International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 11 (2018) pp. 9407-9412


# A Bipartite Graph based Data Placement Technique for Cloud based Storage Area Network

**Shabeen Taj G A[1] and Dr. G.Mahadevan[2]**

[1]*Assistant professor, Department of Computer Science and Engineering, Government Engineering College, Ramanagar, India.*

[2]*Professor of Computer Science and Engineering, AMCEC, 18th KM, Bannerghatta Road, Bangalore, India.*

**Abstract**

The growth of information technology has led to increased adoption of scientific workflow execution for various application uses. Various storage framework is adopted to solve storage and computing problem of scientific workflow execution. Storage Area Network (SAN) is a traditional storage framework used to solve research problems. However, they are not efficient for current dynamic resource requirement. Provisioning efficient resource allocation for user in SAN involves numerous challenges such as data placement, data reconfiguration and data monitoring. To provision dynamic resource requirement, SAN system use hybrid architecture using SDN and cloud platform. Minimizing data access cost and latency on such platform/network is most desired. To solve the data placement problem, graph based and optimization technique is adopted by exiting model which aid in minimizing computation time and latency. However, adopting their architecture on cloud platform incurs latency affecting performance. To address this work present a Bipartite Graph based Data Placement (BGDP) Technique for Cloud based Storage Area Network to provision execution of real-time scientific workflow. Experiment are conducted using real time scientific data workflow. The outcome shows BGDP technique minimize data access latency, computation time and cost.

**Keywords:** Cloud, Bipartite graph, ILP, SAN, SDN.

## INTRODUCTION

The storage platform is considered to be a critical element of any computing framework. The storage platform can be either centralized are distributed in nature. The distributed storage architecture is predominantly adopted by various researcher which aid in providing durability, reliability, availability and scalability. The storage device are prone to disk failure, durability represent the longevity of data access without disk failure. To provide continuous durability and avoid single point failure, the data are stored across multiple server and datacenter. Scalability reduces the data access latency from different datacenter and reliability permits the correctness of the data.

Currently there exist different types of storage technology available in market such as Cassandra [1], Freenet [2] and Bigtable [3] and so on with different features. Therefore, when designing a storage system it is important to identify the most important features. Storage platform are increasingly getting wide attention due to increase usage of smart devices for various application uses such as Bioinformatics, scientific and, space research etc. Therefore minimizing data access is most desired on such application. Related technologies also upgrading itself to cater the current market dynamics and needs such as disk manufacturing and file system technologies. The outcome/performance of these technologies depends on design and optimization technique used.

In scientific domain, scientific application produces massive volume of scientific data that need to be processed/analyzed. To process scientific data requires huge computing resource and to store the raw outcome of these analysis requires an efficient storage system. To process these data new scientific framework are designed such as XrootD [4] and NetCDF [5]. In general, the data in scientific application are predominantly read only or append only. As a result, high performance application requires high I/O (input/output) request on the storage framework, which enables parallelization within scientific application and storage framework. In [6] conducted many physics experiment with storage and computation power which provides data. In order to provision scalable and high performance storage framework different storage technologies like Network attached Storage (NAS), Direct-attached Storage (DAS) and Storage Area Network (SAN).

