International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V10I5P103

Volume 10 Issue 5, 28-37, May 2023

Original Article

Energy-Efficient Data Offloading using Data Access Strategy-Based Data Grouping Scheme

Prabhu Shankar[1*], Sharon M[2], Viji[3], Rajkumar[1], Vetrimani[2], R. Surendiran[4]

[1]*Department of Computer Science and Engineering, Faculty of Engineering and Technology, JAIN (Deemed to be University), Banglore, Karnataka, India*
[2]*School of Computer Science and Engineering & Information Science, Presidency University, Banglore, Karnataka, India*
[3]*Department of Computer Science and Information Technology, JAIN (Deemed to be University), Banglore, Karnataka, India*
[4]*School of Information Science, Annai College of Arts and Science, Kumbakonam, India,*

[1]*Corresponding Author : prabu2000@gmail.com*

Received: 04 March 2023 Revised: 09 April 2023 Accepted: 04 May 2023 Published: 31 May 2023

Abstract *- With technological advancements, the large volume of data generated every day is increasing exponentially. The primary data source is smart devices, the Internet of Things (IoT), and social communication network groups. This paved the way for cloud computing to offer massive storage capacity and powerful computational resources. The conventional method of processing data in the cloud is not scalable and cannot meet the requirements of latency-critical applications. The main problem arises when a single data processing task requires multiple data items stored in different data servers. Efficient data placement minimizes the data access cost and data scheduling between the globally distributed data centres. Giving high priority to this alarming issue, this work aimed to propose an efficient data placement method called Data Access Strategy based Data Grouping scheme (DAS-DGS). The existing data sets are distributed to the data centres using access similarity measures, and the newly generated datasets are also dynamically placed on the appropriate servers in the cloud. Experimental simulations show that the proposed method meets multiple objectives: effective energy-efficient data placement scheme for IoT data, reduces the data access cost, minimization access time, average resource utilization, and energy consumption rate. The proposed DAS-DGS shows a promising result compared to the existing methods.*

Keywords *- IoT, Cloud computing, Data placement, Latency, Access cost, Machine learning.*

1. Introduction

With the vast growth of smart devices and information and communication technology, people's demand for data storage is also increasing. The storage capacity available in the devices is unable to meet the requirements. In such situations, the cloud server is a fruitful technology to overcome the storage issue. Cloud computing is a distributed architecture model that can offer highly effective computational resources and a vast storage system. [1]. However, due to the distributed computing environment's ability to store enormous amounts of data, some situational data saved in a specific data centre cannot be transported to another. Sometimes, a process may need data kept in different data centres.

Data has to be transferred from one data centre to another data centre to get accessed. Scheduling data across datacentres is a major challenge due to complex architecture, enormous amounts of data, and limited network bandwidth. This can be solved by placing simultaneously accessed data in the same data centre, which makes the data processing entirely local and reduces access time. Access delay is intolerable for some applications to run successfully. Numerous methods can be used to place data in a distributed environment.

The data placement method is categorized into static and dynamic data placement. Firstly, a thorough understanding of each file's service time and access rate is necessary for the static data placement approach. Secondly, the dynamic data placement algorithms develop file allocation to handle changing workload patterns [2, 3]. Dynamic data placement updates the placements schemes on every request. This is effective only when the size of the dataset is small.

The most difficult challenge is reducing costs while considering all aspects, including remote data serving, data access, and data storage capacity of the data centre. From the literature study, most existing data placement methods consider storing the datasets in closer data centres or those with related data access requests.

Considering all these issues, this work proposed an efficient energy-saving and high resource utilization method DAS-DGS to place existing datasets in appropriate data centres and dynamically generated datasets in a reasonable data centre during the run time [4]. The data placement method performed effectively regarding average access time, data access time, and power consumption in the proposed DAS-DGS. It reduced the overall latency by 3s and 4s for 800 and 1000 datasets, respectively, compared to the existing IoT Oriented Data Placement (IDP) method. The average utilization of the resources is 20% high in DAS-DGS compared to the traditional DRAW method for all scales of datasets.

This paper is organized as follows: Section 2 describes a detailed literature study of data placement methods. The proposed method is pictured in section 3. Simulation results are discussed in Section 4, and Section 5 concludes this paper.

2. Related Work

This section of the paper presents various strategies, algorithms and methods proposed for placing the data in cloud datacentres. The detailed literature study paved the way to develop an efficient data placement method called DAS-DGS. Jinghui Zhang et al. [5] proposed an integer-programming-based data placement model that resolves the data placement problem. The authors collaboratively enhanced the co-location of related and localised data serving with storage capacity constraints in data centres. In addition, they proposed a LaGrange relaxation-based approach that effectively reduced the overall data access cost.

An energy-efficient cloudlet placement method (ECPM) that reduces the quantity of cloud datacentre to save energy was proposed by Mohammed Islam Naas et al. [6, 7]. Their main objective was to save energy consumption and improve the utilization efficiency of the cloud data centre. ECPM uses the K-Means clustering algorithm to gather the mobile devices and select an excellent place to keep the data in the cloud. This way, the cloudlet's current location matches the nearest aggregation centre location. Simulation results demonstrate that ECPM is an energy-efficient data placement strategy for Fog computing infrastructure [8, 9].

Qiang Xu et al. [10] proposed using an optimal data placement matrix to find an evolved genetic algorithm-based data placement technique. Qiang Xu chose the mathematical model to schedule data between data centres. They considered the effects of data access frequency and history. Because the calculations for frequency are not constant, the data placement must be modified, which increases the cost of effective data management. With the ability of the Genetic Algorithm, generations produced better approximate solutions and obtained better data placement.

The experimental findings demonstrate the effectiveness of genetic algorithms in determining approximative optimal data placement and decreasing data scheduling between data centres.

