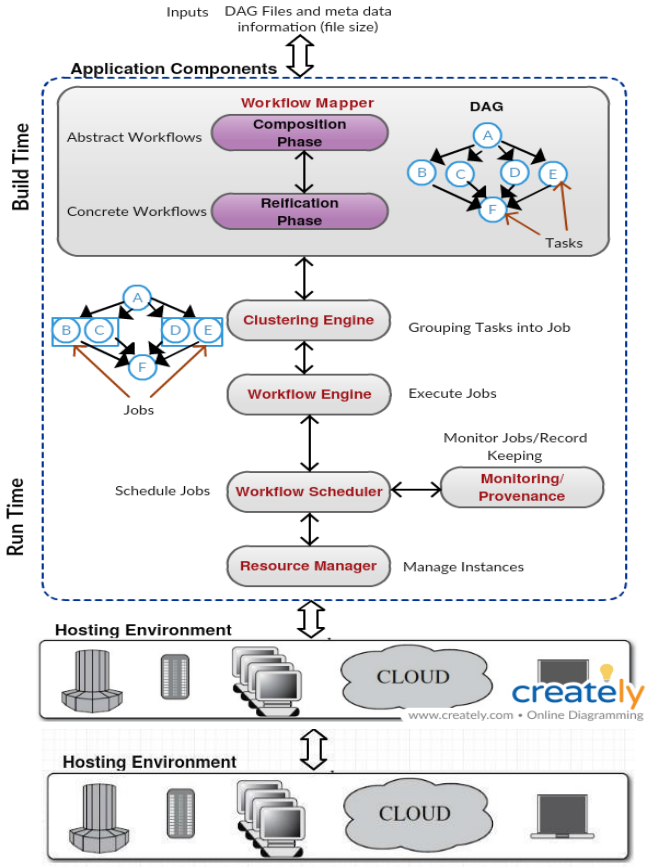

Figure 2: Overview of Generic SWfMS on Cloud

A. *Workflow Mapper*

Workflow mapper job is to import Directed Acyclic Graph files (DAG) and meta-data information such as file size [8]. Further it consists of two phases, namely composition phase and reification phase. Composition phase corresponds to specification of input and output data, and computational tasks. The scientific user design and compose SWf using Abstract Workflow Description Language (AWDL). This SWf cannot be executed until it is reified. In reification phase, SWfMS compiles the SWf which are abstract workflows written in AWDL. After compilation it is converted to concrete workflows, which is an information required for SWf execution. Later, concrete workflow which is an executable file, is executed as it contains all the necessary information. In distinction to abstract workflows, the concrete workflows is implementation dependent and each activity is connected with a specific computational activity. Finally, workflow mapper make a list of tasks and assigned to next phase called clustering engine.

B. *Clustering Engine*

The clustering engine combines tasks into jobs as shown in Figure 2. Assume A, B, C, D, E and F are tasks to be executed. B and C are clustered into jobs to execute in parallelism. A job is a single unit seen by the execution system, which consists of multiple tasks to be executed in sequence or in parallel. Task clustering [9] is a runtime optimization technique that merges multiple short tasks into a single job such that the scheduling overhead is reduced and the overall runtime performance is improved [10]. But in SWf, tasks within a level may have different execution times. Merging SWf tasks within a level without considering the runtime variance might result in load imbalance. Due to which there is a delay in release of tasks from the next level of workflow [11]. To overcome this load imbalance, an over-decomposition method [12] is applied which decomposes workflows into coarse grained tasks resulting in efficient execution and reduce scheduling overheads.

C. *Workflow Engine*

The workflow engine manages the jobs and its dependencies in SWf to ensure job is terminated only when all of its parent nodes are completed its execution. And also, the performance of workflow engine depends on the computing resources. Further, a slight delay is introduced to ensure the SWf task scheduling and execution are not overloaded. [10].

D. *Workflow Scheduler*

The scheduling of interdependent tasks of SWf are managed by workflow scheduler in distributed environment. And also, a delay is introduced between the tasks in worker node to improve an efficiency of job scheduler based on the availability of resources [10]

E. *Monitoring and Provenance*

This component provides an interface for scientific users to monitor workflow execution and metric functions to facilitate the management of composite workflow application environments. In scientific applications, provenance means the storage of meta-data which depicts about computation details to solve origins and derivation of data generated during execution [13]. The provenance of SWf is an essential and supportive component that delivers the knowledge sharing, product reusability, and process verification [14].