The outcome obtained in [7] shows that the SAN gives better performance than NAS. Provisioning efficient resource allocation for user in SAN involves numerous challenges such as data placement, data reconfiguration and data monitoring. Minimizing data access cost and latency on such network is most desired. To address the existing approach [8] and [9] presented cache optimization, cost optimization reconfiguration technique for data placement. However, they are not efficient for present application dynamic computing resource requirement. To address the resource dynamic requirement new architecture is been adopted such cloud and SDN. Many existing model [10], [11] and [12] have adopted hybrid approach using SAN to provision efficient service to user. The growth of storage industries and cloud platform has led to evolution in way user stores and access data. Various hybrid and heterogeneous network is been adopted [13]. The future SAN design should consider heterogeneity of storage in provisioning service to users. In [14] presented a dynamic Solid State Drives (SSD) resource allocation scheme for heterogeneous I/O workload on virtualized storage environment. The model in [14] adopted virtual resource partitioning for cache optimization. However, the model did not consider dynamic traffic pattern of user to solve data placement problems. To address [15] presented an adaptive placement algorithm for high performance scientific applications. They presented a checkpoint based placement optimization algorithm which utilize both burst (traffic) and parallel filesystem. The outcome obtained by [15] shows significance performance improvement over state-of-art architecture. However, implementing such architecture on cloud platform induce data access latency. Since, the data are placed across data center which are placed on different location globally. As a result, induces high cost and computation overhead [16].

To overcome the research issue, graph partitioning and optimization technique is adopted in [13], [14], [15], [18], and [19] respectively. This work present a Bipartite Graph based Data Placement (BGDP) Technique for Cloud based Storage Area Network to provision execution of real-time scientific workflow. The BGDP technique aims at minimizing data access latency, computation time and cost.

*The Contribution of research work is as follows:*

- This work consider Bipartite graph based model for data placement on cloud based SAN network.
- We consider multi-objective function to find optimal data placement solution.
- Experiment are conducted on real-time scientific work flow and performance is evaluated in terms of processing time of task completion time and other system parameters.
- The outcome shows significance performance over state-of-art architecture.

The rest of the paper is organized as follows. In section II the proposed model is presented. In penultimate section experimental study is carried out. The conclusion and future work is described in last section.

## PROPOSED ALGORITHM

Here we present an efficient data placement mechanism for cloud based Storage Area Network (SAN).

*a) System Model:*

Let consider there are $I$ of $K$ data objects to be stored in the cloud based SAN network environment. The data objects can be files, fragments or tables which depends on the type of data storage be used. Each request form user/client in network would not exceed more than $N$ different objects from the $I^{th}$ set. The request pattern $\mathbb{A}$ of user can be represented as

$$\mathbb{A} = \{I, \emptyset\}^N \quad (1)$$

and represent the object or request pattern $a$ . The requests pattern of user generally a subset of Eq. (1), which is represented as

$$\mathcal{A} \subset \mathbb{A} = \{I, \emptyset\}^N. \quad (2)$$

Let consider that data objects are stored across datacenter which are geographically distributed. This distributed storage architecture aid in minimizing data access latency. Let consider that each data object $i \in I$ is stored at distinctive datacenter $j \in J$. Therefore, the mapping of data with respect to location can be defined as

$$\mathcal{Y}: i \rightarrow j \quad (3)$$

which identifies the storage location $j$ of each objects $i$. This work aims at optimizing $\mathcal{Y}$ for efficient data placement and consider $Y_j$ to represent set of data objects placed in datacenter $j$ after data placement decision.

We consider that the request of user are directed toward nodes closest to the location of user. The user may be within or outside the datacenter that stores user's data. The user are considered be present outside the datacenter when requesting for services and another case is that the task running on computing platform may request for data objects. For easiness, we consider the node that request is primarily as source location of the request. As a result, the node in our work has two important roles simultaneously. Firstly, destination location that possess data and source location of request. To distinguish a datacenter $j$ being destination or source request, we use $y$ and $x$ respectively.

