IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 4, 2019, accepted November 18, 2019, date of publication December 3, 2019, date of current version March 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2957436

An Artificial Bee Colony Algorithm for Data Replication Optimization in Cloud Environments

RASHED SALEM[1], MUSTAFA ABDUL SALAM[2], HATEM ABDELKADER[1], AND AHMED AWAD MOHAMED[3]

[1]Information System Department, Faculty of Computers and Information, Menoufia University, Shebin Elkom 32511, Egypt
[2]Faculty of Computers and Artificial Intelligence, Benha University, Benha 13518, Egypt
[3]Information System Department, Cairo Higher Institute for Languages and Simultaneous Interpretation, and Administrative Science, Cairo 11571, Egypt

Corresponding author: Mustafa Abdul Salam (mustafa.abdo@ymail.com)

This work was supported by Menoufia and Benha Universities.

ABSTRACT Cloud computing is a modern technology for dealing with large-scale data. The Cloud has been used to process the selection and placement of replications on a large scale. Most previous studies concerning replication used mathematical models, and few studies focused on artificial intelligence (AI). The Artificial Bee Colony (ABC) is a member of the family of swarm intelligence based algorithms. It simulates bee direction to the final route and has been proven to be effective for optimization. In this paper, we present the different costs and shortest route sides in the Cloud with regard to replication and its placement between data centers (DCs) through Multi-Objective Optimization (MOO) and evaluate the cost distance by using the knapsack problem. ABC has been used to solve shortest route and lower cost problems to identify the best selection for replication placement, according to the distance or shortest routes and lower costs that the knapsack approach has used to solve these problems. Multi-objective optimization with the artificial bee colony (MOABC) algorithm can be used to achieve highest efficiency and lowest costs in the proposed system. MOABC can find an optimal solution for the best placement of data replicas according to the minimum distance and the number of data transmissions, affording low cost with the knapsack approach and availability of data replication.Low cost and fast access are characteristics that guide the shortest route in the CloudSim implementation as well. The experimental results show that the proposed MOABC is more efficient and effective for the best placement of replications than compared algorithms.

INDEX TERMS Cloud computing, multi-objective optimization, artificial bee colony, replication, cloudsim, and knapsack problem.

I. INTRODUCTION

At present, the Cloud provides many developable services on a large scale. The most important service is payment at request for each use. The Cloud increases every day and greatly affects our daily life. It is used in fields, such as wireless sensor networks (WSN) and big data [1]–[7]. Finding the shortest route and lower cost for the knapsack problem is an important job to lower the cost of data centers (DCs) and replication placement through the Cloud. Least-cost path analysis presents a lower cost between two or more sides, and it is important in the replication placement process between DCs through the Cloud [8]. Replication technology ensures data availability, confidentiality through DCs and discovery by many sides to obtain the best performance and optimization. For example, given a failure in a data request, DCs could provide speedy access from the other sites to evoke replication. One can place a replica in another location nearer to users to decrease the load balance pressure of DCs in the Cloud [9].

To maintain the availability and general performance of the system, replicas are created and placed near to users. A replica has popularity and higher access in comparison with other files [10]. Once a particular replica is established, it increases in availability and performance, but other sites increase the load, additional fees and higher cost to users [11]–[13]. We consider replication placement through the Cloud with lower costs and shortest routes to users.

The associate editor coordinating the review of this manuscript and approving it for publication was Xiao Liu.

The knapsack problem is used to find the lowest-cost path via optimization of replication. For data replication enhancement in DCs to load balance, transferring data replication through the shortest distances will decrease the response time of the network to users [14]–[19].

ABC is a tool of AI technology and was developed by Karoboga in 2005 [20], [21]. It is used on a large scale to model the real world and solve its problems. This algorithm was inspired by nature and imitates the conduct of real bees to relentlessly improve performance. In addition to solving access problems that face users [22]–[24], ABC is used to achieve more optimization and reduce costs by replica selection and placement through DCs in the Cloud. We have chosen to perform optimization by using the ABC algorithm because there are few studies regarding this technology that uses the knapsack approach to determine and place the replica through the Cloud. ABC is used to solve the least-cost path, to achieve the lowest cost and to perform replica placement optimization through DCs. This procedure is called the Multi-Objective with the Artificial Bee Colony (MOABC) algorithm.

The main contributions of this paper are as follows. First, the least-cost path problem is solved to find the optimal placement of replicas and lower the cost via the knapsack problem. Second, the ABC feature implemented cost and distance so that the user can access and place replicas via the shortest route with a lower cost and achieve load balance through DCs. Third, the ABC algorithm is executed by DCs to calculate an optimal sequence of data replication to achieve the best least-cost path. Finally, the MOABC algorithm is applied and tested using CloudSim to investigate the best performance, the optimization of data replication and the data availability compared to various similar works.

The rest of this paper is organized as follows. Section II presents a literature review of replication strategies in the Cloud. Section III describes the proposed structure in the Cloud and the heterogeneous system in nature. Section IV explains the proposed system of MOABC. Section V and VI discuss the experimental results and configuration, along with the proposed algorithm's performance and a comparison to other algorithms. In section VII, conclusions and recommendations for future work are presented.

II. RELATED WORK

In the literature, many data replication strategies have a vital role in distributed systems of the Cloud, as indicated in the following works.

Q. Xu et al. presented scalable load balancing for metadata server clusters in Cloud-scale file systems. A new methodology for parallel download via a network was achieved through a system file service and temporary storage for data replication. It aims to prevent the system from choking, and facilitates data migration from the first stage to the final stage in parallel for download and reduction in rounds via the network [25].

S. Long et al. suggested a Multi-objective Optimized Replication Management (MORM) strategy with data replication and low energy in the Cloud. Various criteria are represented including mean file unavailability, mean service time, load variance, energy consumption and mean access latency. Thus, data replication placement and storage between nodes is based on those five suggested objectives. Through these five objectives, the best replication in the Cloud is achieved [26].

Q. Wei et al. designed a Cost-effective Dynamic Replication Management (CDRM) algorithm to optimize and available data replication among DCs for users. This is used for a mathematical model to minimize replicas, thereby reducing costs and achieving a high availability and dynamic redistribution between the nodes in the Cloud. However, heterogeneity is natural on the Cloud, and thus, it is possible to change the homogenous position replicas upon user request [27].

E. Edwin et al. presented an Efficient and Improved Multi-Objective Optimized Replication Management (EIMORM). Many cost and energy aspects are reported in addition to greater data availability. Multi-objective optimization is used to improve the cost between DCs; it can also improve it by the knapsack problem concept and load balancing [28].

