RESEARCHERID
THOMSON REUTERS

[Jagateesan* *et al.*, 6(7): July, 2017]
IC™ Value: 3.00

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

A SURVEY ON BIGDATA SCHEDULING ON CLOUD FRAMEWORK

Dr.S.M. Jagateesan[1] , V.M.Pavithra[2]
Associate Professor[1] Research Scholar[2]
Department Of Computer Science & Engineering
Gobi Arts & Science College, Gobichettipalayam 638453, Tamilnadu

DOI: 10.5281/zenodo.834614

ABSTRACT

Computational science workflows have been successfully run on traditional High Performance Computing (HPC) systems like clusters and Grids for many years. Now a day, users are interested to execute their workflow applications in the Cloud to exploit the economic and technical benefits of this new rising technology. The deployment and management of workflows over the current existing heterogeneous and not yet interoperable Cloud providers, is still a challenging task for the workflow developers. The Pointer Gossip Content Addressable Network Montage Framework allows an automatic selection of the goal clouds, a uniform get entry to to the clouds, and workflow data management with respect to user Service Level Agreement (SLA) requirements. Consequently, a number of studies, focusing on different aspects, emerged in the literature. In this comparative review on workflow scheduling algorithm cloud environment is provide solution for the problems. Based on the analysis, the authors also highlight some research directions for future investigation. The previous results offer benefits to users by executing workflows with the expected performance and service quality at lowest cost.

KEYWORDS: Workflow, Montage framework, Pointer gossip, Content addressable network.

I. INTRODUCTION

Cloud is a group of computers or servers which are interconnected together to provide resources to the clients. It emerges as a brand new computing paradigm that aims to supply reliable, custom-made and QoS (Quality of Service) warranted computing dynamic environments for the stop customers. The main problems related to cloud computing are the network bandwidth, response time, minimum delay in data transfer and minimum transfer cost for data. In cloud computing the resource allocation plays an important role in the performance of the entire system and also the level of customer satisfaction provided by using the system. However, while providing the utmost customer satisfaction, the service provider ought to make sure of the profits to him also.

So the resource allocation should be economical on both views i.e. on the end user and the service company perspective. So as to get such a system the new technologies insist that the system should be with minimum SLA (Service Level Agreements) violation. There are numerous advantages of cloud computing, the most basic ones being lower costs, re-provisioning of resources and remote accessibility. Cloud computing lowers cost by avoiding the capital expenditure by the company in renting the physical infrastructure from a third party provider.

As big-data processing and analysis dominates the usage of Cloud systems today, the need for Cloud-hosted data scheduling and optimization services increases. Online service companies such as Amazon, Facebook, Google, Microsoft, and Yahoo! have made huge investments in networks of datacenters that host their online services and cloud platforms. Similarly, hosting and co-location services such as Equinix and Savvis employ distributed networks of datacenters that include tens of locations across the globe. A quick look at any data center directory service. Reveals that datacenters are popping up in large numbers every-where. The trend is driven primarily by the need to be co-located with customers for QoS (latency directly affects the revenues of web sites), energy and personnel costs, and by the need to be tolerant to catastrophic failures.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

II. LITERATURE SURVEY

Tevfik Kosar, Engin Arslan[9] has proposed the initial design and prototype implementation of Stork Cloud, and shows its correctiveness in large dataset transfers across geographically distant garage websites, data facilities, and collaborating institutions. Stork Cloud's scheduler is a modular, multi-protocol task scheduler which handles queuing and execution of data transfer jobs and ensures their timely completion. The scheduler's plug-in interface allows arbitrary protocol support to be implemented as standalone modules and easily introduced to the system. The scheduler inherits much of its functionality from the Stork Data Scheduler present the initial design and prototype implementation of Stork Cloud, and show its effectiveness in large dataset transfers across geographically distant storage sites, statistics centers, and collaborating institutions.