The real-time workload rate of each pattern $a \in \mathcal{A}$ from the demanding datacenter $j \in J$ can be computed, which is represented by $\mathcal{G}_{aj}$. The data placement decision is done using predicted request rate. However, the predicted rate are not same as the computed one, for easiness we use the request rate set as

$$\mathcal{G} = \{\mathcal{G}_{aj} | a \in \mathcal{A}, j \in J\}. \quad (4)$$

We consider $\mathcal{G}$ to be a bipartite graph, where request patterns and nodes are two sets of vertices, and the edges among them are weighted by the rate $\mathcal{G}_{aj}$. To provide graph based data placement, two types of request rate can be modelled based on $\mathcal{G}$. One is cumulative request rate to a pattern $a$ irrespective location of source. For all pattern $a$, we compute its cumulative rate of request as follows

$$\mathcal{G}_a = \sum_{j \in J} \mathcal{G}_{aj}. \quad (5)$$

Another is the cumulated request rate to a data objects $i$ a source datacenter $j$. For all data objects $i$, we can compute its cumulated rate of request at each source datacenter $j$ as follows

$$\mathcal{G}_a = \sum_{a \in \mathcal{A}} \mathcal{G}_{aj} \, 1(i \in a). \quad (6)$$

The data placement problem of cloud based SAN network depends on the objective of performance, cost and efficiency. The data placement problem can be formulated as, for a given workload $\mathcal{W} = \{\mathcal{A}, \mathcal{G}\}$, compute the optimal policy of $\mathcal{Y}: i \rightarrow j$ that minimize objective parameter defined on $\mathcal{W}$ and $\mathcal{Y}$, subject to optimization limit. We consider optimization limit as the worst case recovery time upon data failure limitation. We define optimization based number of data objects placed in different node. That is the number of data objects placed in each node $j$, should be range of $[(1-\beta)l_d, (1+\beta)l_d]$, where $\beta$ is the optimization parameter and $l_d$ is the mean number of data objects placed in all nodes. The objective parameter is considered to be linear function with placement decision $\mathcal{Y}$ being the variable with $\mathcal{G}_{ij}$being the coefficient and $\mathcal{G}_a$being the accumulated rate of request. The performance of data placement can be measured and optimized by various metric which are discussed later.

*b) Graph based data :*

This work adopts a graph based data placement model to solve the unawareness of the difference among locations and its relationship among multiple objects. Let consider a Bipartite graph $\mathcal{L}(K, H)$, where $K$ represent the vertices and $H$ represent the edges. The graph support multiple vertices for each edges while foe edges only two vertices are allowed utmost. This model considers set of vertices with all datacenter and data objects which is represented as

$$K = I \cup J. \tag{7}$$

The edge set $H$ represent all the request patterns and all the pair among each data objects and datacenter which can be defined as follows

$$H = \{h_a | a \in \mathcal{A}\} \cup \{h_{ij} \big| i \in I, j \in J\}. \tag{8}$$

This work adopt Bipartite graph, as a result there exist multiple data objects for every request pattern edge. Each edge $h \in H$ is a given a weight to assure certain QoS requirement of data placement, in order to minimize latency of data access by end client.

The real-time workflow task completion efficiency depends on amount of data accessed and number of nodes which are distributed across different datacenter used for processing these tasks. Let $X_a$be the time duration of request $a$. Since each datacenter may not possess enough data objects and $X_{aj}$represent the number of data objects in request $a$. Since, each node in a datacenter may not be position to cater all data objects in $a$. The relativity among $X_a$ and $X_aj$ is obtained as follows

$$\sum_{j\in J} X_{aj} = X_a. \tag{9}$$

The computation time to optimally satisfies the request $a$ at datacenter $j$ is given as follows

$$u_{aj} = X_{aj} + \alpha.1(X_{aj}), \tag{10}$$

where $1(X_{aj})$ represent whether $X_{aj} \geq 1$ and $\alpha$ represent the overhead incurred in handling request from user(.i.e. TCP connection establishment). Each user in network has different data requirement (patterns) with different request rates. The objective to fulfill all the user request is

$$\sum_{a\in\mathcal{A}} \mathcal{G}_a \sum_{j\in J} u_{aj} \tag{11}$$

which is equal to

$$\sum_{j\in J}\sum_{a\in\mathcal{A}} \mathcal{G}_a[X_{aj} + \alpha.1(X_{aj})], \tag{12}$$