Jun Wang et al. [11] proposed a data placement scheme called Data-gRouping-Aware that scrutinizes data access from system log files. It captures runtime data grouping patterns and distributes the grouped data among data centres. The first phase in the proposed method learns data grouping from system logs, the second phase clusters the grouped data, and the third phase re-organizes data layouts to achieve the maximum parallelism per group subjective to load balance. Experiments carried out with MapReduce applications proved to improve overall performance.

Dong Yuan et al. proposed a matrix-based k-means clustering strategy for data placement in scientific cloud workflows [12]. This method involves two stages: grouping existing datasets in k centres and clustering the new datasets in appropriate data centres. The suggested technique can successfully reduce data movement during execution, according to experimental simulation. The Jade technology, created by Hao Qian et al. [13], allows Android applications to integrate complex energy-aware computation migration functionalities.

In addition to proposing an estimation technique to choose cloud resources for migration automatically, Xing Chen et al. [14] also offered a design pattern to allow applications to perform computational migration on demand. The COSMOS system was created by Cong Shi et al. [15] to offer computational migration for mobile devices. Due to COSMOS' effective management of the cloud resources used for request migration, the efficiency of transferring requests from mobile devices has been improved, and the overall cost of sending each request to the provider has been reduced.

Based on assessments of short-term green energy forecasting for a data centre scale by Aksanli et al., overall energy usage was decreased [16]. Jiao et al. [17] proposed a technique for optimizing multi-objective data placement across clouds for socially aware services by leveraging graph cuts. Xueli Huang et al. [18] plan included a one-to-one mapping function for image encryption and a method for effectively achieving image data privacy over hybrid clouds.

Wang et al. discussed several approaches to securing data in hybrid clouds and intercloud authentication models [19]. Abrishami et al. [20] proposed an approach to scheduling that preserves data privacy while reducing costs and accommodating user limitations. The flaws and difficulties of protecting IoT devices and their interactions with cloud and enterprise applications were examined.

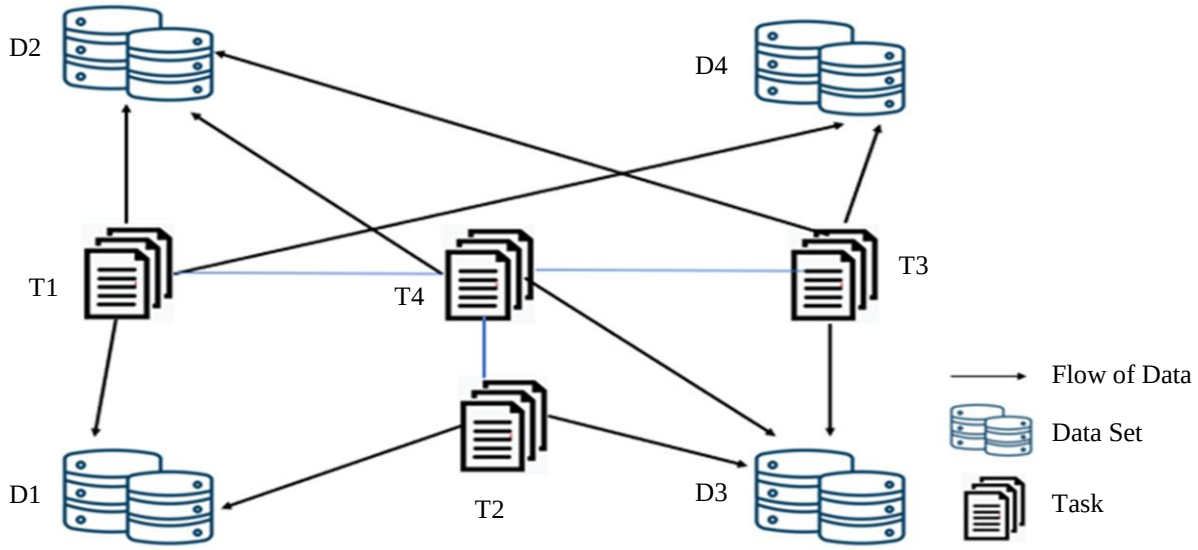

Fig. 1 Sample dataset-task utility

3. Proposed Methodology

3.1. Background & Motivation of the Proposed Work

Social media has driven the generation of data mountains many organizations use to develop their business. Similarly, data is generated at every nook and corner of the world. The researchers carry out Data Analysis to extract useful patterns out of it. Here comes a question about data storage.

Usually, the data is kept highly confidential within the organization's Research and Development team to work on it. If another team from any other part of the world needs the same data resources, they have to contact team 1 requesting data. Team 1 will now check their data repositories to find whether the data they have would satisfy Team 2 requirements. If satisfied, the data has to be transferred through hard disks to the team 2 location. Data resource sharing in this manner seems tedious and inefficient, which is not recommended. This issue is resolved with the advent of cloud computing technology.

With minimal effort or involvement from service providers, cloud computing offers on-demand networks, servers, storage, software, and data services over the Internet. As a result, Resources are thereby more easily accessible, powerful, and affordable than ever before. For example, Group 1 data can be accessed/stored by any cloud data centre. The cloud platform's data sources make their services available to anybody and anywhere online. A cloud data management system controls all cloud-based data resources. This motivated researchers to use the data sources that were available. The data centres of a cloud platform must be able to store data from other users as well. This situation may arise when a particular data stored in d1 is accessed very frequently by another data centre, d2; then it is loyal to store that data in d2 instead of d1. This made the Internet and business applications drastically move the large data centres with massive storage capacity.

As a result, numerous complex resource management issues exist, such as i) the high processing power needed to run these systems. ii) Data sources are currently kept in several data centres, and data centres are connected to restricted bandwidth. iii) Occasionally, a process might require significant bandwidth to transfer data across data centres. Data movement becomes highly tedious due to the bandwidth constraint. Moving data from one data centre to another is very costly than placing them in a location. The above problem is even faced by a popular cloud system Amazon EC2 [22], with limited bandwidth [23]. These facts show that data movement is not easy. A solution for this is to place frequently accessed datasets to store in the current data centre to minimize the data movement during execution. Often data are more flexible than storage due to user independency. Service Oriented Architecture (SOA) is implemented in a cloud platform that provides dynamic cloud services where the users are unaware of the infrastructure.