F. *Resource Manager*

Resource manager will negotiate, reserve and allocate resources that are required for execution of SWf applications on cloud [15]. It also manages Virtual Machines for different SWf. Resources are deallocated and made available for successive workflows on completion of the workflow.

III. DIFFERENT SCIENTIFIC WORKFLOW MANAGEMENT SYSTEM

This section presents a features of SWfMS and a number of workflow management systems especially designed for SWf applications, and Table 1 compares different scientific workflow management systems with respect to scientific domains, structures, user interface type, scheduling and open source.

A. Features of SWfMS

1. *Maps abstract workflows to executable workflows:*

In the workflow life cycle, abstract workflows (high-level abstraction) is constructed which identifies the components of SWf applications and the data required without the details of physical resources. An abstract workflow is mapped into an executable plan, called a concrete workflow. Concrete workflows describes the resources required for SWf execution. A better mapping enhances the resource-usage efficiency and performance.

2. *Handles data dependencies*:

Running scientific workflow applications on cloud require high performance computing resources and massive storage. Scientists need to analyse tera bytes of data either from the existing data resources or collected from physical devices. The data generated during these processes might have similar amounts of new data as final products [16]. Data dependency is considered while rerunning, and executes only part of the workflows that are affected by the change in parameter and thus reduce the execution time.

3. *Processing Elements*

Elements in workflows are software components, these components may be written in a Java class, Phyton script, etc. The software components are standalone applications that receives input from various sources, process it, and produces desired output. Workflow is created by combining these software components together. SWfMS is responsible for pipelining by fetching output from previous components and feeding them as input to successive components [17].

4. *Data Processing Model*

Data processing model is categorised into single input and data stream. In a single input, the processing elements read a single dataset at a time, and produces single output, and in data stream data arrives in continuous, multiple and time-varying data streams. In data stream, processing elements read a part of data item from the data stream and produce a part of output data [18].

5. *SWf Constructive Approach*

There are mainly two approaches to construct SWf: top-down and bottom-up. In top-down approach SWf is described first and then right mapping is done to execute programs. If there is no such suitable program, then users have to develop and include them into mapping repository [17]. In bottom-up approach, individual program is developed for each task and user need to describe the workflow execution making use of high level description language. For example, in Swift SWfMS, an application is developed and added in transformation catalogue and used in implementing Swift program.

6. *Optimization Stage*

Optimization for SWf can be achieved in stages. First stage is the build time, in which resource mapping and composition takes place. Second stage is runtime, in which execution and monitoring phase takes place, and is managed by workflow execution engine. The execution engine gets the live status of computing resources utilized and optimise the task scheduling. Workflow optimisation are performed at both the mapping phase (build time) and execution phase (run time). The optimisation approaches might vary depending on processing elements, i.e., web services or executable programs [17].

SWfMs	Scientific Domains	Structures	Scheduling
Pegasus	Astronomy, Biology	DAG	Static/Dynamic
Taverna	Biology	DAG	Dynamic
Triana	Astronomy	DCG	Dynamic
Kepler	Biology, Ecology	DAX	Dynamic
Galaxy	Bioinformatics	DCG	Dynamic
Askalon	Bioinformatics	DCG	Dynamic/Hybrid
Swift	Biology, Chemistry	DCG	Dynamic
VistTrials	Astronomy	DAG	Dynamic
Airavata	Biology, Chemistry	DAG	Dynamic
Discovery Net	Biology, Chemistry	DAG	Dynamic
Karajan	Oceanography	Non-DAG	Dynamic

Table 1: Comparison of Different SWfMs

IV. CHALLENGES OF SCIENTIFIC WORKFLOWS IN CLOUD COMPUTING

Despite cloud computing opens many advantages and opportunities for running SWf, there are several challenges to be faced while running SWf on the cloud. In this section, we briey describe major challenges of SWf on clouds and their impact on SWfMS in cloud environments.

A. Architectural Challenges

The system architecture for SWf on cloud should follow the general architecture of cloud computing, but at the same time, it also needs to be adapted according to different system requirements [19]. The reference architecture [1] consists of 4 layers, namely operational layer, task management layer, workflow management layer and presentation layer. The separation of operational layer from other layers' isolates data sources, software tools and other associated computing environments. To integrate SWfMS with cloud environment, it is not as easy as to exchange the operational layer with a cloud infrastructure [20]. A bottom-up approach is required to evaluate the requirements; cloud service providers have to rethink in order to implement a typical architecture for SWf applications on cloud.