Minimizing (11) aid in reducing the cost of provisioning service for service provider. (.i.e. the amount of resource required for provisioning SAN on cloud environment. For case, let consider when placing data objects in a pattern of $a$ to a particular datacenter, the computation time on $a$ will be lower limit of $\mathcal{G}_a(X_{aj} + \alpha)$. Since, $\sum_{j\in J}\sum_{a\in\mathcal{A}} \mathcal{G}_a . X_{aj}$ in Eq. (12) is a constant for any scientific workflow task, thus cost of computation is defined as follows

$$\mathcal{C}^{[X]} = \sum_{j\in J}\sum_{a\in\mathcal{A}} \mathcal{G}_a.\alpha.1(X_{aj}). \tag{13}$$

By assigning weight for edge $h_a$ using

$$m_a^{[X]} = \alpha\mathcal{G}_a, \tag{14}$$

to minimize weight of Bipartite graph partitioning identical to minimizing Eq. (12).

The data access latency of requested data objects for Cloud based SAN network is modelled as follows. Let consider retrieving latency from datacenter $x$ to datacenter $y$ is a constant $S_{xy}^{Q}$, the objective our model is to minimize the cumulative latency

$$Q^{[Q]} = \sum_{i\in I}\sum_{j\in J} 1_{ij}\mathcal{O}_{ij}^{[Q]}, \tag{15}$$

where

$$\mathcal{O}_{ij}^{[Q]} = \sum_{s\in J} S_{xy}^{[Q]}\mathcal{G}_{ix}. \tag{16}$$

We can still optimize the QoS by obtaining series of optimization parameter as follows

$$\mathcal{O}^{[*]} = \sum_{i\in I}\sum_{j\in J} 1_{ij}\mathcal{O}_{ij}^{[+]}, \tag{17}$$

The Eq. (17) is subjected to constraint that object $i$ is exactly placed at one of the datacenter in $J$ which can further minimize $\sum_{j\in J} 1_{xy}\mathcal{O}_{ij}^{[*]}$ for each $i$. As a result, we need to compute weight $m_{ij}$ for the node edge $h_{ij}$, so that the partitioning is identical to data placement objectives. For every data object $i \in I$ in the cloud based SAN network, we attain an equation array as follows

$$\begin{cases} \mathcal{O}_{i1}^{[+]} = \quad m_{i2} + m_{i3} + \cdots \quad + m_{iT} \\ \mathcal{O}_{i1}^{[+]} = m_{i1} \quad + m_{i3} + \cdots \quad + m_{iT} \\ \quad \ldots\ldots\ldots \\ \mathcal{O}_{iT}^{[+]} = m_{i1} + m_{i2} + m_{i3} + \cdots + m_{i,T-1} \end{cases} \tag{18}$$

In Eq. (18) each line $j$ computes overhead in placing object $i$ in datacenter $j$. The weight among data object $i$ and datacenter $j$ is computed as follows

$$m_{iy}^{[+]} = \sum_{j \in J} \frac{\mathcal{O}_{ij}}{(T-1) - \mathcal{O}_{iy}}. \quad (19)$$

The cost of storage of a data object $i$ at location $y$ by $S_{ij}^{[S]}$. Therefore to minimize the cost of storage on such platform is obtained as follows

$$Q^{[S]} = \sum_{i \in I} \sum_{j \in J} 1_{ij} \mathcal{O}_{ij}^{[S]}, \quad (20)$$

where

$$\mathcal{O}_{iy}^{[S]} = S_{iy}^{[S]} \quad (21)$$

We compute the edge weight $h_{iy}^{[S]}$ by applying $\mathcal{O}_{iy}^{[S]}$ to Eq. (19) in order handle heterogeneous size of data objects. As a result, $\mathcal{O}_{iy}^{[S]}$is divided by the respective data objects size. Since this work considers multi-objective function, we set the weight of every edge in the graph to the multi-objective function which is shown as follows

$$S = M.\left(\mathcal{C}^{[X]}, Q^{[Q]}, Q^{[S]}\right)^{U} \quad (22)$$

where $M$ is the weighted vector of multi-objective optimization metrics factor. The cumulative weight of each individual metrics is obtained as follows

$$\begin{cases} M.\left(m_a^{[X]}, 0,0\right)^{U}, for\ each\ edge\ h_a \\ M.\left(0, m_{ij}^{[Q]}, m_{ij}^{[S]}\right)^{U}, for\ each\ edge\ h_{ij} \end{cases} \quad (23)$$