J. Wu et al. proposed a Service-Based Application (SBA) model based on the network star topology to place replication and communication between nodes. It also uses Multicast Growth Codes (MCGC) via the network by PC, which allows the proposed system to be more efficient in replication placement between nodes. Additionally, it should be noted that the proposed system is static, and the site has to be determined for replication placement before SBA operation. The proposed method was not ideal and requires further modification from static to dynamic in the future. It should be scalable and have more service data replication components [29].

R. Xiong et al. suggested a Snake-Like Data Placement (SLDP) strategy to create a heterogeneous cluster environment in Hadoop Distributed File System (HDFS). Alternately, this algorithm depends on node division in different storage levels of Virtual Storage Tiers (VSTs) and, at the same time, places the data among nodes. According to the hotness location, moreover, SLDP uses replicas to keep it in disk space. It also controls power very effectively [10].

C. Afzal et al. suggested Proactive Replica Checking for Reliability (PRCR), which can reduce the storage cost of similar replicas and reliability in the replication factor. The proposed algorithms reduce the storage space from one-third to two-thirds, thus achieving storage cost reduction [30].

I. Casas et al. presented Balanced and file Reuse-Replication Scheduling (BARRS), a symmetrical way to reuse determine workflows and data replication access in a Cloud. Workflow (WF) tasks achieve balance and use replication techniques to optimize data amounts through execution times and require that tasks between users are conducted in the best way. This algorithm analyzes the best workflow through task execution time and file sizes in the Cloud [31].

A. Spivak et al. suggested storage tier-aware replicative data reorganization with prioritization. An optimized algorithm was proposed to restructure data in the Cloud, through the use of replication and the transfer of data via the network. A smart calculation algorithm is allocated regarding availability, authenticity, and task performance time. Alternately, storage operations attempt to solve it in the proposed Genetic Algorithm (GA) and Categorical Genetic Algorithm (CGA) [32].

M. Gribaudo et al. suggested improvements to reliability and performance in large-scale distributed applications. A methodology of data replication, erasure code, and dividing differently sized data file blocks in different nodes was proposed. This proposal leads to a loading burden by using vast resources in the infrastructure, but resource optimization is not ideally used. This model succeeded in authenticity, data replication performance, efficiency in optimizing data availability, and reducing the time to access replication [33].

L.Zhang et al. presented a Power-aware Data Replication Strategy (PDRS). A replication strategy to have access data and calculate used power was employed by the 20/80 rule, where 80% of the data accesses 20% of the storage space so that the data replication space has a rate with a high effect of repeated access. Data nodes are divided into two parts; the first part contains hot nodes, and the second part contains cold nodes. The first part stores a hot data replication amount that can be accessed to ensure the Shadowbase Audit Log (SAL) in cases of activity in the system. However, the second part stores a large amount of inaccessible cold data which leads to a decrease in costs, power consumption and balance between cluster node processes [34].

B. Spinne et al. suggested Resilient Application Placement for Geo-distributed Clouds (RAPGC). Three data replication algorithms use AI techniques. First, the GA is used with varoptimization and second, metadata heuristics are used, called VARGA and VARSUB, and according to the optimal first algorithm in optimality process, replication is achieved in the best way. However, for the other two ways, a distribution unit uses the GA technique to distribute the data replication population. Third, the last algorithm provides central and rapid access to the best operations to solve the problem 28% [35].

N. Mansouri et al. suggested Prefetching-aware Data Replication (PDR). The connection is between data replication and accessing data from farther locations that are more popular. According to the suggested strategy, fuzzy logic is used to obtain the data replication. It is based on four factors: number of access, cost of replication, last accesses time for the replica and data availability [36].

N. Saadat et al. proposed a new Dynamic Data Replication Algorithm (PDDRA) based on Virtual Organization (VO). This algorithm is based on similar considered data replications and procures data replication from various locations through access time, consumption, cost and bandwidth of the network. The PDDRA consists of three basic stages: storing file access patterns, requesting a file performing a replica and prefetching in the data grid [37].

Discussion: Based on previous studies using ABC that have ignored MOO, MOABC achieves optimal data replication, speed data transmission and cost reduction with the knapsack problem.

III. PROPOSED ARCHITECTURE FOR REPLICA SELECTION AND PLACEMENT OPTIMIZATION

This section describes the suggested structure model to share data between nodes in the Cloud to place replication. It has been used in many studies [8], [9]. The proposed model framework describes data replication access and its placement through nodes in the Cloud. Thus, to acquire the best access to selected nodes with the lowest cost and the shortest route between DCs, we rely on the studies previous groups. We used the heterogeneous system to optimize the data replication placement by using the statistical distribution among nodes in different ways. Every DC consists of different stages, such that the VM differs from DCs, and so on. At the same time, it differs in terms of RAM, CPU, PE, etc. DCs differ in terms of the cost and number of available replicas. Alternately, it has access to data replicas and the required optimization placement with a minimum distance and least-cost path. All DCs are connected hierarchically and at the same time circularly at every DC level. Therefore, access to the replica and placing the DC at the appropriate placement are important according to the guidance bees that consider the least-cost path and lower cost through the proposed system. Users are in the outward layer of the system, and they send tasks to data replication to access the best placement according to a shortest time, shortest route and lower cost via DCs in the Cloud. We apply AI technologies that perform optimization replication placement in DCs, thereby attaining replica placement at a nearer place to users with a minimal cost via DCs. The proposed system employing MOABC techniques achieves high data availability, minimizes cost and achieves maximum optimal benefit by implementing CloudSim as shown in Fig. 1.