3.2. Proposed Work

The data placement method proposed in this work involves two stages. Stage 1 involves placing the existing datasets. Stage 2 deals with placing the newly generated data into its reasonable data centre during the runtime. The work mainly deals with existing data already stored in databases and new data that might be generated during the execution or satellite streaming data. Due to bandwidth restrictions, moving data between data centres can be insignificant. Many tasks will access a few datasets simultaneously, while others will be sparsely accessed.

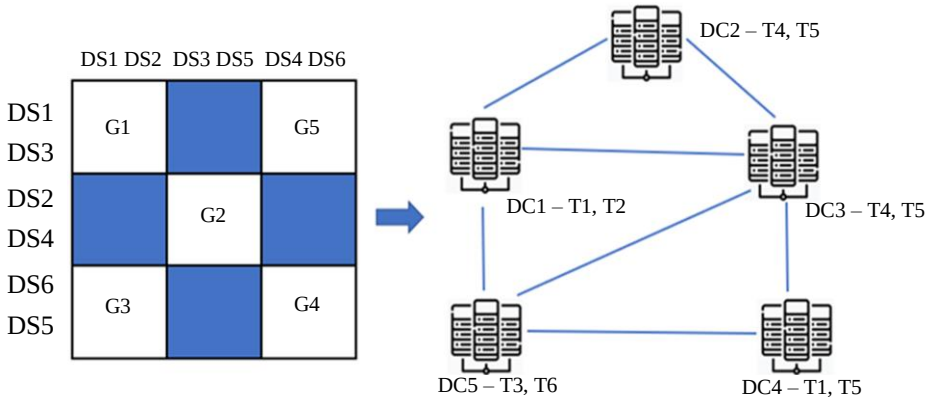

Fig. 2 Existing data placement strategy

Frequently accessed datasets should be placed in the same data centre to avoid data movement. Step1 of this work uses a strategy of access similarity among the datasets and groups them in a Data Access Matrix (DAM). The similarity between the two datasets is found by finding the total number of common tasks that use the datasets. Step 2 deals with placing the newly generated dataset into the relevant data centres.

The placement strategy for existing and new datasets will differ since the new datasets can be generated dynamically at any time during the execution process. So, reserving a space in a data centre for a new dataset will not be valid and predetermining the data centre for the dynamic dataset is also not a good solution due to the size constraint of the dataset.

Figure 1 illustrates a sample Instance of tasks utilizing the datasets. In Figure 4, four tasks are shown utilizing four datasets. Datasets are represented by D, and tasks are represented by T. Each Dataset d1 is accessed by task 1, and task2.d2 is accessed by task1, task3 and task4. Dataset d3 is used by task2, task3 and task4. D4 is utilised by task 1 and 3. The representation shows that Datasets d2 and d3 are commonly accessed by Task 3 and Task 4. So, it is reasonable to place d2 and d3 in one data centre.

3.3. Data Placement Strategy for Existing Datasets

Initially, the Access Similarity of the available datasets is calculated to form DAM, where the Access similarity AS_{ij} of d_i and d_j is calculated by the total number of tasks that uses both d_i and d_j.

$$\text{Access Similarity}(AS_{ij}) = \sum(t_i \cap t_j) \quad (1)$$

Where t_i represents all the tasks that utilise $d_i t_j$ represents all the tasks that access d_j.

The total number of datasets, n x n is represented by a matrix called DAM. Figure 2 displays the DAM matrix for the five datasets (d1 to d5). The column DAM in the matrix below indicates the total number of times the datasets have been accessed together.

$$\text{DAM (d1…..d5)} = \begin{bmatrix} 3 & 2 & 1 & 1 & 3 \\ 1 & 2 & 3 & 3 & 1 \\ 2 & 1 & 2 & 3 & 2 \\ 1 & 1 & 1 & 1 & 3 \\ 2 & 2 & 1 & 3 & 1 \end{bmatrix}$$

Bond Energy Algorithm (BEA) is currently used to turn DAM into Clustered Data Access Matrix (CDAM). The BEA was proposed by McCormick, Schweitzer and White [24]in 1972. By finding an adequate solution, this algorithm reduces the processing cost [25]. In distributed database systems, it is frequently used for vertical table partitioning of large tables. [26]. BEA groups the data sets in the matrix with similar values by permutating rows and columns. Input given to BEA is DAM and output produced by BEA is CDAM, where the values are grouped together based on their access.

$$\text{CDAM (d1….d5)} = \begin{bmatrix} 3 & 3 & 2 & 1 & 1 \\ 3 & 3 & 2 & 1 & 1 \\ 1 & 1 & 2 & 2 & 2 \\ 1 & 1 & 1 & 2 & 3 \\ 1 & 1 & 1 & 2 & 3 \end{bmatrix}$$

The next step is to divide CDAM into two divisions using Division Measurement (DM). Each division has higher and lower access similarity with the datasets in other divisions. This is achieved by using the equation 2. Division 1 ranges from 1……r, Division 2 ranges from, r+1 ….n.

$$DM = \sum_{i,j=1}^{r} CDM_{ij} * \sum_{i=r+1,j=r+1}^{n} CDM_{ij} - \left(\sum_{i=1,j=r+1}^{r} CDM_{ij}\right) \quad (2)$$