Note that in Eq. (19), the computed weight of some node edge is negative, and our Bipartite graph supports only non-negative edge weights. For such scenario, we can increase the edge of all computed weight by $\delta$, in order to make all the weight positive which is represented as follows

$$\delta = \min\{m_{ij} | i \in I, j \in J\}. \quad (24)$$

Since before partitioning we cumulate each edge from each data objects to each datacenter and there exist only one edge after partitioning. This enables us an optimal solution for optimizing cost. Thus we have

$$\begin{cases} M.\left(m_a^{[X]}, 0,0\right)^{U}, for\ each\ edge\ h_a \\ M.\left(0, m_{ij}^{[Q]}, m_{ij}^{[S]}\right)^{U} + \delta, for\ each\ edge\ h_{ij} \end{cases} \quad (25)$$

Let consider an input $\mathcal{W} = \{\mathcal{A}, \mathcal{G}\}$, we can determine it as a Bipartite graph $(K, H)$, where the weight if the edge is computed based on Eq. (25). From the optimal solution of Bipartite graph with parameters $\{\mathcal{A}, T, \beta\}$, we achieve the data placement $\mathcal{Y}$ and the cost of portioning is defined by $\mathcal{L}$, which satisfies $\mathcal{L} = S - R$, where $R$ is a constant, and $S$ is obtained from Eq. (22) based on $\mathcal{Y}$. Similarly, $\mathcal{L} + R$ is the minimum feasible parameter of Eq. (22) for optimal and efficient data placement for cloud based SAN network.

In next section the performance of proposed model is evaluated considering different types of real time scientific application and is compared with state-of-art architecture.

## SIMULATION RESULT AND ANNALYSIS

The experiments are conducted on windows 10 operating system, 64-bit I-5 quad core processor with 12GB RAM with 2 GB dedicated CUDA GPU. This work consider dataset obtained [17] such as Inspiral and Montage. The proposed and existing model [15] is designed using Java 8 and eclipse neon framework. The performance of proposed BGDP model is evaluated in term of workflow execution time, latency and computation cost. The proposed BGDP model performance is compared with existing model [15].

### a) *Computation time performance:*

Experiments are conducted to evaluate the performance of BGDP in terms of computation time. Two cases/dataset are considered namely Inspiral_30 and Montage workflow. The number datacenter are varied from 10 to 40, 100 number of user are considered and each datacenter consist of 10 nodes each. The experimental outcomes show the proposed BGDP performs better than existing model. An average improvement of 89.02% and 92.86% is achieved by proposed model over existing model for Inspiral and Montage respectively.

