Available Online at http://www.recentscientific.com

CODEN: IJRSFP (USA)

International Journal of Recent Scientific Research
Vol. 8, Issue, 6, pp. 17335-17338, June, 2017

ISSN: 0976-3031

International Journal of Recent Scientific Research

DOI: 10.24327/IJRSR

Research Article

IMPLEMENTATION OF DATA MINING CLOUD FRAMEWORK ALGORITHM FOR CREATING AND MANAGING WORKFLOWS

***Divya C., Satish Kumar T and Krishna Prasad R**

Department of Computer Science Engineering, Global Academy of Technology,
Bengaluru, India

DOI: http://dx.doi.org/10.24327/ijrsr.2017.0806.0333

ARTICLE INFO

Article History:

Received 20th March, 2017
Received in revised form 29th April, 2017
Accepted 30th May, 2017
Published online 28th June, 2017

Key Words:

Data Analysis Workflow, DMCF, workflow Language.

ABSTRACT

The most complex process is to collect huge amount of beneficial information from available data. But this can be made convenient by modelling a Data Analysis Workflow. To analyze the large data sets, Data Analysis Workflows may consume more time to execute completely. This situation is the same even when complex data mining Algorithms are used. Hence to make execution of Data Workflow Analysis more scalable for efficient systems, there is a need to make exploit the computing services of the cloud structures where in information is increasingly being saved. This Paper presents a method to design and implement a Data Mining Cloud Framework (DMCF), which integrates the Data Workflow Language and parallel runtime SaaS model. The main goal of DMCF is to simplify the development of data mining Algorithms keeping Real Data Mining Applications in mind.

INTRODUCTION

Long-ago mankind has witnessed that there is an incremental growth of digital records manufactured in many field. Very huge datasets are produced every day from sensors, devices, mobile gadgets and computer systems, and are frequently saved in disbursed repositories. As an instance, astronomers examine huge photo records that every day comes from telescopes and artificial satellites. Physicians should have a look at the massive quantity of information generated with the aid of particle accelerators to recognize the laws of Universe. Medical docs and biologists accumulate massive amount of records about patients to go looking and try to apprehend the causes of diseases. Such examples display how the exploration and automated analysis of big datasets powered by using computing talents are essential to get better awareness in lots of fields.

It is difficult to discover and know massive datasets, and above all models and patterns hidden in them can't be understood neither by humans directly, nor by ancient analysis methodologies [1]. To address massive information repositories, parallel and distributed information analysis techniques should be used. In fact, usually professionals and scientists use information analysis environments to execute complicated simulations, validate models, compare and share results with colleagues placed world-wide.

Related works

Various Models have been proposed to Design and Execute Workflow-based Applications.

Galaxy [2] is a web-based platform for developing genomic science applications, now used as a general bioinformatics workflow management system. A Galaxy workflow is a reusable template that a user can run repeatedly on different data. The basic working unit in Galaxy is a "tool". Tools run as custom execution units for existing program interpreters or "runners" such as shell, perl and python. As a first integration scheme, we developed a generic tool with a capability of execution of user provided arbitrary Swift scripts [3].

Taverna [4] is a workflow management system mostly used in the life sciences community. Taverna can orchestrate Web Services and these may be running in the Cloud, but this is transparent for Taverna, as demonstrated in the BioVel project. Most Taverna workflows are composed from a mixture of distributed Web Services, local scripts and other service types. Orange4WS [5] is a service-oriented workflow system which is an extended version of Orange, a data mining toolbox and a visual programming environment for the visual composition of data mining workflows.

Corresponding author:* **Divya C
Department of Computer Science Engineering, Global Academy of Technology, Bengaluru, India

Kepler [6] is a visual workflow management system that provides a graphical user interface for designing scientific workflows. Kepler includes a graphical user interface for composing workflows in a desktop environment, a runtime engine for executing workflows within the GUI and independently from a command-line, and a distributed computing option that allows workflow tasks to be distributed among compute nodes in a computer cluster or computing grid. E-Science Central (e-SC) [7] allows scientists to store analyze and share data in the Cloud. It's in browser workflow editor allows users to design a workflow by connecting services, either uploaded by themselves or shared by other users of the system. One of the most common use cases for e-Sc is to provide a data analysis back end to a standalone desktop or Web application.

ClowdFlows [8] is a Cloud-based platform for the composition, execution, and sharing of interactive data mining workflows. Its service-oriented architecture allows using third-party services. Its service-oriented architecture allows using third-party services.

Pegasus [9] includes a set of technologies to execute workflow-based applications over clusters and Grids. The system can manage the execution of an application formalized as a visual workflow by mapping it onto available resources. The system can manage the execution of an application formalized as a visual workflow by mapping it onto available resources.

ASKALON [10] is a Cloud-based application development environment designed as a distributed service-oriented architecture. Users can compose workflow using a UML graphical modeling tool.

Proposed Method

Workflow Management System is used for proper management of executing the workflow tasks on the computing resources. The method introduced here provides an idea to describe the application.