Each datacenter can be offered in DC form $= \{dc_1, dc_2, \ldots, dc_n\}$, where n is a set of various DCs in the Cloud. It differs in the number of DCs, given high, mid and low datacenters. The system is heterogeneous by its nature. Hosts can be represented as $PM = \{pm_1, pm_2, \ldots, pm_x\}$, where x is a various pm set existing in DCs. A virtual machine can be presented as $VM = \{vm_1, vm_2, \ldots, vm_s\}$, where s is a VM set that is placed in PM. There are two aspects of VM work first, a time-shared and second, a space-shared operation in the Cloud. Data file replication can be presented in $F = \{f_1, f_2, \ldots, f_y\}$ form, where y is a set of various data replicas that can be placed in DCs. The main storage unit is a block and can be presented as follows $B = \{b_1, b_2, \ldots, b_y\}$, where y represents a variable data replication stored set in DCs. General placed data file replication at a high datacenter is randomly distributed with other datacenters. Various values of probability $= pro(bap_j)$ can be saved

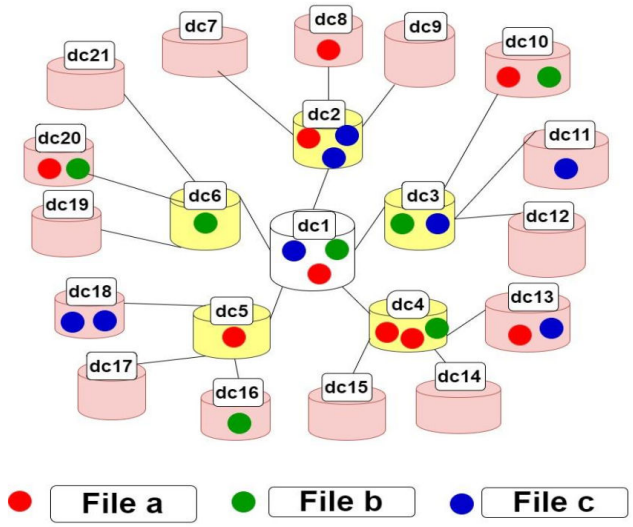

FIGURE 1. Proposed system architecture for replica selection and placement optimization.

in every DC by a replica catalog with every replica position in various DCs.

A. MULTI-OBJECTIVE OPTIMIZATION IN THE CLOUD

Multi-objective optimization is represented in the Cloud by two parts: the cost of every DC and the shortest route based on the distance among DCs. On the other hand, to achieve this goal, it is important to optimize access to data replication placement, as well as the nearest place to users. Moreover, the cost has been improved by knapsack processes for cost improvement among DCs in the proposed system, which is heterogeneous and natural.

B. DATA FILE AVAILABILITY

In general, availability can be defined as that achieved simply without any failure in nodes or data file replication. Users can access files anytime they need them [32], [33]. Data file availability and unavailability are different in a heterogeneous system. Data file availability is accessible for each user, and the failure ratio is reduced, whereas the system is more available and reliable. The concept can be calculated using (1)-(4) as follows:

$$pro(bap_j)_{high_{DC}} > pro(bap_j)_{mid_{DC}} > pro(bap_j)_{low_{DC}} \quad (1)$$

$$pro(fla_k) = \begin{cases} (1 - \prod_{i=1}^{bnr_k}(1 - pro(bap_j)_i))^{nb_k} & \text{case 1} \\ \prod_{i=1}^{nb_k}(1 - \prod_{i=1}^{bnr_k}(1 - pro(bap_j)_i)) & \text{case 2} \end{cases} \quad (2)$$

$$\overline{pro(fla_k)} = \begin{cases} 1 - (1 - \prod_{i=1}^{bnr_k}(1 - pro(bap_j)_i))^{nb_k} \\ 1 - \prod_{i=1}^{nb_k}(1 - \prod_{i=1}^{bnr_k}(1 - pro(bap_j)_i)) \end{cases} \quad (3)$$

Let the block available probability $pro(bap_j)$ be represented as follows:

$$high_{DC} = 0.9 > mid_{DC} = 0.6 > low_{DC} = 0.3 \quad (4)$$

whereas notations in equations (1)-(4) are described as follows:

$pro(bap_j)$	Probability of block availability
b	Blocks
$pro(fla_k)$	Probability of file availability
nb_k	Number of block
bnr_k	Number of replica of a data file df_k
$\overline{pro(fla_j)}$	Probability of block unavailability
na_k	Number of access task have request
$\overline{pro(fla_k)}$	Probability of file unavailability
$high_{DC}$	High data-centers
mid_{DC}	Medium data-centers
low_{DC}	Low data-centers

C. ANALYZE THE TYPE OF FRAGMENTS AND THE ACCESS TIME TO REPLICATION

The Improve Time-Based Decaying Function (ITBDF) is used to determine access data file replication, distinguishing high popularity over other files. ITBDF gives different weights to each data file during reach. The data file replication process selects and creates high popularity files compared with other data files. Previously, "data file1" has 886 accesses in total and "data file 2" has 799 accesses. Indeed, "data file 1" is better than "data file 2", but this notion is incorrect. To solve this problem, we use ITBDF with ABC, according to (5)-(7):

$$ITBDF(t_a, t_b) = e^{-(t_a - t_b)k} \quad \forall\, k \in (1, 2, 3, ...n) \quad (5)$$

$$ITBDF(t_a, t_b) = e^{-(\triangle t)k} \quad (6)$$

where, $\triangle t = (t_a - t_b)$
whereas other notations are described as follows:
t_a is the current time,
t_b is the start time,
k is increasing value,
e is the exponential function decay.

$$RF_k = \frac{\sum_{t_i=t_b}^{t_a}(na_k(t_i, t_i + 1) * ITBDF(t_i, t_a))}{bnr_k * \sum_{i=1}^{nb_k} sb_i} \quad (7)$$

whereas notations of (5)-(7) are described as follows:
na_k is the number of accesses,
bnr_k is the number of replicas,
nb_k is the number of blocks,
sb_i is the size of a block.

D. COST OF REPLICATION

The cost between different DCs is computed according to the data replica number within every DC. For access to data replica, having low cost and placement to the nearest users is important. Also, maintaining the total cost of the system within every DC through different routes via the Cloud is crucial, whether a budget is provided or not. It can be calculated using (8):

$$cost_k(dc_s) = \sum_{x=1}^{y}(cost(dc_y) * bnr_k(dc_y)) \quad (8)$$

E. MINIMUM DISTANCE OF REPLICATION

The shortest distance and the shortest route between DCs has been calculated. Thus, it is important to access the least cost path via DCs in the Cloud. Moreover, it is crucial to ensure access to the optimal data replica placement via DCs in the Cloud. It is calculated using (9) and (10) as follows:

$$Min \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_{ij} \tag{9}$$

S.T :

$$\sum_{i=1}^{n} n_i x_{i \geq k}, \quad x_i \in \{0, 1\}, \ (1 \leq i \leq n) \tag{10}$$