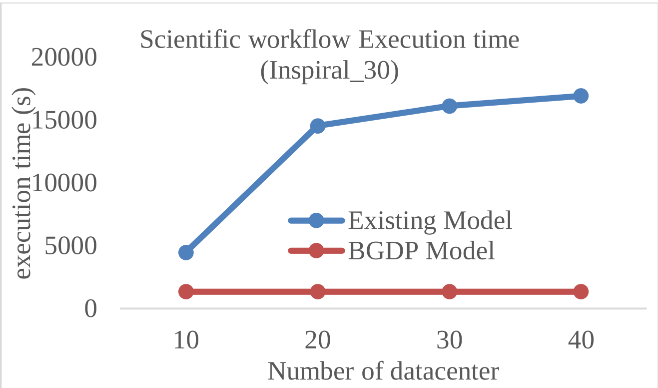


**Figure 1.** Scientific workflow execution time considering Inspiral_30 dataset

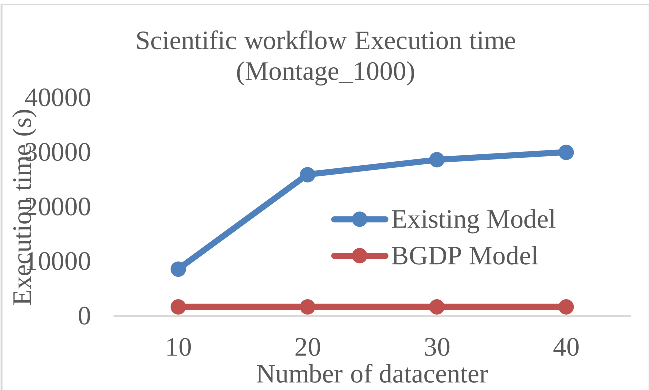


**Figure 2.** Scientific workflow execution time considering Montage_100 dataset

*b) Latency performance:*

Experiments are conducted to evaluate the performance of BGDP in terms of latency incurred. Two cases/dataset are considered namely Inspiral_30 and Montage workflow. The number datacenter are varied from 10 to 40, 100 number of user are considered and each datacenter consist of 10 nodes each. The experimental outcomes show the proposed BGDP performs better than existing model. An average improvement of 2.67% and 2.34% is achieved by proposed model over existing model for Inspiral and Montage respectively.

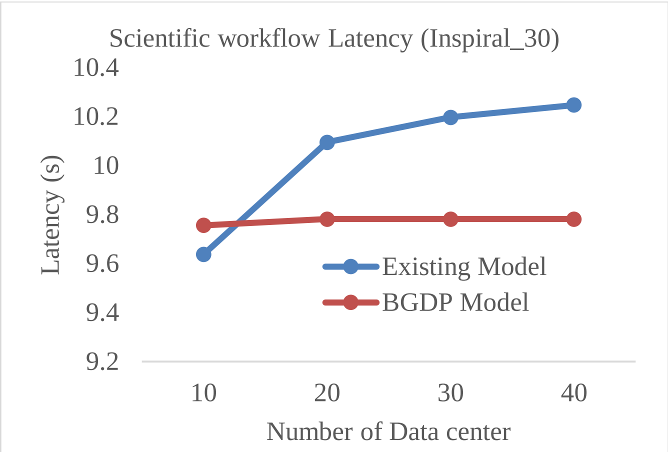


**Figure 3.** Latency performance considering Montage_100 dataset

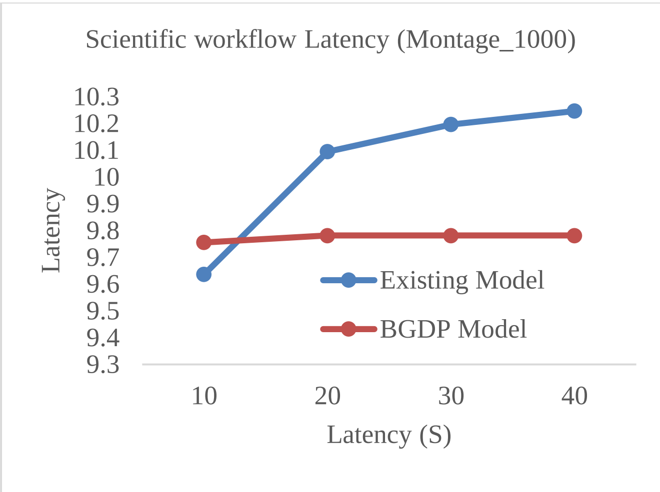


**Figure 4.** Latency performance considering Montage_100 dataset

*c) Computing cost performance:*

Experiments are conducted to evaluate the performance of BGDP in terms of computation cost. Two cases/dataset are considered namely Inspiral_30 and Montage workflow. The number datacenter are varied from 10 to 40, 100 number of user are considered and each datacenter consist of 10 nodes each. The experimental outcomes show the proposed BGDP performs better than existing model. An average improvement of 5.02% and 4.88% is achieved by proposed model over existing model for Inspiral and Montage respectively.