J. Li, M. Humphrey[2] has proposed their experiences taking MODISAzure, their satellite data processing system built on the Windows Azure cloud computing platform, from the evidence-of-concept stage to some extent of being able to run on significantly larger trouble sizes (e.g., from national-scale statistics sizes to worldwide-scale data sizes). To their knowledge, this is the longest-running eScience application on the nascent Windows Azure platform. They found that while many infrastructure-level issues were thankfully masked from their by the cloud infrastructure, it became treasured to design extra redundancy and fault-tolerance capabilities such as transparent idempotent task retry and logging to support debugging of user code encountering unanticipated data issues. Further, found that using a commercial cloud means anticipating in consistent performance and black-box behaviour of virtualized compute instances, as well as leveraging converting platform capabilities over time. They believe that the experiences presented in their paper can help future e-Science cloud application developers on Windows Azure and other commercial cloud providers

J. Dias, D. Oliveira[3] has proposed an algebraic approach (inspired by relational algebra) and a parallel execution model that enable automatic optimization of scientific workflows. They conducted a thorough validation of their approach using both a real oil exploitation application and synthetic data scenarios. The experiments were run in Chiron, a data-centric scientific workflow engine implemented to support their algebraic approach. Their experiments demonstrate performance improvements of up to 226% compared to an ad-hoc workflow implementation. A scientific workflow is composed of activities. An activity is a workflow component capable of running programs, with input and output parameters. Parameters are used to represent the dependency between activities within a workflow, i.e., if an input parameter of an activity B is connected to an output parameter of activity A it means that A must execute before B, and the data produced by A is consumed by B

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

Figure:

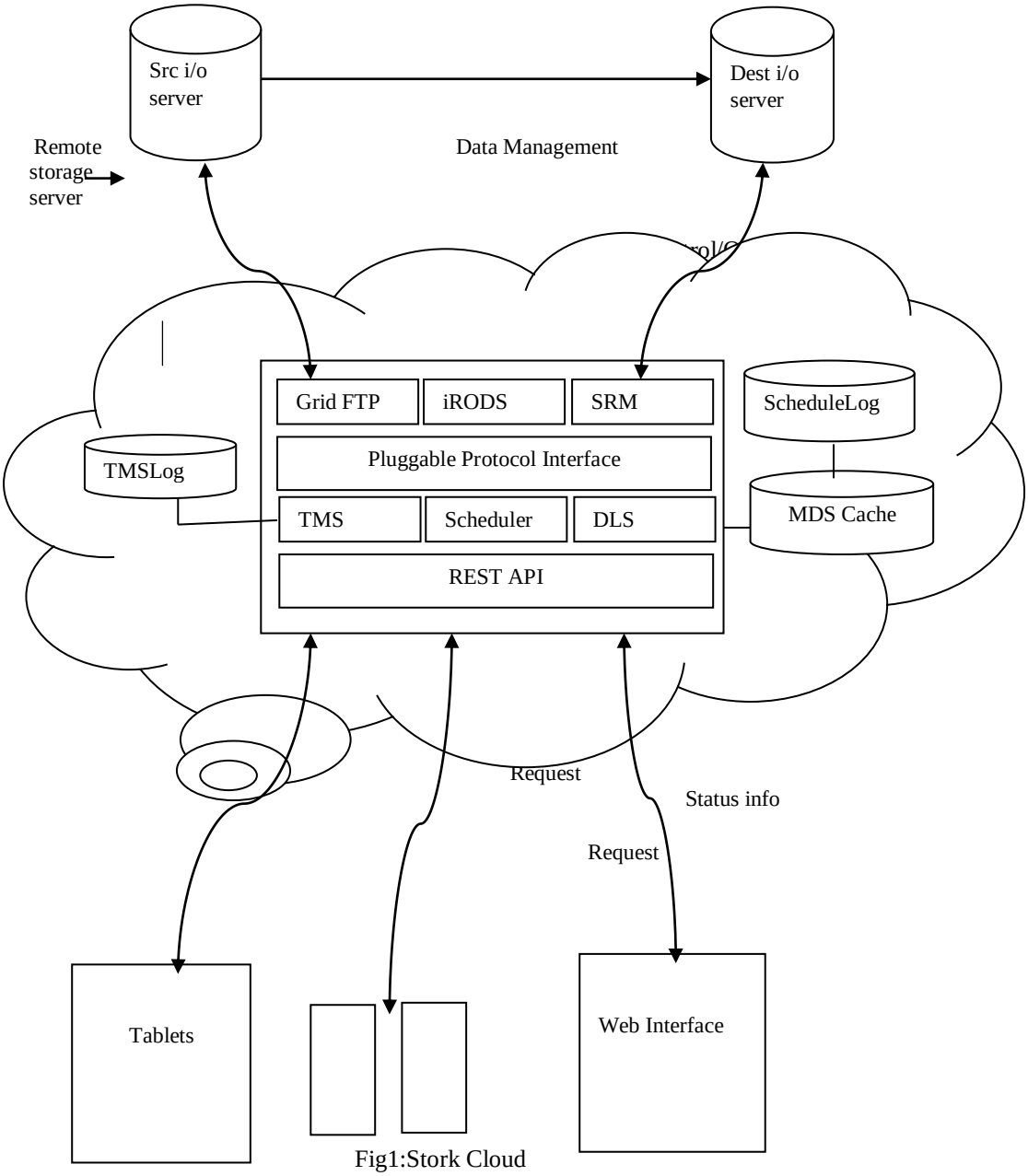

Fig1:Stork Cloud

G. Khanna, U. Catalyurek[8] has proposed develop a file transfer scheduling algorithm that incorporates the two optimizations in conjunction with the use of available file replicas. The algorithm makes use of information from past Grid FTP transfers to estimate network bandwidths and resource availability. The effectiveness of these optimizations is evaluated using several application file transfer patterns: one-to-all broadcast, all-to-one gather, and data redistribution, on a wide-area testbed. The experimental results showed the architecture and algorithm achieve significant performance improvement. The observed throughput of Grid FTP transfers in a wide-area environment may be lower than the maximum achievable throughput, due to a number of factors, such as the slow-start and congestion control mechanisms used by TCP. The technique of dividing a TCP connection into a set of shorter, better performing connections by splitting it at multiple intermediate points with the goal of improving the overall throughput. A split-TCP connection may perform better than a single end-to-end TCP connection due to several reasons. First, the round-trip time on each intermediate hop is shorter as compared to the direct end-to-end path. The congestion control mechanism of TCP would sense the maximum

throughput quickly thereby attaining steady state, wherein it will give maximal possible throughput until a congestion event occurs. Second, any packet loss is not propagated all the way back to the source but only to the previous intermediate hop. In this work, as an alternative to a direct TCP connection between a source and destination, they explore the use of multi-hop pipelined transfers using intermediate nodes. If the bandwidth on each of the intermediate hops is higher than the direct path, the overall throughput can be expected to improve.

T. J. Hacker, B. D. Noble[6] has proposed a new adaptive weighted scheduling approach for multiplexing data blocks over a set of parallel TCP streams. Their new scheduling approach, compared with the scheduling approached used by GridFTP, reduces reordering of data block between individual TCP streams, maintains the aggregate throughput gains of parallel TCP, consumes less receiver memory for buffering out-of-order packets, and delivers smoother application good put. They demonstrate the improved characteristics of their new scheduling approach using data transmission experiments over real and emulated wide-area networks.

W. Liu, B. Tieman[10] has proposed a data transfer framework comprising a high performance data transfer library primarily based on gridftp, a statistics scheduler, and a graphical consumer interface that allows users to transfer their data easily, reliably, and securely. Their system incorporates automatic tuning mechanisms to Select at runtime the number of concurrent threads to be used for transfers. Also covered are restart mechanisms capable of dealing with client, network, and server failures. Experimental results indicated the data transfer system can significantly improve data transfer performance and can recover well from failures. Grid FTP GUI provides users a convenient tool for data movement based on a graphical interface. The data scheduler accepts jobs and dispatches them to the data transfer library according to a specified scheduling coverage. The records transfer library hides the complexity and heterogeneity of underlying data switch protocol. it gives a information switch thread pool and supports error recovery. It can interact with diverse data transfer protocols, although currently they support only Grid FTP using CoG jglobus. The CoG jglobus library includes a pure Java Grid FTP client API; it can be used to build applications that communicate with Grid FTP servers.

C. Raiciu, C. Pluntke[15] has proposed a natural evolution of data centres transport from TCP to multipath TCP. They show that multipath TCP can effectively and seamlessly use available bandwidth, providing improved throughput and better fairness in these new topologies when compared to single path TCP and randomized flow-level load balancing. Also show that multipath TCP outperforms leggy centralized flow scheduling without needing centralized control or additional infrastructure.

From a high-level perspective, there are four main components to a data center networking architecture:

•Physical topology
•Routing over the topology
• Selection between the trails supplied by way of routing
• Congestion manage of site visitors on the chosen paths.

These are not independent; the performance of one will depend on the choices made by those preceding it in the list, and in some cases by those after it in the list. They were discussed each in turn, but it is worth noting now that MPTCP spans both path selection and congestion control, which is why it is able to offer benefits that cannot otherwise be obtained.