F. KNAPSACK PROBLEM

The knapsack problem is NP-hard. Every item has a particular value and weight. The target is to maximize the resources in the bag with the constraint such that the carrying value of the bag shall not exceed the following (11) and (12):

$$\text{maximize} \quad px = \sum_{j=1}^{n} p_j \tag{11}$$

S.T :

$$wx = \sum_{j=1}^{n} w_{ij} x_j \leq v_i \tag{12}$$

$x_j \in \{0, 1\}, \quad j = 1, 2, \ldots, n$
$p = (p_1, p_2, \ldots, p_n)$
$w = w_1, w_2, \ldots, w_n$
$i = 1, 2, \ldots, m$

Each object $j \in J$ has profit p_j and weight w_j in dimension i, where $(1 \leq i \leq m)$. Binary variable x_j indicates whether object j is included in the knapsack ($x_j = 1$) or not ($x_j = 0$). Notations of Knapsack problem are described as follows:

w_j	Weight of object j
p_j	Value of object js
px	Total value
wx	Total weight
j	Item j
p	Value vector of all item j
w	Weight vector of all item j
Set j of n	Object and knapsack with m dimension
v_i	Each dimension of the knapsack has a capacity v_i

IV. PROPOSED MODEL OF ABC-BASED ALGORITHM FOR CLOUD ENVIRONMENTS

As we mentioned in the literature review, most studies depend on mathematical models to select and place replicas. However, in our case, a heterogeneous system and AI in the form of ABC are used to find the shortest ways and reduced costs to access and place replicas across nodes. We execute the proposed system through the ABC algorithm to determine the best and shortest route to access data placed to users. It consists of three main features which are the cost, the distance and the knapsack of the process for the selection and placement of the replica at the best place via nodes. The ABC algorithm consists of three bee types: scout bees, onlooker bees and employed bees. Half are employed bees, and the other half are onlooker bees. Employed bees are responsible for using nectar resources and sending information to onlooker bees that wait in the cell. They are responsible for the quality of the appropriate site of two of the resources being used. Onlooker bees decide whether the scout bees at the new site obtain new food, depending on external and internal motives. The nectar quantity from food resources or the site corresponds with the nectar of bees. The principal steps of the algorithm can be summarized as follows:

A. INITIALIZATION OF FOOD SOURCE SITES

A search space is supposed in the environment where the cell contains sites of food resources. The algorithm randomly produces new sites of food sources that match solutions in the search space. The primary food sources' limits are produced within parameters according to (13):

$$x_{ij} = x_j^{min} + rand[0, 1)(x_j^{max} - x_j^{min}) \tag{13}$$

$i = 1, 2, \ldots, s_n, j = 1, 2, \ldots, d$

s_n	Number of food sources
d	Number of optimization parameters
$rand$	Random number in range [0,1)
x_j^{min}	Lower border in the j^{th} diminution of the problem space
x_j^{max}	Upper border in the j^{th} diminution of the problem space

B. SENDING EMPLOYED BEES TO THE FOOD SOURCE SITES

Here every employed bee is connected with one site of only food sources so the number of sites of food sources in the solution area is equivalent to the number of employed bees. At the same time, it modifies the different sites of food sources (solutions) in memory depending on new and visual information of local information and finds a food source or its nearby site. Its quality is estimated using (14) and (17):

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{14}$$

whereas notations of employed bees are described as follows:

x_{ij}, x_{kj}	Difference between parameter decrease, the perturbation on the position $_{xij}$ decrease
x_i	Neighborhood of every food source site
v_i	Food source is determined by changing one parameter of x_i
j	Random integer in the range [1,d]

k	$k \in \{1, 2, \ldots, s_n\}$ is a random chosen index that has to be different from i
ϕ_{ij}	A uniformly distributed real random number in the range [-1,1]
f_j	The cost value of the solution v_i for maximum problem
$x_i\&v_i$	Greedy selection is applied between $x_i\&v_i$
p_j	Probability value
mcn	Iteration number

The difference between the values of x_{kj} and x_{ij} decreases. In addition, the x_{ij} site decreases, so the step length decreases adaptively. Searching for the best solution within the matter search space is an important case. If the value produced by this process exceeds the definite limit, the value shall be reset to acceptable values in the case of exceeding its limit to the definite limit; these values are:

Minimize problem fitness value:

$$\text{if} \quad x_i > x_i^{max} \quad \text{then} \quad x_i = x_i^{max}, \tag{15}$$

$$\text{if} \quad x_i < x_i^{min} \quad \text{then} \quad x_i = x_i^{min} \tag{16}$$

After producing v_{ij} within the specified limits and standards in the matter, the fitness for the solution is specified by using the following formula:

$$fitness_i = \begin{cases} 1/(1+f_i) & \text{if} \quad f_i \geq 0 \\ 1 + abs(f_i) & \text{if} \quad f_i \leq 0 \end{cases} \tag{17}$$

Then, the best solution is selected depending on the fitness value that represents the nectar amount of the food sources at x_i and v_i. If the v_i source is higher than x_i regarding profit, employed bees store the new site in their memory and forget the old one. If there is no improvement in the new process, it saves the previous site in its memory, increases the counter by one in tests and offers zero otherwise.

C. CALCULATING PROBABILITY VALUES INVOLVED IN PROBABILISTIC SELECTION

After completing the search, an employee bee shares information, nectar quantity, sites and food sources with any onlooker bee in the area via a dance. There are two types of dance (dance round and waggle tail dance). The onlooker bee estimates information and nectar taken from the employed bee and selects sites and food sources that may connect with the nectar quantity. This step is an advantage in the ABC algorithm. This selection is estimated according to the fitness values in society. Selection may be made by a roulette system or other systems. The roulette system is shown in (21):

$$p_i = \frac{fitness_i}{\sum_{i=1}^{sn} fitness_i} \tag{18}$$

For the probable selection, the nectar quantity increases, and onlooker bee visits increase to those sites. It is considered an advantage of the ABC algorithm

D. FOOD SOURCE SITE SELECTION BY ONLOOKER BEES BASED ON THE INFORMATION PROVIDED BY EMPLOYED BEES

A real number is created in the range of 0 to 1 for each site. If the probable value p_i is in (21), the site is larger than the random number. Onlooker bees modify this site by using (21) as employed bees. The site is estimated by a greedy algorithm because the onlooker bee forgets the old site and saves the new one in the improvement state, or the old site stays as it is in its memory if the solution is not improved. The ABC algorithm makes a test counter and increases it by 1 each time, unless it is returned to 0. This process is repeated until the onlooker bee is distributed to all food sources.