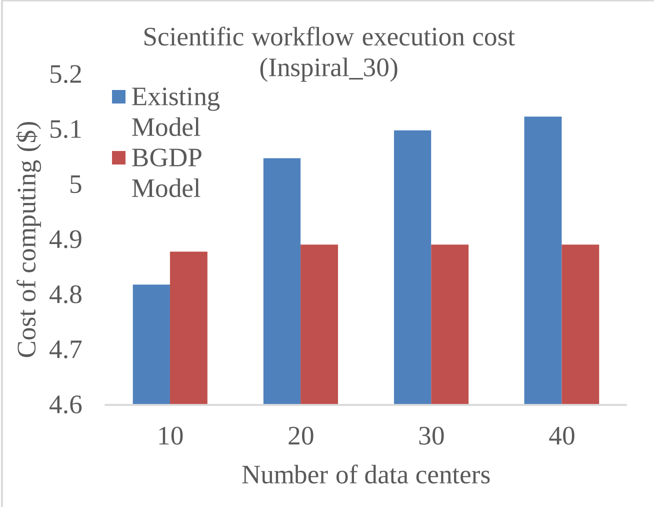


**Figure 5.** Scientific workflow computing cost considering Montage_100 dataset

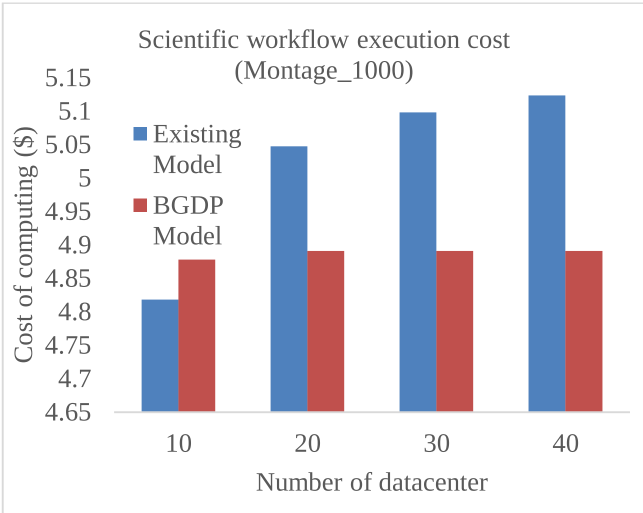


**Figure 6.** Scientific workflow computing cost considering Montage_100 dataset

## CONCLUSION

Many model has been presented recently to solve data placement problem for scientific work flow execution on SAN. To provide dynamic resource requirement, SAN system use hybrid architecture using SDN and cloud platform. However, implementing such hybrid architecture on cloud platform induce data access latency. Since, the data are placed across data center which are placed on different location globally. As a result, induces high cost and computation overhead. To address we present an efficient Data placement model for cloud based SAN namely Bipartite Graph based Data Placement (BGDP). Experiment are computed using real-time workflow. The outcome shows an average performance improvement of 89.08%, 2.67% and 5.02% in terms computation time, latency and cost respectively. The overall result achieved shows the efficacy and robustness of our model. The future work would consider content caching optimization, energy minimization and other system parameter in to the optimization function to further minimize latency, cost and energy consumed.

## REFERENCE


[1] Lakshman, Avinash, and Prashant Malik. ”Cassandra: a decentralized structured storage system.” ACM SIGOPS Operating Systems Review 44, no. 2: 35-40, 2010.