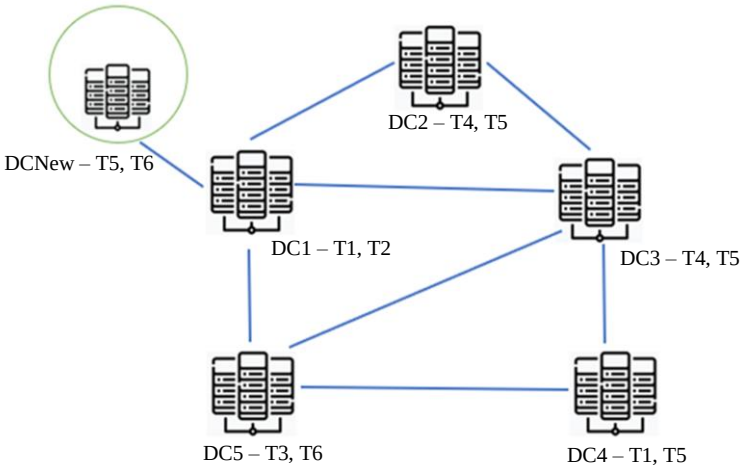

Fig. 3 Dynamic data placement strategy

This process results in two new matrices, namely CDAM-F and CDAM-S. The dataset's size is denoted by SDS_i and size of the data centre is denoted by DC_j and is represented as SDC_j. Now, the divided matrix can be placed in the data centres. Each and every data centre is set up with a threshold percentage ($SDC_j \times t$ %) to store the datasets since it is not optimal to fill the data centre to its maximum storage capacity. This is because we might have a situation to distribute the dataset generated during the run time in the data centre. Suppose the division is too large to fit into the data centre. In that case, the division process is repeated using equation 2 to divide the matrix so that the divided matrix can fit in the data centre.

This recursive process is performed until all the existing datasets are distributed among the data centres. Algorithm 1 illustrates the steps of this operation, and Figure 2 presents the current data insertion technique.

Algorithm 1 : Existing datasets Placement Algorithm

Input : Existing Datasets (DS) and Data Centres (DC)
Output : Data Centres with Initial Data Sets

Step 1 : For every $DS_i \in DS$, find the access similarity and form DAM by finding the the total number of tasks accessing DS_i.
Step 2 : Transform DAM to CDAM by using Bond Energy Algorithm.
Step 3 : Divide CDAM into two submatrices, CDAM-F and CDAM-S, using Division Measurement.
Step 4 : Identify the threshold percentage ($tp= SDC_j \times t$ %) of all DC to hold the datasets.
Step 5 : If the size of the submatrix fits the data centre, place the submatrices there; if not, divide the submatrix recursively until it fits the data centre.
Step 6 : Data centres with initial datasets.

The distribution of existing data sets in the datacentres is shown in Figure 2. The first image shows the grouping of all the datasets visited by comparable tasks the same number of times, after which the groups are deployed in the datacentres.

3.4. Data Placement Strategy for Dynamically Generated Datasets

This section the distribution of dynamically generated datasets among data centers involves several steps. Let's break down the process:

3.4.1. Identifying the Ready-To-Execute Task

This step involves determining which process or task is ready to be executed. A process becomes ready when all the datasets required for its execution are readily available in the data centers.

3.4.2. Generating New Datasets during Execution

While a process is being executed, it may generate new datasets. These datasets can be used by subsequent states of the same process or by entirely new processes.

3.4.3. Distributing New Datasets among Data Centers

When a new dataset (dsnew) is generated by a process (p), it needs to be distributed among the available data centers (k). The distribution is typically based on the concept of access similarity.

3.4.4. Calculating Access Similarity and Adding to DAM

Access similarity refers to the similarity between the new dataset (dsnew) and other existing datasets. The access similarity is calculated, and the results are added to the DAM (Data Access Matrix). The DAM keeps track of the access patterns between datasets and tasks.

It's important to note that the specific algorithms and methodologies for calculating access similarity and distributing datasets can vary depending on the system

architecture and requirements of the application. The equations and algorithm you provided describe the process of calculating access similarity and determining the placement of a newly generated dataset (ds_{new}) in a data center

$$(AS_{newi}) = \sum(t_i \cap t_n),), \text{ where } i=1\ldots\ldots n \quad (3)$$

The access similarity (ASnewi) between the new dataset (dsnew) and an existing dataset (t_i) is calculated as the intersection (represented by ∩) between the two datasets. This equation is used to calculate the access similarity of dsnew with all existing datasets (ti) in the system.

$$\text{Access Similarity } (AS_{newj}) = \sum_{AS_m \in dc_j} AS_{newm} ,$$
$$\text{Where } j = 1,2\ldots k \quad (4)$$

The access similarity (ASnewj) of the new dataset (dsnew) with a data center (DC_j) is calculated as the sum of access similarity (ASnewm) of dsnew with all the datasets (AS_m) in the data center DC_j. This equation is used to determine the access similarity between the new dataset and each data center. The access similarity (AS_{newj}) values obtained from Equation (4) indicate the suitability of placing the newly generated dataset (dsnew) in a specific data center (DC_q). To determine the placement, a constraint related to the storage capacity of the data center is considered. The variable z represents the percentage of available space in the data center (100 - tp). If the size of the new dataset (SDSnew) is less than the available space (z), then the dataset can be placed in the data center DC_q. Otherwise, if SDSnew exceeds z, an alternative data placement strategy is used.

The algorithm 2 mentioned refers to the overall data placement process for dynamic datasets. It likely includes steps for iterating through the available data centers, calculating access similarities, checking storage capacity constraints, and determining the optimal placement location for the newly generated dataset.

Algorithm 2 : Dynamic dataset Placement Algorithm

Input : Newly generated dataset
Output : Data Centres with existing and new datasets

Step 1 : Identify the tasks.
Step 2 : Calculate the Access Similarity of the new dataset AS_{newi} and other datasets.
Step 3 : Update DAM with the new dataset.
Step 4 : Calculate the Access Similarity of a new dataset and all K data centres to find a data centre((DC_q) with high access similarity.
Step 5 : Find the storage space(z) available in DC_q.
Step 6 : Place the new dataset in the data centre(DC_q) if SDS_{new}<z; otherwise, alter the threshold percentage of the Data Centre in Algorithm 1 and repeat the process.
Step 7 : Data centres placed with new datasets.

In Figure 3, the newly created dataset is placed in the data centre with the highest access similarity by calculating the access similarity of the new dataset with all the datasets in the chosen data centre.