E. ABANDONMENT CRITERIA: LIMIT AND SCOUT PRODUCTION

After employed bees and onlooker bees complete and finish the search processes in their cycle, the ABC algorithm discovers a depleted source that can be abandoned and decides to dispense with them by updating the counter during a search test counter. If the counter is larger than the control standard, then the ABC algorithm can be known by the limit; hence, this source or site is depleted and must be abandoned. The ABC algorithm that is called the limit shall be employed. Its formula is shown in (19):

$$LimitValue = (sn * mcn) \tag{19}$$

Food sites that the bee abandoned are to be exchanged with another food source that is explored by scout bees during changes that occur in the ABC algorithm and by producing a new site randomly instead of the abandoned one. Employed bees are the ones that may be scouts. If it is supposed that there are more than the counters exceeding the limit value, in this case the highest one is selected pro-grammatically. This selection appears through the following flowchart shown in Fig. 2:

F. ZIPF AND GEOMETRIC DISTRIBUTIONS

Zipf and Geometric Distributions can be compared with the data replication process distributions among datacenters. Data file replication is placed in DCs according to the users' tasks.

First: Zipf is used in a random distribution process of data file replica placements among DCs closer to users. It can be calculated by (20) and (21):

$$p(f_i) = \frac{1}{i^{\alpha}} \tag{20}$$

where $i = 1, 2, \ldots, n$ and α is a is a factor data replication distribution, $0 \leq \alpha < 1$.

Second: the Geometric distribution represents a random distribution to solve optimal data file replica placements with various parameters. It can be calculated by the following equation:

$$p(i) = (1-p)^{i-1}.p \tag{21}$$

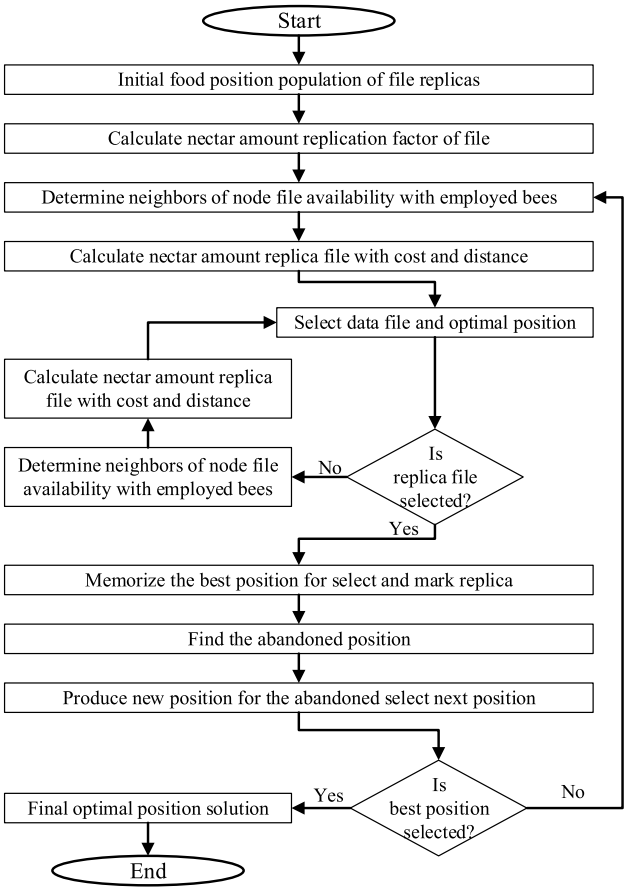

FIGURE 2. Flowchart of the proposed ABC algorithm for replication position optimization.

where $i = 1, 2, \ldots, n$ and $0 < p < 1$. A wide p represents the file replication access.

V. EXPERIMENTAL EVALUATION

A. EXPERIMENTS OF OPTIMIZATION

This section discusses the experimental results and configuration details of the data replication framework. The MOABC strategy determines and places data replicas by using the ABC algorithm with MOO, in addition to the Zipf and Geometric distributions. The proposed algorithm is compared with other algorithms. According to reduced waiting time, the knapsack problem, data availability, the speed number of data transmission and the optimal data replica placement, several experiments have been performed on CloudSim to prove the validity of the proposed system.

B. CONFIGURATION

Fig. 1 proposes 21 DCs, as used in Table 1. Every datacenter consists of hosts with the available virtual machine number; it provides a number of blocks for data availability. Thus, we create 3 different file placements in high DCs. At the same time, 1000 cloudlets are sent and chaotically performed on available DCs to request data files according

TABLE 1. Simulation parameters of configuration system.

Parameter	High DC	Mid DC	Low DC
Datacenters			
Number of DCs	1	5	15
Cost of DCs	500	300	100
No. of hosts per DC	20	30	60
Host			
Processing elements per host	12 – 16	4 – 8	1 – 4
MIPS for each processing element	1000 – 2000	500 – 1000	100 – 500
Bandwidth for each processing element	5 – 10 GB	2 – 4 GB	1 – 2 GB
Virtual Machine			
Number of VMs	200	150	120
MIPS	800	400	200
RAM	2 GB	1 GB	512 MB
Bandwidth	10 GB	2 GB	1 GB
Number of processing elements	8 – 16	4 – 8	1 – 4
Cloudlet scheduler	Time & space shared		
VM scheduler	Time & space shared		
Cloudlet			
Cloudlet task	1000 task		
Length of task	1000 – 20000		
File			
Number of files	A	B	C
Cost of replication	500	300	100
User			
Number of users	10 – 50		

TABLE 2. ABC algorithm parameters.

Parameter	Value
Number of employed bees	25
Number of onlooker bees	25
Number of scout bees	5
mcn = Max Iterations	1000

to our system. The ABC algorithm is performed based on availability, the knapsack problem, reduced waiting times and fast transmission of data files. Therefore, optimal data replica placement is based on the MOABC strategy.

C. RESULTS AND DISCUSSION

This paper focuses on verifying CloudSim to experiment with the MOABC strategy. Our strategy is proven to be superior to other strategies. A comparison between th proposed strategy and most related works is provided in Table 3.

D. FIRST EXPERIMENTS FOR CLOUDLET OPTIMIZATION

CloudSim has been used to perform the suggested MOO with ABC techniques. The experimental results show transmitted tasks over CloudSim within 3 seconds. Using the second experiment to apply the proposed algorithm is more reduced in CloudSim, whereas the ratio of 66% of the total execution time tasks shows the run is within one second. We show a comparison among the tasks, availability, average response

TABLE 3. Comparison between related work and the proposed strategy.