[2] Clarke, Ian, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. ”Freenet: A distributed anonymous information storage and retrieval system.” In Designing Privacy Enhancing Technologies, pp. 46-66. Springer Berlin Heidelberg, 2001.

[3] Rew, Russ, and Glenn Davis. ”NetCDF: an interface for scientific data access.” Computer Graphics and Applications, IEEE 10, no. 4: 76-82, 1990.

[4] XRootD, http://xrootd.org/, Oct 2015.

[5] Chang,Fay, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. ”Bigtable: A distributed storage system for structured data.” ACM Transactions on Computer Systems (TOCS) 26, no. 2 : 4, 2008.

[6] GSDC, http://en.kisti.re.kr/supercomputing/, Nov. 2015.

[7] A. Jaikar, S. A. R. Shah, S. Y. Noh and S. Bae, "Performance Analysis of NAS and SAN Storage for Scientific Workflow," 2016 International Conference on Platform Technology and Service (PlatCon), Jeju, pp. 1-4, 2016.

[8] Y. Ren, T. Li, D. Yu, S. Jin and T. Robertazzi, "Design, Implementation, and Evaluation of a NUMA-Aware Cache for iSCSI Storage Servers," in IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 2, pp. 413-422, Feb. 2015.

[9] Hadas Shachnai a,∗, Gal Tamir a, Tami Tamir. “Minimal Cost Reconfiguration of Data Placement in Storage Area Network” International Workshop on Approximation and Online Algorithms, pp 229-241, 2012.

[10] O. Sadov et al., "OpenFlow SDN testbed for Storage Area Network," 2014 International Science and Technology Conference (Modern Networking Technologies) (MoNeTeC), Moscow, 2014, pp. 1-3.

[11] Rekha P M and Dakshayini M, "Dynamic network configuration and Virtual management protocol for open switch in cloud environment," 2015 IEEE International Advance Computing Conference (IACC), Banglore, 2015, pp. 143-148.

[12] N. Yoshino, H. Oguma, S. Kamedm and N. Suematsu, "Feasibility study of expansion of OpenFlow network using satellite communication to wide area," 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), Milan, Italy, 2017, pp. 647-651.

[13] J. J. Kuo, S. H. Shen, M. H. Yang, D. N. Yang, M. J. Tsai and W. T. Chen, "Service Overlay Forest Embedding for Software-Defined Cloud Networks," 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 2017, pp. 720-730.

[14] Z. Yang, J. Tai, J. Bhimani, J. Wang, N. Mi and B. Sheng, "GReM: Dynamic SSD resource allocation in virtualized storage systems with heterogeneous IO workloads," 2016 IEEE 35th International Performance Computing and Communications Conference (IPCCC), Las Vegas, NV, pp. 1-8, 2016.

[15] Lipeng Wan, Qing Cao, Feiyi Wang, Sarp Oral ”Optimizing checkpoint data placement with guaranteed burst buffer endurance in large-scale hierarchical storage systems,” Journal of Parallel and Distributed Computing, Volume 100, Pages 16-29, 2017.

[16] I. Sadooghi et al., "Understanding the Performance and Potential of Cloud Computing for Scientific Applications," in IEEE Transactions on Cloud Computing, vol. 5, no. 2, pp. 358-371, April-June 1 2017.

[17] Bharathi S, Chervenak A, Deelman E, Mehta G, Su MH, Vahi K. Characterization of scientific workflows. In: Workflows in Support of Large-Scale Science, 2008. WORKS 2008. Third Workshop on; p. 1±10, 2008.

[18] J. Wei et al., "Minimizing Data Transmission Latency by Bipartite Graph in MapReduce," 2015 IEEE International Conference on Cluster Computing, Chicago, IL, 2015, pp. 521-522.

[19] Ankur Sahai “Online Assignment Algorithms for Dynamic Bipartite Graphs” arXiv.org, arXiv:1105.0232, 2011.