4. Experimental Results and Analysis

4.1. Simulation Environment

A data access and storage platform is created for a demo application to test the proposed DAS-DGS's effectiveness. There are 15 Dell Power Edge T410 servers that make up this platform. Each has 16GB DDR3 memory, 8 Intel Xeon E5606 CPUs running at 2.13 GHz, and a 2TB SATA hard drive every server act as datacentres. Users can send data and compute through a demo application that dynamically generates datasets. This simulation environment is developed like the environment developed in [27].

4.2. Simulation Result & Analysis

Compared to other methods already in use with the same configuration, the suggested method DAS-DGS performance is evaluated in this subsection. The existing methods used for comparison are Data-g Rouping-Aware(DRAW) [28], Monte Carlo algorithm (MCA), First Fit Decreasing-Privacy Reservation (FFD-PR) and IoT Oriented Data Placement (IDP) [29].

DAS-DGS tends to obtain multiple objectives: average resource utilization reduces access time, reduces energy consumption, and minimizes computational cost. Fifty experiments were conducted to converge each dataset scale, and multiple solutions were obtained. The utility function is constructed with Simple Additive Weighting (SAW) and Multiple Criteria Decision Making (MCDM) 320 to find the optimal result derived from Figure 5 compares the average resource usage of four existing approaches with the proposed DAS-DGS. The proposed method reaches better average resource utilization. The average resource use of the suggested strategy gradually rises as the number of datasets also increases. IDP results appear to be less significant than DAS-DGS equation 5.

$$V`(ci) = \frac{1}{3}\cdot\frac{U(ci)-U^{min}(x)}{U^{max}(x)-U^{min}(x)} + \frac{1}{3}\cdot\frac{T^{max}(x)-T(ci)}{T^{max}(x)-T^{min}(x)} + \frac{1}{3}\cdot\frac{E^{max}(x)-E(ci)}{E^{max}(x)-E^{min}(x)} \quad (5)$$

Where the three objective functions' respective fitnesses for the data placement method ci are U (ci), T (ci), and E (ci). Figure 4 clearly shows that multiple solution sets are obtained in each dataset scale using the SAW and MCDM results. According to Figure 4a, for 200 datasets, solution 1 is the desired outcome. For 400, 600, and 1000 datasets,

respectively, solution 1, solution two and solution 1 are the desired outcomes. The results show that the utility function can generate a suitably balanced result for an incomplete dataset. Figure 5 compares the average resource usage of four existing approaches with the proposed DAS-DGS. The proposed method reaches better average resource utilization. The average resource use of the suggested strategy gradually rises as the number of datasets also increases. IDP results appear to be less significant than DAS-DGS. Figure 6 and Table 1 contrasts the proposed DAS-DGS's average access time with the current approaches. IDP, DRAW, MCA, and FFD-PR. The figure shows unequivocally that DAS-DGS has a shorter average access time than other approaches. As the number of datasets increases correspondingly rises, the average access time.

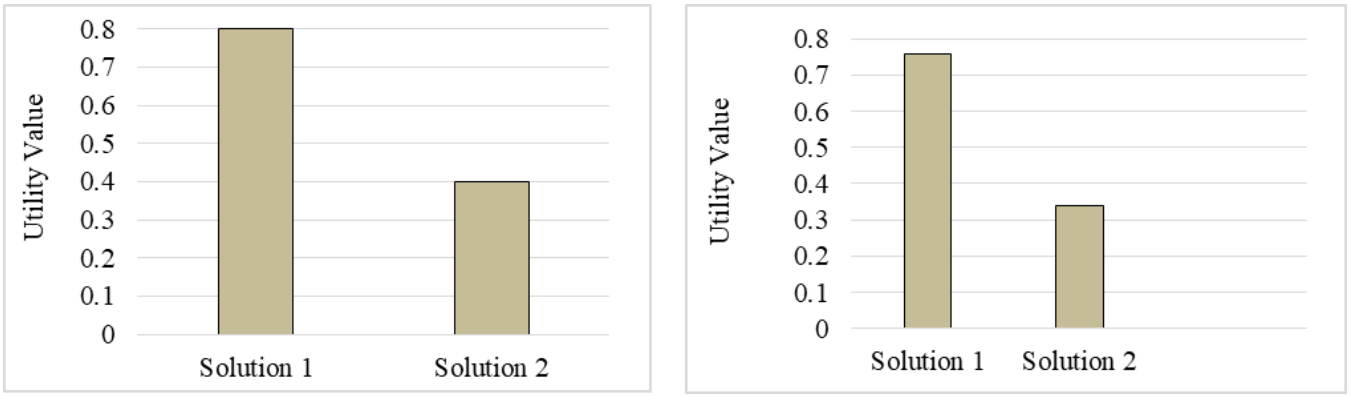

Fig. 4a Multiple sets of answers from each dataset scale

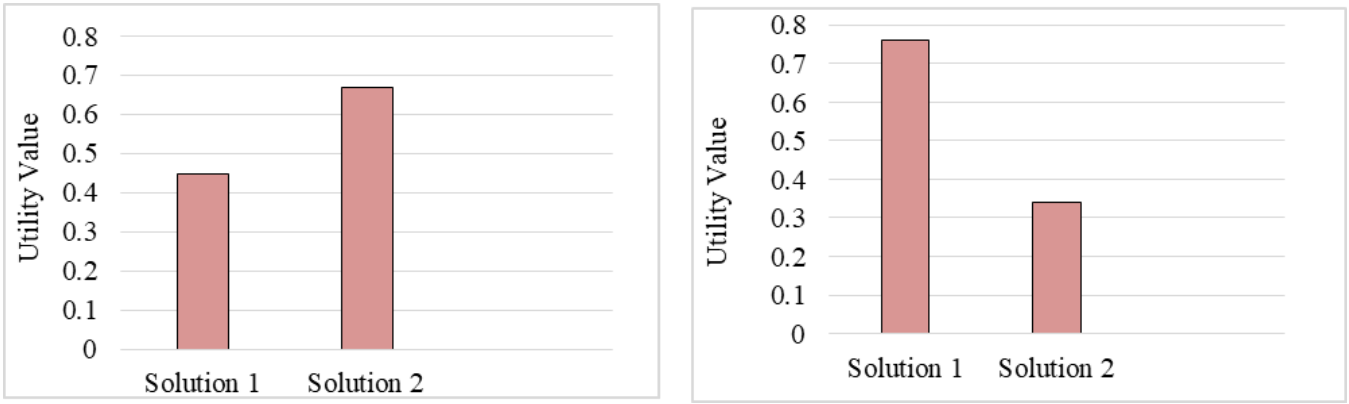

Fig. 4b A comparison of utility values using various data set scales using DAS-DGS systems