Strategy	Year	Availability	Load balancing	Heterogeneity	Storage cost	Knapsack problem	Distance	Least cost path	Optimal location
DCR2S [8]	2016	✓	✓	✓	✓	✓			
SLDP [10]	2015	✓	✓						
CDRM [27]	2010	✓	✓	✓	✓				
BARRS [31]	2016	✓	✓						
MORM [26]	2013	✓	✓		✓				
RAPGC [35]	2016	✓	✓		✓				
Fuzzy-FP [38]	2016	✓	✓						
PDDRA [37]	2012	✓	✓						
EIMORM [28]	2017	✓	✓						
DPRS [34]	2017	✓	✓						
PDR [36]	2018	✓	✓		✓				
ACO [39]	2016	✓	✓						
GASA [40]	2016	✓	✓						
DRSACO [41]	2013	✓	✓						
Genetic [42]	2015	✓	✓						
EFS [43]	2011	✓	✓						
Proposed	2019	✓	✓	✓	✓	✓	✓	✓	✓

time, waiting time, and transmission data over datacenters. The experimental results show that the proposed strategy enhances the optimal data replica placement on CloudSim. The experiment has been implemented within 1 second.

E. COST OF REPLICATION AND NUMBER OF REPLICAS USING THE KNAPSACK PROBLEM

Figure 3 shows the comparison between the MOABC algorithm and Dynamic Cost-aware Re-Replication and Rebalancing Strategy (DCR2S) algorithm regarding the total cost of replication when using the knapsack problem, which affects the increased replication number. Regardless, without DCR2S, when the requested numbers of data file replications increase, the total cost of replication is increased automatically. Therefore, when requesting the number of replicas, the costs of data replication increase at a high rate. Alternately, our MOABC algorithm shows cost savings within the request number of replicas by users. For comparison of the DCR2S algorithm in the process saving cost, we notice that the MOABC algorithm is superior to the DCR2S algorithm. Figure 3 depicts various scenarios about the cost of replication with the number of replicas for explaining six budget cases. There are many scenarios from Fig. 3(a) to Fig. 3(f), and we have devised many budget scenarios from 1200 to 4000. Thus, the MOABC algorithm can improve the cost of replication with the knapsack problem when users request the number of replicas.

F. SECOND EXPERIMENTS FOR THE OPTIMAL SELECTED REPLICA

Figure 4 shows the use of the MOABC, EFS and DC2RS algorithms, showing the number of requested users, as well as their influence on cost. The number of Cloudlets determines data replication, and increases availability through DCs.

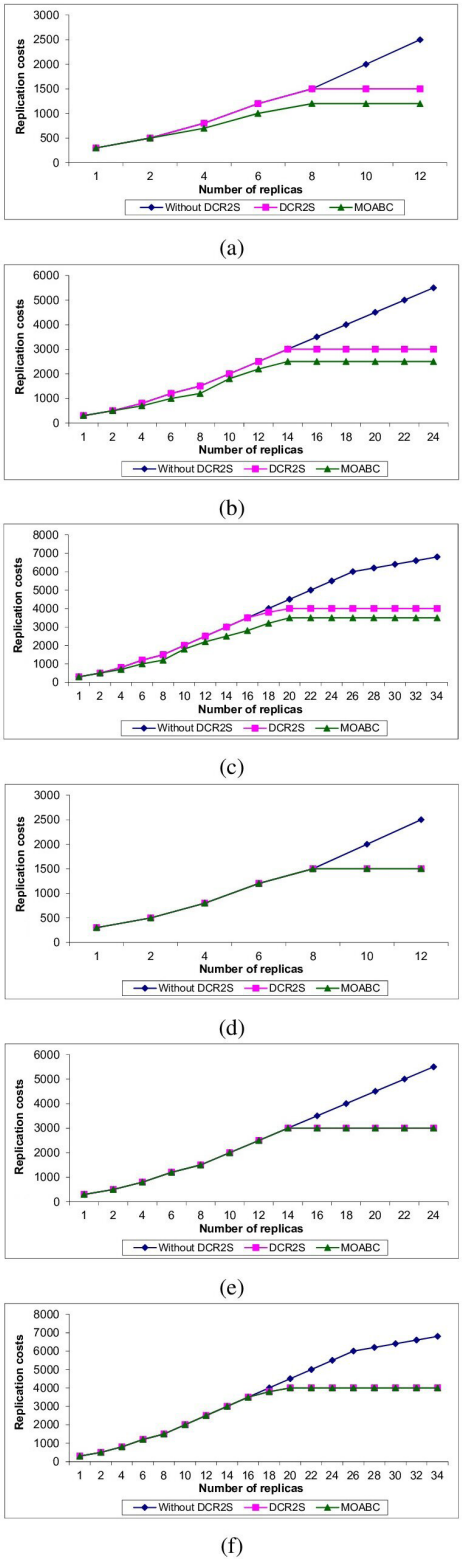

FIGURE 3. Various scenarios about cost of replication with the number of replicas for solving six types of budget cases.

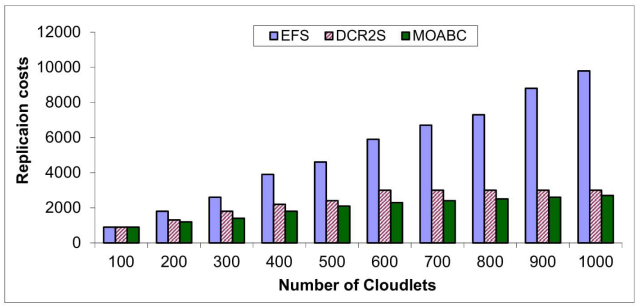

FIGURE 4. Cost of replication and number of cloudlets.

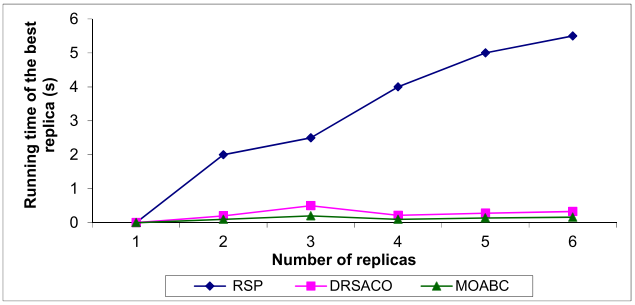

FIGURE 5. Probability of data availability and number of replicas.

Figure 5 shows access to the best data replica, which has the highest popularity through the MOABC algorithm to determine the optimal data replication. Based on ITBDF, which determines the high popularity of data replication, the proposed MOABC algorithm is superior to other algorithms when accessing the optimal data replication and probability of data availability.