P. Carns, W.B. Ligon[1] stated the design and implementation of PVFS and present performance results on the Chiba City cluster at Argonne. They provide performance results for a workload of concurrent reads and writes for various numbers of compute nodes, I/O nodes, and I/O request sizes. They also present performance results for MPI-IO on PVFS, both for a concurrent read/write workload and for the BTIO benchmark. They compare the I/O performance when using a Martinet network versus a fast-Ethernet network for I/O-related communication in PVFS. They obtained read and write bandwidths as high as 700 Mbytes/sec with Martinet and 225 Mbytes/sec with fast Ethernet. As a parallel file system, the primary goal of PVFS is to provide high-speed access to file data for parallel programs. in addition, pvfs offers a cluster extensive consistent name space, enables user-controlled striping of data across disks on extraordinary i/o nodes, and lets in present binaries to function on PVFS files without the need for recompiling. Like many other file structures, pvfs is designed as a consumer-server system with multiple servers, called I/O daemons. I/O daemons typically run on separate nodes inside the cluster, known as i/o nodes, which have disks attached to them. Each PVFS file is striped across the disks on the i/o nodes. utility strategies have interaction with pvfs via a client library. PVFS also has a manager daemon that handles only metadata operations such as permission checking for document creation, open, close, and cast off operations. The manager does not participate in read/write operations; the client library and the I/O daemons handle all file I/O without the intervention of the supervisor. The clients, I/O daemons, and the manager need not be run on different machines. Running them on different machines may result in higher performance.

R. L. Grossman, Y. Gu[5] has proposed a cloud based infrastructure that have developed that is optimized for wide area, high performance networks and designed to support data mining programs. the infrastructure includes

a storage cloud called Sector and a compute cloud called Sphere. Sector has a layered architecture: there is a routing layer and a storage layer. Sector services, such as the Sphere compute cloud described below, are implemented over the storage layer. The routing layers provide services that locate the node that stores the metadata for a specific data file or computing service. That is, given a name, the routing layer returns the location of the node that has the metadata, such as the physical location in the system, of the named entity. Any routing protocols3 that can provide this function can be deployed in Sector. Currently, Sector uses the Chord P2P routing protocol. The next version of Sector will support specialized routing protocols designed for uniform wide area clouds, as well as non-uniform clouds in which bandwidth may vary between portions of the cloud. Data transport itself is done using specialized high performance network transport protocols, such as UDT. UDT is a rate-based application layer network transport protocol that supports large data owes over wide area high performance networks.

R. Tudoran, O. Nano[16] Listed a set of strategies for efficient transfers of events between cloud data-centers and they introduce JetStream: a prototype implementing these strategies as a high performance batch-based streaming middleware. Jet Stream is able to self-adapt to the streaming conditions by modelling and monitoring a set of context parameters. it similarly aggregates the available bandwidth by enabling multi-route streaming across cloud web sites. The prototype became confirmed on tens of nodes from US and Europe data centers of the Windows Azure cloud using synthetic benchmarks and with application code from the context of the Alice test at cern. The effects show an increase in transfer rate of 250 times over individual event streaming. Stream computing refers to the processing of continuous sequences of relatively small data items, Otherwise referred to as events. According to the characteristics of data streams are: 1) a stream is formed of events, which are tuples of records; 2) it is a potentially infinite sequence of events ordered by time (e.g., acquisition time, generation time); and 3) the events can be produced by multiple external sources at variable rates, independent of the constraints of the stream processing engine. The stream processing engines are systems which treat the events from the stream by applying the user's queries. As it is generally infeasible to store the streams entirely, the queries are applied continuously and iteratively over windows of events. The computation is event-driven, therefore most of the data management is done at the granularity of an event. Generally, the size of an event is small, in the range of bytes to kilobytes, which allows handling them in-memory, but also makes their transfer sensitive to any overhead.