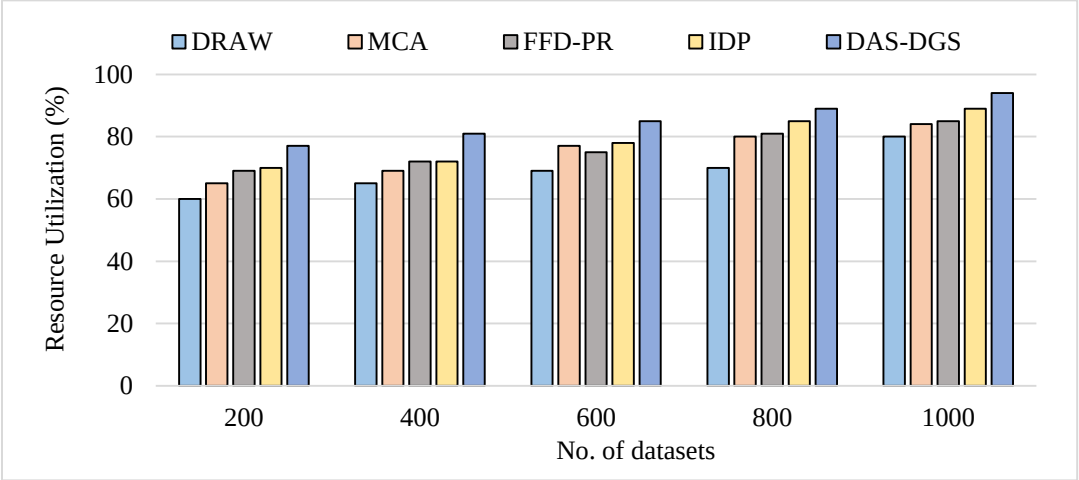

Fig. 5 The average utilization of resources by existing and proposed DAS-DGS

Table 1. Comparison of average access time by existing and DAS-DGS

No. of Dataset	Access Time (S)				
	DRAW	MCA	FFD-PR	IDP	DAS-DGS
200	4.5	4	5	5	2.4
400	5	5.5	5.3	6	4
600	6.2	6	6	7	4.6
800	7	7.5	6.4	8	5
1000	8.3	8	7	9	5.65

Table 2. Comparison of data access cost between DAS-DGS and other methods

Data Size	Cost				
	DRAW	MCA	FFD-PR	IDP	DAS-DGS
1	0.92	0.96	0.85	0.8	0.76
1.1	0.94	0.96	0.9	0.82	0.79
1.2	0.94	0.96	0.92	0.84	0.75
1.3	0.96	0.96	0.82	0.85	0.75
1.4	0.97	0.96	0.92	0.83	0.73
1.5	0.98	0.96	0.82	0.88	0.7