B. Integration Challenges

A change is required as how traditional SWfMS acquires resources, dispatch tasks, monitors the progress of tasks, etc. [20]. Integrating SWfMS with cloud and considering SWf applications, scripts as web services is suitable for small data transaction, movement of large datasets in and out from cloud might induce serious overhead [1]. However, integrating SWf applications with third party cloud services is one of the biggest challenge even though the resources in cloud is tremendous [21]. To extensively explore the scalability of cloud, a workflow engine in SWfMS needs to be re-engineered to communicate directly with diverse cloud services such as storage, monitoring, resource allocation, task scheduling, etc.

C. Computing Challenges

For SWf, leveraging large-scale computational resources in the cloud is not as straightforward as requesting a certain number of computing nodes [1], there are certain challenges with respect to resource provisioning and scheduling due to overheads introduced by different virtualization techniques which might not be widely accepted by the scientific community. Fault tolerance and reruns are the key features of SWf, if these features are not handled properly then it affects the computation of scientific applications. Due to scale of SWf there is a need of more components such as VMs to compute. An extra measure is required to support for large-scale scientific problems.

D. Data Challenges

Current SWf applications are becoming more data intensive, the execution of SWf usually consume and generate large amounts of data objects. These data objects are either primitive or complex types, files in different formats and sizes, database tables, or data objects in other forms. Currently the scientific community is facing a “data deluge" [22] coming from experiments, networks, simulations, sensors, and satellites, and the data that needs to be processed are growing faster than computational resources required for execution. On the other part, these large scale data needs to be stored in distributed data centers. In order to store these datasets effectively, an appropriate data centers should be chosen by a data manager. When single task requires several datasets which are located in different data centers, then the transfer of large datasets becomes a challenge [23].

E. Language Challenges

There were a different coordination languages and systems developed during 80's and 90's. Different workflow management system supports different workflow languages, there is no uniformity in language to choose and execute. No matter in which language we choose to run SWf, all the SWf languages should meet the challenges of mapping the input-data and output-data into logical patterns to enable data integration. Further, it should encourage for large-scale parallelism in order to execute faster and clearly partition data and task so that scale of computation and data processing is not effected. MapReduce technique has been the only model adopted for clouds to perform computations with large scale scientific datasets till recently. There is a need of some more languages that can be used for computations on cloud.

F. Heterogeneity Challenges

Heterogeneity in cloud for SWf exists in different forms. Cloud service provider uses different hardware and software resources to build cloud environment and there is a vertical heterogeneity of cloud resources, suppose a client subscribe to an IaaS from one provider, couple it with a workflow management system from different vendor and running SWf upon that. Now a day’s scientific research projects are becoming more collaborative in nature and involve different distributed organizations which results in heterogeneity challenge to both scientists and application developers.

G. Resource Selection Challenges

Resource allocation in cloud computing mainly deals with scheduling tasks and the required resources while considering both resource availability and project time. Every resource has three features, namely execution speed, resource ID, and execution cost [24]. SWf consists of various tasks and these tasks can be of atomic or multi-instance. Running these tasks in cloud involves choosing the appropriate number and type of virtual machines to execute. It is hard to determine optimal scheduling and hence it is a challenging to choose different configured virtual machines for a given workflow.

H. Security Challenges

Although adequate amount of research being done on SWf security on cloud, but it is in initial stage. Access control, information flow control and secure electronic transaction protocols are the three mechanisms which result in security challenge. Access control deals with which scientific users have privileges to access resources such as SWfMS services, SWf, tasks, provenance and record keeping, datasets, etc. Since SWf data are critical and it should not be disclosed to unauthorised users. A secure electronic transaction protocol is introduced to ensure atomicity. Further research has to be done to ensure the security of cloud-based transactions [1].
Researchers can focus on the above mentioned challenges and provide an optimal solution towards the future development of SWf in cloud environments.

V. REAL WORLD CASE STUDIES

The cloud environment provides a convenient platform for executing SWf. The cost of running SWf applications on cloud is relatively small, however the cost might vary significantly based on multiple virtual nodes that are required to run larger workflows. Here, we discuss two case studies which depicts about SWf on cloud.

A. Scientific Workflows on Amazon EC2

Gideon Juve et al. [25] performed experiments on three real SWf applications on cloud namely Montage which belongs to astronomy domain, Broadband which is a seismology

application and Epigenome, a bioinformatics application. These three are chosen because it covers a wide range of domains. Table 2 presents the maximum resource required by the Montage, Broadband and Epigenome applications in three different categories: I/O, memory and CPU. The resource usage of the applications is measured using a SWf profiler, which measures the I/O, CPU and memory usage.