Workflow contains tuple like *workflow identifier* which is a unique ID generated for that particular workflow, *customer ID* is an identifier of the customer who uses the workflow, *Status* represents whether workflow is running, done, ready, or failed, and *List* contains the list of task which forms the workflow

Task is modelled as a tuple which contains *Task identifier*, *workflow identifier* which is a unique ID generated for that particular workflow to which the task belongs to, *Status* represents whether task is new, ready, running, done or failed, and dependence List contains the ID of the other tasks if this task depends on.

Tool contains tuple like *Tool Identifier*, *name* which describe name of the tool, *executable* is refers to the tools that are executed, *librarylist* contains all the required libraries for the particular task, *parameter List* contains list of parameters like input and output data.

Parameter contains tuple like *Parameter Name*, *description* of the parameter, *Type* of the parameter whether it is an input or output.

System Design

The Architecture can be organized as six modules. They are as follows:

Data Access Layer

Data access layer is the one which exposes all the possible operations on the data base to the outside world. It will contain the DAO classes, DAO interfaces, POJOs, and Utils as the internal components. All the other modules will be communicating with the DAO layer for their data access needs.

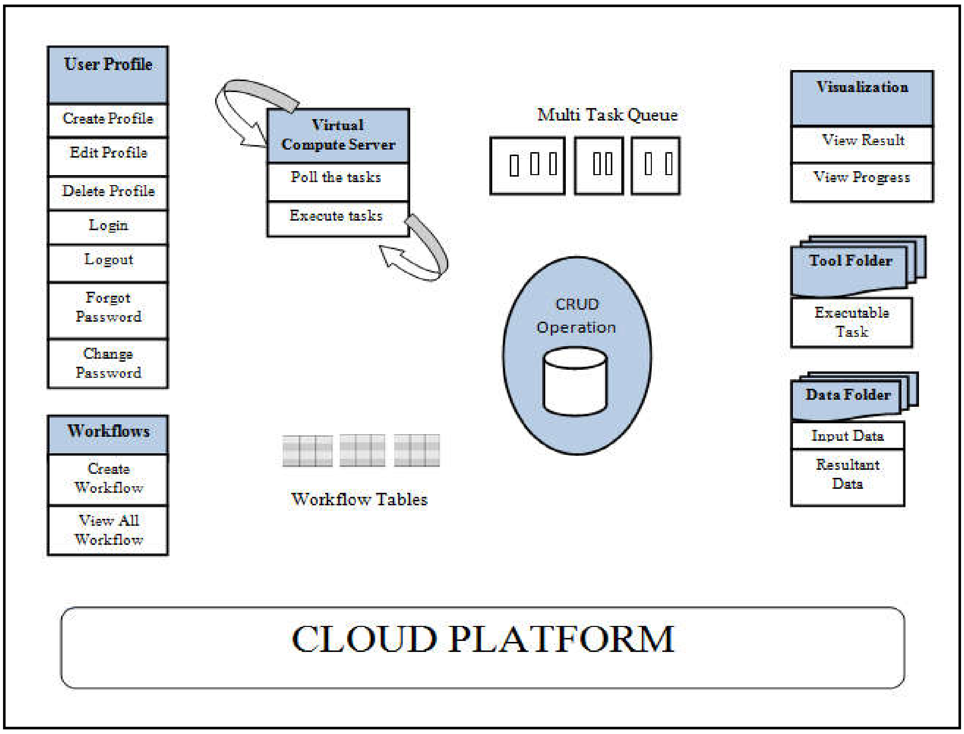

Figure 1 System Architecture

Account Operations

Account operations module will allow the end users to register a new seller/ buyer account, login to an existing account, logout from the session, edit the existing profile, change password for security issues, forgot password and receive the current password over an email and delete an existing account.

Workflow Creation and Listing

The end user will be able to create a new workflow by providing the name and description for the same. After creating the workflow, the user should be adding the tasks relevant to his/her workflow. The user must select any of the tasks that are available in the tool folder and add it to their workflow and provide the inputs to run the task along with the timestamp at which the task must be run. After creating the tasks, all the inputs provided by the user will be stored in the data folder.

Virtual Compute Server

Here, we are have created a background task which will be running all the time in every ‘x’ seconds duration where, ‘x’ can be configured by the user. This background task will be identified as the virtual compute server. This virtual compute server will be pulling out all the tasks created by all the users and will compare its time of execution with the current time and decides if the task must be executed at that instance or not. If the virtual compute server decides to execute the task at that instance, it’s going to load the task from tool folder into the JVM’s in-memory and loads the input from the data folder into the JVM’s in-memory which we will also call it as the local data folder and starts executing the task inside JVM. The result of the task will be written back into the data folder. The status of the task i.e. success or the failure will be notified to the creator of the task via email.

Visualization

Here, the end user will be provided with the visualization tool where he/she can visualize the status of the workflow he/she created. Basically, the visualization tool will be displaying a table with ‘n’ rows where ‘n’ is the number of workflows the user has created. Each row will have a percentage completed progress shown with green colour if the workflow is completed, or the red colour if the workflow has some errors. This colour code and the progress bar helps the customer to understand the status/progress of the workflow they created.