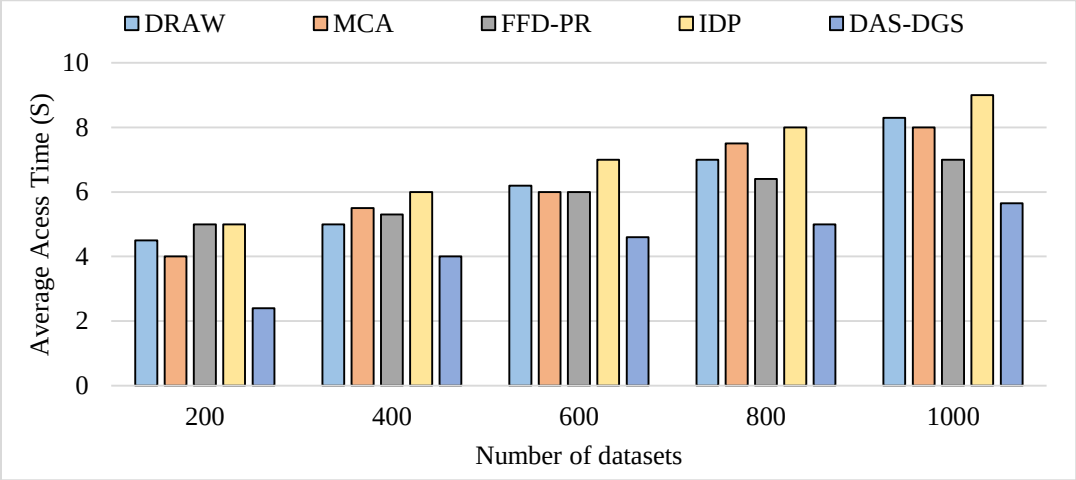

Fig. 6 The average access time by existing and DAS-DGS

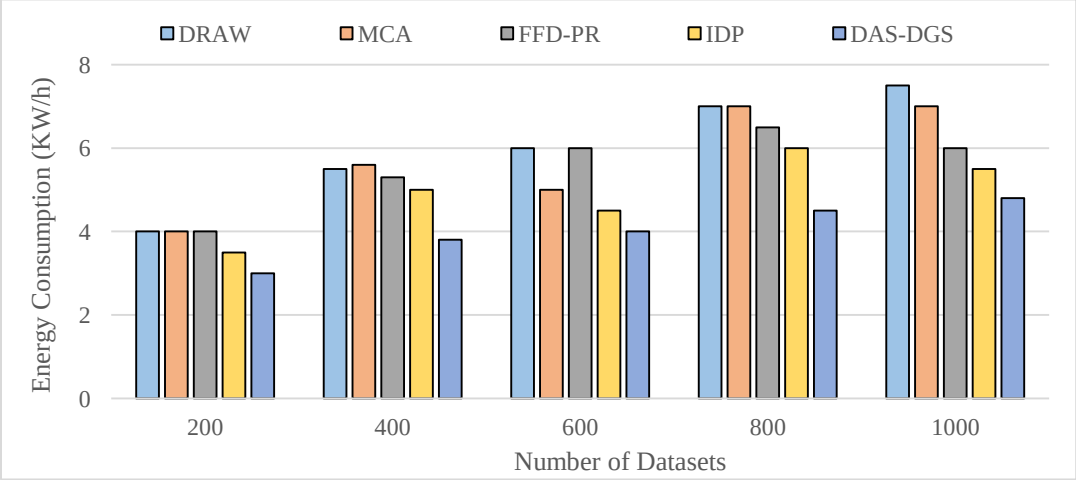

Fig. 7 The energy consumption by existing methods and DAS-DGS

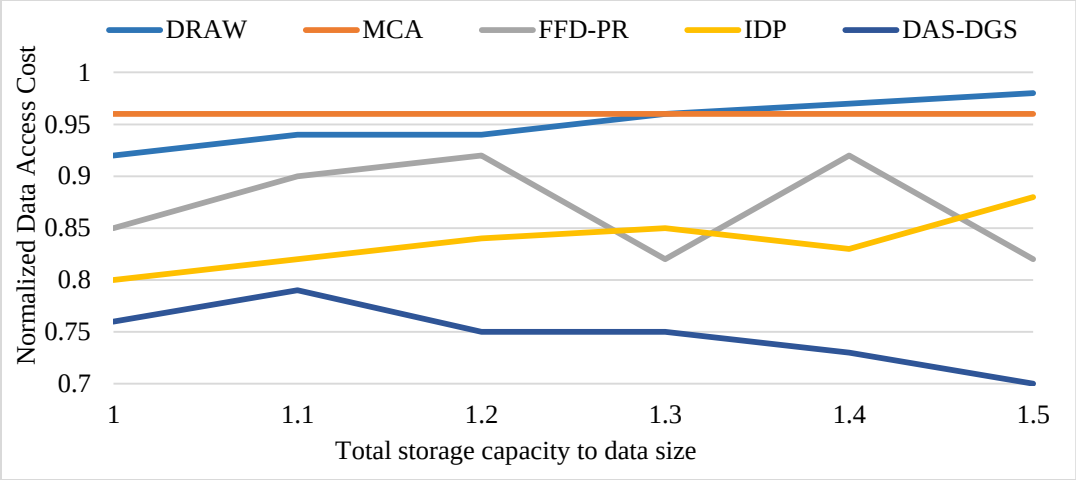

Fig. 8 Data access cost between DAS-DGS and other methods

This might be the case because more datasets appear to be completed utilizing the hosts, but many datasets cannot be put on a small number of hosts. Figure 7 shows the energy consumption by existing DAS-DGS and other methods, DRAW, MCA, FFD-PR and IDP. It is evident from the figure that the energy consumed by DAS-DGS seems to be less compared to other methods. Data access costs between the proposed DAS-DGS and other existing methods are shown in Figure 8 and Table 2. The cost value here is normalized. DAS-DGS exhibits the lowest access cost compared to other approaches since it considers the relationships between the data items in the placement strategy. IDP seems to be closer to DAS-DGS in terms of access cost. MCA is much inferior in access cost compared to all other methods. This clearly shows how effective the proposed DAS-DGS is in reducing access costs.

5. Conclusion and Future Work

This paper focussed on placing datasets effectively in the cloud environment to avoid unnecessary data movement. The proposed DAS-DGS allocates the existing data automatically to the appropriate data centres based on an access similarity strategy. In addition to placing the existing datasets, the data sets generated dynamically by the executed processes are also placed in reasonable data centres efficiently during the run time. The experimental simulation shows that the proposed DAS-DGS outperforms other currently used approaches regarding average access time, energy usage, and data access cost. This work can be further enhanced by developing effective replication schemes that maintain the data balance and storage strategy.

References

[1] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela Hung Byers, *Big Data: The Next Frontier for Innovation, Competition, and Productivity,* McKinsey Global Institute, 2011. [Google Scholar] [Publisher Link]

[2] Lili Qiu, Padmanabhan, V. N., and Voelker, G. M., "On the Placement of Web Server Replicas," *In Proceedings of the IEEE INFOCOM, Anchorage*, pp. 1587-1596, 2001. [CrossRef] [Google Scholar] [Publisher Link]

[3] Wolf, J. L., and Pattipati, K.R., "A File Assignment Problem Model for Extended Local Area Network Environments," *In Proceedings of the 10th International Conference on Distributed Computing Systems*, pp. 554-561, 1990. [CrossRef] [Google Scholar] [Publisher Link]

[4] P. Vijitha Devi, and K. Kavitha, "A Novel Fuzzy Enhanced Black Widow Spider Optimization for Energy Efficient Cluster Communication by Optimal Cluster Head Selection in WSN," *SSRG International Journal of Electrical and Electronics Engineering*, vol. 9, no. 12, pp. 49-58, 2022. [CrossRef] [Publisher Link]