G. THIRD EXPERIMENTS FOR THE OPTIMAL PLACEMENT REPLICA

This experiment shows the influence of MOABC on data replication placement among DCs, thereby accessing optimal placement by using many aspects of MOO such as distance, cost and the Zipf and Geometric distributions. We have implemented the MOABC strategy and compare it with other strategies such as the random strategy and genetic strategy. More scenarios have been conducted to execute the proposed strategy, as shown in Fig. 6, 7 and 8. The MOABC strategy proves that the data transmission process optimizes data replica placement through data nodes by a lower-cost path with reduced time. The experimental results show that our strategy is superior to other strategies.

As shown in Fig. 9, 10 and 11, MOABC is used to send users' tasks to request a number of data replications. We observe that our proposed strategy reduces the number of data transmissions and the total number of data transmissions. Our strategy works to reduce cost, distance and users' response time. Based on our results, our strategy is more effective in the data transmission process and has optimal data replica placement in DCs. The results also show that our

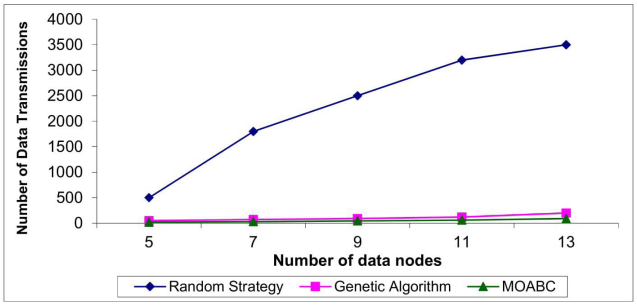

FIGURE 6. The number of data transmissions with different data nodes.

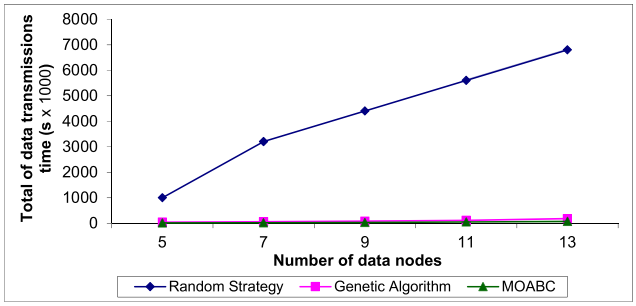

FIGURE 7. Total data transmission time and data nodes.

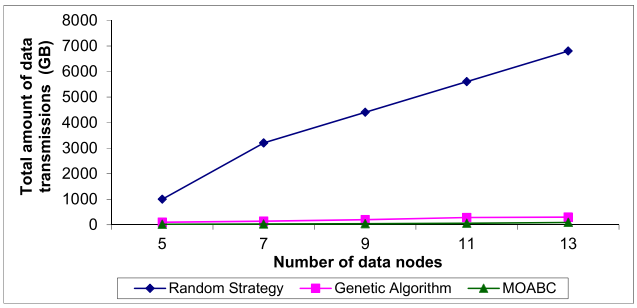

FIGURE 8. Total amount of data transmission time and data nodes.

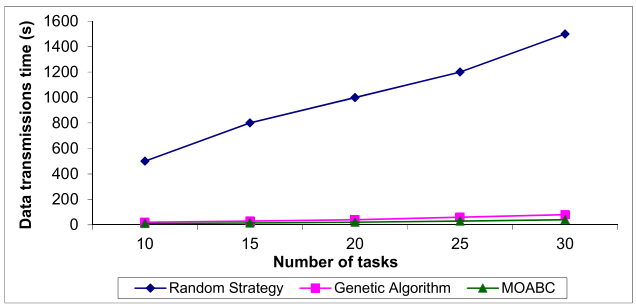

FIGURE 9. The number of data transmissions with different tasks.

strategy is superior to those of other algorithms such as the random strategy and genetic algorithm.

H. EVALUATION ANALYSIS

In Fig. 12, in general, the relation between the number of cloudlets and response time increases the cloudlets and high

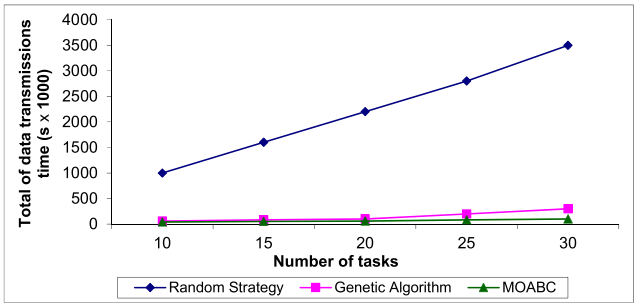

FIGURE 10. Total data transmission and number of tasks.

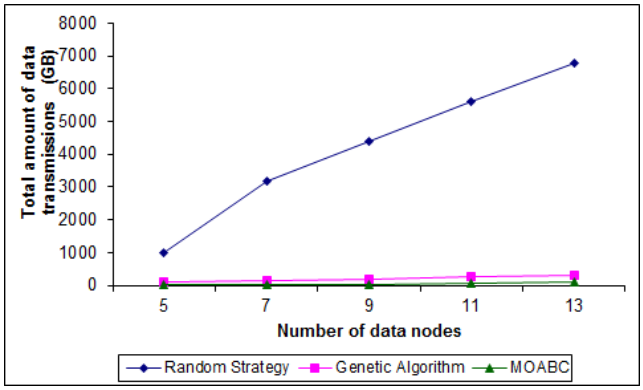

FIGURE 11. Total amount of data transmission and number of tasks.

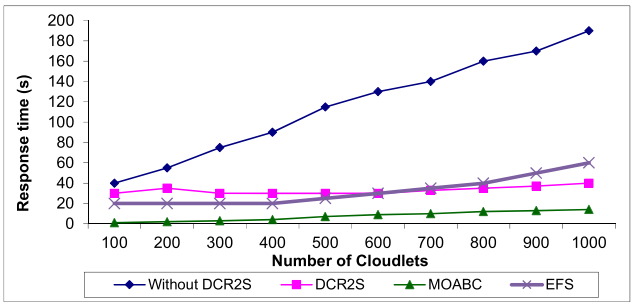

FIGURE 12. Response time for cloudlets.

cost continuously. Comparing the proposed strategy and other strategies, the cost of data replication and other tasks ceases according to the proposed users' budget. The proposed algorithm works effectively and excels when implemented with other algorithms.