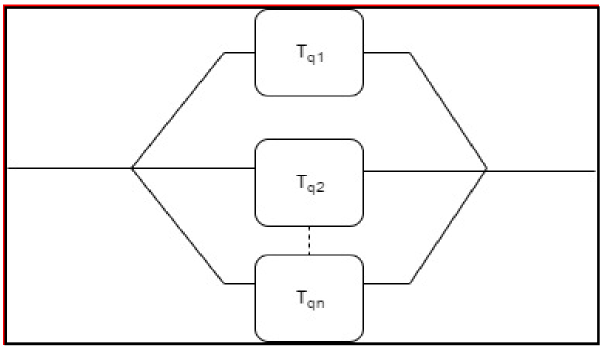

Figure 2 Multi-queues to do multi job scheduling

Data and Tool Folder

Data folder and tool folder are the two persistence storages. These storages will be realized in terms of MySQL tables. Basically, the data folder will contain the inputs for each of the task users have created and it will contain the output from each task the virtual compute server is going to execute. The tool folder will be containing the list of all executable tasks which will be used while creating a workflow by the user.

Figure 2 describes multi-queues which does multi job scheduling. Multiple tasks are added to the task table having execution time and date, the multi task table having its scheduling time is executed in ascending order.

Algorithm

Initialize multiple T_{q1}, T_{q2}, T_{q3} . . . . T_{qn} with constant size on each core

```
While ($T_{qi}$ != Empty)
    TT Status(RUNNING)
    Create LDF,LTF (for  I&O_list)
    If $P_T$ == IP_Value
        IP_list <- IP_Value
    Else
        If $P_T$ == OP_Value
            OP_list <- OP_Value
    Copy the content from GDF to LDF
    Transfer Exe_file &  Lib_file from GTF to LTF
    If current Task is complete
        TT Status(DONE)
        Copy the content from LDF to GDF
    else
        TT Status(FAIL)
    Remove the task from the table
    Delete LDF,LTF
```

In algorithm virtual compute operation, T_{qi} is multiple taskQueue where i=1, 2, 3…. n, TT is taskTable, LDF is localDatafolder, LTF is localToolfolder, P_T is parameterType, GDF is Global DataFolder, GTF is Global Toolfolder, IP and OP stands for input and output, Exe_file is the executable files that particular task, Lib_file contains all the library file for the execution of task.

CONCLUSION

Cloud framework is provided according with SaaS model which implies that no installation is required on the user machines which is independent from the infrastructure layer.

The end user will launch through a unique URL where it will be deployed LIVE on the clouds Platform as a Service. The user will be able to create a new workflow by defining what data mining problem he wants to execute. Each data mining problem will be containing multiple tasks (sub problems) details of which will have to be provided by the user during this process. He/she can create a new task for his problem or use any of the prepopulated tasks. During this process, the user also specifies the timestamp (includes both date and time) at which each of those tasks have to be executed. The user also specifies the email ID for which the notification of the status of this data mining problem will be notified intermittently. The user will be able to visualize the status of the workflow at any instance of the time by logging in into the application.

References

1. Marozzo, Fabrizio, Domenico Talia, and Paolo Trunfio. "A Workflow Management System for Scalable Data Mining on Clouds." *IEEE Transactions on Services Computing* (2016).

2. Goecks, Jeremy, Anton Nekrutenko, and James Taylor. "Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences." *Genome biology*11.8 (2010): R86.
3. Maheshwari, Ketan, *et al*. "Enabling Multi-task computation on Galaxy-based Gateways using Swift." *Cluster Computing (CLUSTER), 2013 IEEE International Conference on*. IEEE, 2013.
4. Wolstencroft, Katherine, *et al*. "The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud." *Nucleic acids research* 41.W1 (2013): W557-W561.
5. Podpečan, Vid, Monika Zemenova, and Nada Lavrač. "Orange4WS environment for service-oriented data mining." *The Computer Journal* 55.1 (2012): 82-98.
6. Ludäscher, Bertram, *et al*. "Scientific workflow management and the Kepler system." *Concurrency and Computation: Practice and Experience* 18.10 (2006): 1039-1065.
7. Hiden, Hugo, *et al*. "Developing cloud applications using the e-science central platform." *Phil. Trans. R. Soc. A* 371.1983 (2013): 20120085.
8. Kranjc, Janez, Vid Podpečan, and Nada Lavrač. "Clowdflows: a cloud based scientific workflow platform." *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2012.
9. Deelman, Ewa, *et al*. "Pegasus: A framework for mapping complex scientific workflows onto distributed systems." *Scientific Programming* 13.3 (2005): 219-237.
10. Ostermann, Simon, Radu Prodan, and Thomas Fahringer. "Extending grids with cloud resource management for scientific computing." *Grid Computing, 2009 10th IEEE/ACM International Conference on*. IEEE, 2009.

How to cite this article:

Divya C and Satish Kumar T.2017, Implementation of Data Mining Cloud Framework Algorithm for Creating and Managing Workflows. *Int J Recent Sci Res.* 8(6), pp. 17335-17338. DOI: http://dx.doi.org/10.24327/ijrsr.2017.0806.0333

****