Application	I/O	Memory	CPU
Montage	High	Low	Low
Broadband	Medium	High	Medium
Epigenome	Low	Medium	High

Table 2: Application Resource Usage Comparison

1. *Montage*

Montage [26], generates science grade astronomical images that are collected from ground based telescopes. The footage of this SWf depends on the area of the sky which are measured in square degrees by the mosaic output. Gideon Juve et al. [25] configured Montage workflow to produce an 8 degree square mosaic. This workflow contains nearly 10,429 tasks and reads 4.2 GB of input and generates 7.9 GB of output. Therefore this workflow is an I/O bound as it spends most of the time (95%) waiting on I/O operations.

2. *Broadband*

In Broadband seismograms are generated from many high and low frequency earthquake simulation codes and the result of these are analysed to know about the intensity of earthquake. Gideon Juve et al. [25] used 6-sources and 8-sites to produce a workflow which contains nearly 768 tasks. It reads 6 GB of input and generates 303 MB of output. Broadband is memory-intensive because more than 75% of its runtime is consumed by tasks requiring more than 1 GB of physical memory.

3. *Epigenome*

Epigenome maps short-DNA segments measured from high throughput gene-sequencing machines to a preceding generated reference genome using the MAQ software. Epigenome splits various input segments into small chunks and maps these chunks to the reference genome. These chunks are executed in parallel and generate a single output. The workflow maps human DNA sequences from 21 chromosome. The Epigenome contains 529 tasks, reads 1.9 GB of input data, and produces 300 MB of output data. Epigenome to be a CPU-bound because it spends 99% of its runtime in the CPU and only 1% on I/O and other activities.

B. BioVLab: Virtual Collaborative Lab for Bio/medical on Amazon EC2

BioVLab is a cloud workbench for the integration analysis, and increased computing infrastructure to hold large data sets. BioVLab utilizes Amazon EC2 and S3 cloud services and XBaya as a graphical client program for workflow composer. The main aim of the BioVLab is to develop a noval system for genome analysis and provide a platform for small biology research to deal with ever growing biological data. BioVLab is a three layered architecture, the first layer is Xbaya graphical workflow engine that facilitates the composition of SWf. The second layer is a gateway that registers the SWf and monitor the workflow execution. The third layer is a cloud based environment that hosts applications, data sets and run time environment. In BioVLab, two systems were developed BioVLab-MMIA (MicroRNA and MRNA Integrated Analysis) [27] and BioVLab-mCpG and are discussed as follows:

1. *BioVLab-MMIA*

It is a web based application for the analysis of both microRNA and mRNA data. The complexity arises from two different genetic element types. This BioVlab-MMIA uses inversely correlated expression patterns between microRNA and mRNA and record the results. These results are later uploaded to Amazon S3 for analysis.

VI. COMPARISON OF SCIENTIFIC WORKFLOWS ON CLOUD WITH CLUSTER AND GRID COMPUTING

Scientific applications are executed in different computing infrastructures such as cluster, grids and clouds. We compare and contrast the infrastructures of cluster, grid and cloud which runs SWf applications with respect to several parameters as presented in Table 3.

Parameters	Clusters	Grids	Clouds
Set Up	More time consuming	Less time Consuming	Very less time consuming
Service Oriented	No	Yes	Yes
Usability	Low	Medium	High
Loose Coupling	No	Half	Yes
SLA	Limited	High	High
Business Model	No	No	Yes
Pricing	Allocation Model	Allocation Model	Pay-as-you-go model
Resource Availability	Less	Medium	High
Strong Fault tolerant	Half	Half	Yes
Performance	Medium	Medium	High
TCP/IP based	No	Half	Yes
Virtualization	Medium	Medium	High
Multi-tenancy	No	Yes	Yes
Failure Rate	High	No	No
Security	Low	Medium	No
Resource Management	Centralised	Distributed	Centralized/Distributed

Table 3: Comparison of clusters, grids and clouds which runs scientific workflows

VII. OPEN ISSUES

This section broadly presents some of the open issues to be addressed and addressing these issues are vital for the development of algorithms for SWf on cloud.

A. Architectural issues

The scientific applications on cloud need to adopt the general architecture of cloud computing. A generic architecture may not be sufficient when market opposition enforces business

policies for cloud providers. Several issues needs to be addressed with regard to reference architecture.