III. CONCLUSION

In existing methods [1][12][13] does not supports large data in general comes in different formats, at different speed and at different volume. Their proposed approach processing consists of not just one application but several applications combined to form a workflow to achieve a certain goal. The previous studies do not work with large data variation and at different speed, but their work will focus applications' execution and resource needs will also vary at runtime. those are called dynamic workflows. One can say that the author can just throw more and more resources during runtime. Applied to efficiently schedule computation jobs among processing resources onto the cloud datacenters in a way to reduce execution time by spreading the jobs on to available resources. In [10][14] authors proposed is to create energy efficient clusters in cloud datacenters that shows that cluster creation that helps in decreasing the energy consumption as compared to other available algorithm. The future work should focus for this problem of economic power dispatch to obtain a system that is fast and more robust

IV. REFERENCES

[1] P. Carns, W.B. Ligon, R. B. Ross, and R. Thakur, "PVFS: A parallel file system for linux clusters," in Proc. 4th Annu. Linux Showcase Conf., 2000, pp. 317–327.

[2] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," Commun. ACM, vol. 51, no.1,pp. 107–113,Jan. 2008.

[3] J. Dias, D. Oliveira, M. Mattoso, "An Algebraic Approach for Data-Centric Scientific Workflows," Concurrency Comput: Practice Experience, vol. 25, no. 16, pp. 2327–2341, 2013.

[4] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: Research problems in data center networks," SIGCOMM Comput. Commun. Rev., vol. 39, no. 1, pp. 68–73, Dec. 2008.

[5] R. L. Grossman, Y. Gu, M. Sabala, and W. Zhang, "Compute and storage clouds using wide area high performance networks," Future Gener. Comput. Syst., vol. 25, pp. 179–183, 2009.

[6] T. J. Hacker, B. D. Noble, and B. D. Athey, "Adaptive data block scheduling for parallel TCP streams," in Proc. 14th IEEE High Perform. Distrib. Comput., 2005, pp. 265–275.

[7] H. Hiden, P. Watson, S. Woodman, D. Leahy "E-Science Central: Cloud-based e-Science and its application to chemical property modelling," in Proc. Roy. Soc. A, 2012, vol. 371, pp. 52–67.
[8] G. Khanna, U. Catalyurek, T. Kurc, R. Kettimuthu, P. Sadayappan, I. Foster, and J. Saltz, "Using overlays for efficient data transfer over shared wide-area networks," in Proc. ACM IEEE Conf. Super-comput., 2008, pp. 47:1–47:12.
[9] T. Kosar, E. Arslan, B. Ross, and B. Zhang, "Storkcloud: Datatransfer scheduling and optimization as a service," in Proc. 4th ACM Workshop Sci. Cloud Comput.,2013, pp. 29–36.
[10] W. Liu, B. Tieman, R. Kettimuthu, and I. Foster, "A data transfer framework for Large-scale science experiments," in Proc. 19th ACM Int. Symp. High Perform. Distrib. Comput., 2010, pp. 717–724.
[11] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-datacenter bulk transfers with netstitcher," in Proc. ACM SIGCOMM Conf., 2011, pp. 74–85.
[12] J. Li, M. Humphrey, Y. Cheah, Y. Ryu, D. Agarwal, K. Jackson, "Fault Tolerance and Scaling in e-Science Cloud Applications: Observations from the Continuing Development of MODISAzure," in Proc. BigData Conf., 2013, pp. 273–281.
[13] S. Pandey and R. Buyya, M. Vaquero , A. Celorio , F. Cuadrado "Deploying Large-Scale Data Sets on-Demand in the Cloud:Treats and Tricks on Data Distribution," in Proc. 10th IEEE/ACM Int. Conf. Cluster, Cloud Grid Comput., 2010, pp. 359–367.
[14] S. Pandey and R. Buyya, "Scheduling workflow applications based on Multi-source parallel data retrieval," Comput. J., vol. 55, no. 11, pp. 1288–1308, Nov. 2012.
[15] C. Raiciu, C. Pluntke, S. Barre, A. Greenhalgh, D. Wischik, and M.Handley, "Data center networking with multipath tcp," in Proc. 9th ACM SIGCOMM Workshop Hot Topics Netw., 2010, pp. 10:1–10:6.
[16] R. Tudoran, O. Nano, I. Santos, A. Costan, H. Soncu, L. Bouge, and G. Antoniu, "JetStream: Enabling high performance event streaming across cloud Data-centers," in Proc. 8th ACM Int. Conf. Distrib. Event-Based Syst., 2014, pp. 23–34.

CITE AN ARTICLE

Jagateesan, S. M., and V. M. Pavithra. "A SURVEY ON BIGDATA SCHEDULING ON CLOUD FRAMEWORK ." *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY* 6.7 (2017): 922-27. Web. 25 July 2017.