[5] Jinghui Zhang, Jian Chen, Junzhou Luo, and Aibo Song, "Efficient Location-Aware Data Placement for Data Intensive Applications in Geo-Distributed Scientific Data Centers," *Tsinghua Science and Technology*, vol. 21, no. 5, pp. 471-481, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] Mohammed Islam Naas, Philippe Raipin Parvedy, Jalil Boukhobza, and Laurent Lemarchand, "iFogStor: An IoT Data Placement Strategy for Fog Infrastructure," *In Proceedings of the 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, pp. 97-104, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[7] Amarnath, .V, Mallikarjuna, .K, Nagendra, .J, Umesha, D. M., and Nalina, .V, "Review on Energy Efficiency Green Data Centers," *International Journal of Recent Engineering Science*, vol. 5, no. 2, pp. 21-26, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[8] S. Gopinath, E. Veera Boopathy, S. Pragadeswaran, S. Madhumitha, and N. Sureshkumar, "Location based Energy Efficient Routing Protocol for Improving Network Lifetime in WSN," *SSRG International Journal of Electrical and Electronics Engineering*, vol. 10, no. 2, pp. 84-91, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[9] S. Shanmadhi, K. Sekar, and T. Dheepa, "Enhancing Energy Efficient in Fault Node Recovery for a Wireless Sensor Network," *SSRG International Journal of Computer Science and Engineering*, vol. 2, no. 4, pp. 13-16, 2015. [CrossRef] [Publisher Link]

[10] Qiang Xu, Zhengquan Xu, and Tao Wang, "A Data-Placement Strategy Based on Genetic Algorithm in Cloud Computing," *International Journal of Intelligence Science*, vol. 5, no. 3, pp. 145-157, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[11] Jun Wang, Pengju Shang, and Jiangling Yin, "DRAW: A New Data-gRouping-AWare Data Placement Scheme for Data Intensive Applications with Interest Locality," *Cloud Computing for Data-Intensive Applications*, pp. 149-174, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[12] Dong Yuan, Yu Yang, Xiao Liu, and JinJun Chen, "A Data Placement Strategy in Scientific Cloud Workflows," *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1200-1214, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[13] Hao Qian, and Daniel Andresen, "Extending Mobile Device's Battery Life by Offloading Computation to Cloud," *0049n Proceedings of the Second ACM International Conference on Mobile Software Engineering and Systems*, vol. 40, pp. 150-151, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[14] Xing Chen, Shihong Chen, Xuee Zeng, Xianghan Zheng, Ying Zhang, and Chunming Rong, "Framework for Context-Aware Computation Offloading in Mobile Cloud Computing," *Journal of Cloud Computing*, vol. 6, no. 1, pp. 1-17, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[15] Cong Shi, Karim Habak, Pranesh Pandurangan, Mostafa Ammar, Mayur Naik, and Ellen Zegura, "Cosmos: Computation Offloading as A Service for Mobile Devices," *In Proceedings of the 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 287-296, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[16] Baris Aksanli, Jagannathan Venkatesh, Inder Monga, and Tajana Simunic Rosing, "Renewable Energy Prediction for Improved Utilization and Efficiency in Datacenters and Backbone Networks," *Computational Sustainability*, vol. 645, pp. 47-74, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[17] L. Jiao, J. Lit, W. Du, and X. Fu, "Multi-Objective Data Placement for Multi-Cloud Socially Aware Services," *In Proceedings of the IEEE Conference on Computer Communications*, pp. 28-36, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[18] Xueli Huang, and Xiaojiang Du, "Achieving Big Data Privacy via Hybrid Cloud," *In Proceeding of the IEEE Conference on Computer Communications Workshops*, pp. 512-517, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[19] J. K. Wang, and X. Jia, "Data Security and Authentication in Hybrid Cloud Computing Model," *In Proceedings of the IEEE Global High Tech Congress on Electronics*, pp. 117-120, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[20] H. Abrishami, A. Rezaeian, G. K. Tousi, and M. Naghibzadeh, "Scheduling in Hybrid Cloud to Maintain Data Privacy," *Fifth International Conference on the Innovative Computing Technology*, pp. 83-88, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[21] Peter Mell, and Timothy Grance, *The NIST Definition of Cloud Computing*, National Institute of Standards and Technology, Gaithersburg, MD, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[22] Amazon EC2 Secure and resizable compute capacity for virtually any workload, [Online]. Available: http://aws.amazon.com/ec2/

[23] Huan Liu, and Dan Orban, "GridBatch: Cloud Computing for Large-Scale Data-Intensive Batch Applications," *In Proceedings of the IEEE International Symposium on Cluster Computing and the Grid*, pp. 295-305, 2008. [CrossRef] [Google Scholar] [Publisher Link]

[24] Jeffrey A. Hoffer, and Dennis G. Severance, "The Use of Cluster Analysis in Physical Database Design," *In Proceedings of the 1st International Conference on Very Large Data Bases*, New York, USA, pp. 69-86, 1975. [CrossRef] [Google Scholar] [Publisher Link]

[25] William T. McCormick, Paul J. Schweitzer, and Thomas W. White, "Problem Decomposition and Data Reorganization by a Clustering Technique," *Operations Research*, vol. 20, no. 5, pp. 993-1009, 1972. [CrossRef] [Google Scholar] [Publisher Link]

[26] N. Gorla and Kang Zhang, "Deriving Program Physical Structures using Bond Energy Algorithm," *In Proceedings Sixth Asia Pacific Software Engineering Conference*, Takamatsu, Japan, pp. 359-366, 1999. [CrossRef] [Google Scholar] [Publisher Link]

[27] M. Tamer Özsu, and Patrick Valduriez, "Principles of Distributed Database Systems," Prentice Hall, Inc., Upper Saddle River, USA, 1991. [CrossRef] [Google Scholar] [Publisher Link]

[28] Qiang Xu, Zhengquan Xu, and Tao Wang, "A Data-Placement Strategy Based on Genetic Algorithm in Cloud Computing," *International Journal of Intelligence Science*, vol. 5, no. 3, pp. 145-157, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[29] Xiaolong Xu, Shucun Fu, Lianyong Qi, Xuyun Zhang, Qingxiang Liu, Qiang He, and Shancang Li, "An IoT-Oriented Data Placement Method With Privacy Preservation in Cloud Environment," *Journal of Network and Computer Applications*, vol. 124. pp. 148-157, 2018. [CrossRef] [Google Scholar] [Publisher Link]