B. Scheduling Dependent Tasks

In scientific workflows, tasks are usually dependent and the dependencies are both control and data dependencies. The consideration of these two types of dependencies are equally important during scheduling. Efficient strategies are required to schedule these dependent tasks in order to achieve better task runtime and minimum task failure rate.

C. Quality of Service

In cloud computing, scientific users need to gain access to resources that are located in the cloud. However, scientific users may face difficulties with respect to network disconnection and congestion due to limited bandwidth. Hence, Quality of Service (QoS) degrades drastically. Moreover, a huge number of scientific applications are migrating to cloud environments and that needs to be resolved to achieve better QoS.

D. Security and Privacy Issues

SWf applications are bound to be data-intensive and preserving these sensitive data on cloud is of great challenge. Providing authorized access and constructing a secure environment is an open issue. Since SWf data are critical, it should not be disclosed to unauthorised users. Secure electronic transaction protocols need to be introduced to ensure atomicity. Further research has to be done to ensure the security of cloud-based transactions.

VIII. CONCLUSIONS

Cloud offers remarkable opportunities to solve large-scale scientific problems by providing a seamless and rich infrastructure on demand. In this paper, we have presented a comprehensive survey of SWf especially in the cloud. In the opening, the details of the key concepts related to SWf on clouds are explained so as to better provide the basis to understand this survey work. Different types of workflows and their differences are discussed. Then, we have discussed the factors that influence SWf to run on cloud environments. This is followed by a broader classification of SWf, namely I/O intensive, compute intensive, data intensive, memory intensive and instance intensive. The interesting features of scientific workflow management system are discussed and an overview of generic SWfMS for clouds is given. Different SWfMS and their comparison with respect to various parameters is presented. Cloud computing opens many advantages and opportunities for running SWf but there are several challenges that need to be addressed. We present the scientific workflow challenges related to clouds. We have also identified some of the popular projects on SWf and two real world case studies are presented.

Finally, we compare scientific workflows on clusters, grids and clouds with various parameters. This survey provides an evidence of interest towards SWf on clouds. As this area of research matures, it is anticipated to perceive more design of algorithms to match the specific requirements. In conclusion, SWf can be deployed on cloud and enable the scientific community to automate the workflow execution in an efficient manner.

Acknowledgement

We would like to thank the following contributors of scientific workflow systems: Ewa Deelman, Gideon Juve, Zhao Yong, Jia Yu, Rajkumar Buyya, R N Calheiros, Domenico Talia, Bruce Berriman, Mats Rynge, Ian Taylor, Matthew Schields, Ji Liu, Yong Zhao, Shiyong Lu Chee Sun Liew, Ke Liu and the Workflow Management Coalition Technical Report WFMC-TC-1011.

References

[1] Y. Zhao, Y. Li, I. Raicu, S. Lu, W. Tian, H. Liu, "Enabling Scalable Scientific Workflow Management in the Cloud," Future Generation Computer Systems, Elsevier, vol. 46, pp. 3-16, 2015

[2] Foster Ian and Zhao Yong and Raicu Ioan and Lu. Shiyong, "Cloud computing and Grid Computing 360-degree Compared," Grid Computing Environments Workshop, pp. 1-10, 2008

[3] Yu, Jia and Buyya, Rajkumar, "A Taxonomy of Workflow Management Systems for Grid Computing," Journal of Grid Computing, Springer, vol. 3, pp. 171-200, 2005

[4] Gil, Y. and Deelman, E. and Ellisman, M. and Fahringer, T. and Fox, G. and Gannon, D. and Goble, C. and Livny, M. and Moreau, L. and Myers, J, "Examining the Challenges of Scientific Workflows", IEEE Computer, vol. 40, pp. 12-20, 2007

[5] G. B. Berriman, G. Juve, E. Deelman, M. Regelson, P. Plavchan, The application of cloud computing to astronomy: A study of cost and performance, in: Sixth IEEE International Conference on e-Science Workshops, IEEE, 2010, pp. 1-7.

[6] Y. Zhao, Y. Li, I. Raicu, S. Lu, W. Tian, H. Liu, "Enabling Scalable Scientific Workflow Management in the Cloud," Future Generation Computer Systems, Elsevier, vol. 46, pp. 3-16, 2015.

[7] J. Blythe, S. Jain, E. Deelman, Y. Gil, K. Vahi, A. Mandal, K. Kennedy, Task scheduling strategies for workflow-based applications in grids, in: IEEE International Symposium on Cluster Computing and the Grid (CCGrid), Vol. 2, IEEE, pp. 759-767, 2005.

[8] T. Kosar, M. Livny, Stork: Making data placement a _first class citizen in the grid, in: Proceedings of 24th International Conference on Distributed Computing Systems, IEEE, pp. 342-349, 2005.

[9] K. Liu, Scheduling algorithms for instance-intensive cloud workflows, Swinburne University of Technology, Faculty of Engineering and Industrial Sciences, Centre for Complex Software Systems and Services, 2009.

[10] W. Chen, E. Deelman, Workflowsim: A toolkit for simulating scientific workflows in distributed environments, in: IEEE 8th International Conference on e-Science, IEEE, pp. 1-8, 2012.

[11] W. Chen, R. F. da Silva, E. Deelman, R. Sakellariou, Using imbalance metrics to optimize task clustering in scientific workflow executions, Future Generation Computer Systems 46 (2015) 69-84.

[12] W. Chen, R. Da Silva, E. Deelman, R. Sakellariou, Balanced Task Clustering in Scientific Workflows, in: IEEE 9th International Conference on eScience, IEEE, 2013, pp. 188-195.

[13] W. Chen, E. Deelman, R. Sakellariou, Imbalance optimization in scientific workflows, in: Proceedings of the 27th international ACM conference on International conference on supercomputing, ACM, 2013, pp. 461-462.

[14] J. Lifflander, S. Krishnamoorthy, L. V. Kale, Work stealing and persistence-based load balancers for iterative over decomposed applications, in: Proceedings of the 21st international symposium on High-Performance Parallel and Distributed Computing, ACM, 2012, pp. 137-148.

[15] G. Juve, E. Deelman, Scientific workflows and cloud, Crossroads-Plugging into the Cloud 16 (3) (2010) 14-18.

[16] Y. Li, O. Boucelma, Provenance Monitoring in the Cloud, in: IEEE 6th International Conference on Cloud Computing, IEEE, 2013, pp. 802-809.

[17] T. Fahringer, R. Prodan, R. Duan, J. Hofer, F. Nadeem, F. Nerieri, S. Podlipnig, J. Qin, M. Siddiqui, H.-L. Truong, et al., Askalon: A development and grid computing environment for scientific workflows, in: Workflows for e-Science, Springer, 2007, pp. 450-471.

[18] E. Deelman, A. Chervenak, Data management challenges of data-intensive scientific workflows, in: 8th IEEE International Symposium on Cluster Computing and the Grid (CCGRID), IEEE, 2008, pp. 687-692.

[19] C. S. Liew, Optimisation of the enactment of _ne-grained distributed data-intensive workflows.

[20] B. Babcock, S. Babu, M. Datar, R. Motwani, J. Widom, Models and issues in data stream systems, in: Proceedings of the twenty-_first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, ACM, 2002, pp. 1-16.

[21] E. N. Alkhanak, S. P. Lee, S. U. R. Khan, Cost-aware challenges for workflow scheduling approaches in cloud computing environments: Taxonomy and opportunities, Future Generation Computer Systems 50 (2015) 3-21.

[22] Y. Zhao, X. Fei, I. Raicu, S. Lu, Opportunities and challenges in running scientific workflows on the cloud, in: International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), IEEE, 2011, pp. 455-462.

[23] G. Bell, T. Hey, A. Szalay, Beyond the data deluge, Science 3 (5) (2009) 1297-1298.

[24] D. Yuan, Y. Yang, X. Liu, J. Chen, A data placement strategy in scientific cloud workflows, Future Generation Computer Systems 26 (8) (2010) 1200-1214.

[25] Z. Wu, X. Liu, Z. Ni, D. Yuan, Y. Yang, A market-oriented hierarchical scheduling strategy in cloud workflow systems, The Journal of Supercomputing 63 (1) (2013) 256-293.

[26] G. Juve, E. Deelman, G. B. Berriman, B. P. Berman, P. Maechling, An evaluation of the cost and performance of scientific workflows on amazon ec2, Journal of Grid Computing 10 (1) (2012) 5-21.

[27] D. S. Katz, J. C. Jacob, E. Deelman, C. Kesselman, G. Singh, M.-h. Su, G. Berriman, J. Good, A. Laity, T. A. Prince, A comparison of two methods for building astronomical image mosaics on a grid, in: International Conference Workshops on Parallel Processing (ICPP), IEEE, 2005, pp. 85-94.