Figure 13 shows a constant increase in the job number, which becomes optimal with replication selection. Based on the MOABC strategy, the number of jobs enhances the average time of tasks in the Cloud. The aim is also to improve network performance, time execution, distance and load balancing. Experimental results demonstrate that the proposed strategy is superior to other strategies.

VI. DISCUSSION

In previous analyses, the MOABC strategy has been used for the selection and placement of replicas in suitable places

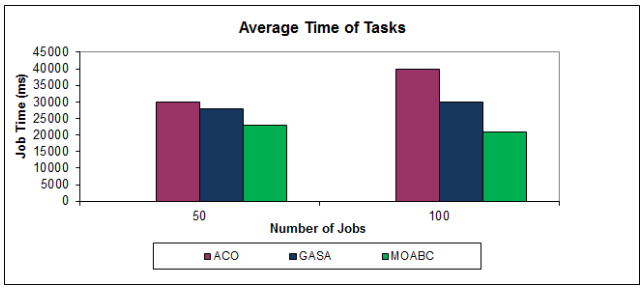

FIGURE 13. Average time between job time and job number.

based on cost, the knapsack problem, time transmission and the distance between DCs. On the other hand, the number of iterations changes the experiment from 10 to 100 bees in the results but is the acquired best global solution given increasing studies and explorations of a new research area. More optimization of every cycle builds solutions through ABC in terms of cost, availability, distance and time of data replication. Obtaining the best selected data replication and placement will be saved as the best global solution after several attempts using criteria as shown in Table 2. The proposed algorithm is superior to the other algorithms.

VII. CONCLUSION AND FUTURE WORK

Cloud computing ensures availability, cost minimization, and improved access to data. In this work, we have created a new algorithm derived from a combination ABC and Multi-Objective Optimization. Therefore, access and optimized replica placement are presented at the appropriate best place for the minimum distance and least-cost path and, for direct bees, for shortest routes in distance and lower cost. Meanwhile, the proposed MOABC algorithm has contributed fast access to data and selected the best replica placement nearest to users. Additionally, the placement optimization provides more least-cost paths, better response times and replication costs within the budget. Once the replication cost exceeds the user budget, the replication knapsack algorithm optimizes the replication cost in addition to using different statistical distributions, such as the Zipf and the Geometric distributions, to perform access optimization replication placement in DCs. The results show that the MOABC algorithm allows replicas access to the nearest route and least cost path in seconds for optimizing replica selection and placement. In addition, the MOABC algorithm is superior to other algorithms such as EFC, DCR2S, ACO, Genetic algorithm and GASA. The results show that the proposed MOABC algorithm is the most effective and efficient of the considered algorithms. The proposed algorithm has been simulated and assessed via CloudSim. The simulation results clarify the efficiency of the proposed algorithm. In future work, we aim to verify the system on the Cloud. We are also considering other topics of interest, such as applying the mixed Al techniques to ensure system availability to users, as well as fast data replication access and cost improvement on the Cloud.

REFERENCES

[1] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *Proc. Int. Conf. High Perform. Comput. Simulation*, Jun. 2009, pp. 1–11.

[2] M. Xu, A. V. Dastjerdi, and R. Buyya, "Energy efficient scheduling of cloud application components with brownout," *IEEE Trans. Sustain. Comput.*, vol. 1, no. 2, pp. 40–53, Dec. 2016.

[3] P. S. Rawat, G. P. Saroha, and V. Barthwal, "Quality of service evaluation of SaaS modeler (Cloudlet) running on virtual cloud computing environment using CloudSim," *Int. J. Comput. Appl.*, vol. 53, no. 13, pp. 35–38, 2012.

[4] S. F. Piraghaj, R. N. Calheiros, J. Chan, A. V. Dastjerdi, and R. Buyya, "Virtual machine customization and task mapping architecture for efficient allocation of cloud data center resources," *Comput. J.*, vol. 59, no. 2, pp. 208–224, Feb. 2016.

[5] B. Sharma, "Improving QoS parameters for cloud data centers using dynamic particle swarm optimization load balancing algorithm," *IJARIIE*, vol. 3, no. 2, pp. 265–272, 2016.

[6] R. Kumar and G. Sahoo, "Cloud computing simulation using CloudSim," *Int. J. Eng. Trends Technol.*, vol. 8, no. 2, pp. 82–86, 2014.

[7] Vikash, "Dynamic creation and placement of virtual machine using CloudSim," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 4, no. 8, pp. 675–679, 2014.

[8] N. K. Gill and S. Singh, "A dynamic, cost-aware, optimized data replication strategy for heterogeneous Cloud data centers," *Future Gener. Comput. Syst.*, vol. 65, pp. 10–32, Dec. 2016.

[9] D.-W. Sun, G.-R. Chang, S. Gao, L.-Z. Jin, and X.-W. Wang, "Modeling a dynamic data replication strategy to increase system availability in cloud computing environments," *J. Comput. Sci. Technol.*, vol. 27, no. 2, pp. 256–272, 2012.

[10] R. Xiong, J. Luo, and F. Dong, "Optimizing data placement in heterogeneous Hadoop clusters," *Cluster Comput.*, vol. 18, no. 4, pp. 1465–1480, 2015.

[11] Y. Mansouri and R. Buyya, "To move or not to move: Cost optimization in a dual cloud-based storage architecture," *J. Netw. Comput. Appl.*, vol. 75, pp. 223–235, Nov. 2016.

[12] D. Poola, K. Ramamohanarao, and R. Buyya, "Enhancing reliability of workflow execution using task replication and spot instances," *ACM Trans. Auton. Adapt. Syst.*, vol. 10, no. 4, p. 30, 2016.

[13] P. Matri, A. Costan, G. Antoniu, J. Montes, and M. S. Pérez, "Towards efficient location and placement of dynamic replicas for geo-distributed data stores," in *Proc. ACM 7th Workshop Sci. Cloud Comput.*, 2016, pp. 3–9.

[14] I. De Falco, E. Laskowski, R. Olejnik, U. Scafuri, E. Tarantino, and M. Tudruj, "Extremal optimization applied to load balancing in execution of distributed programs," *Appl. Soft Comput.*, vol. 30, pp. 501–513, May 2015.

[15] E. Y. Daraghmi and S.-M. Yuan, "A small world based overlay network for improving dynamic load-balancing," *J. Syst. Softw.*, vol. 107, pp. 187–203, Sep. 2015.

[16] W. Li, Y. Yang, and D. Yuan, "A novel cost-effective dynamic data replication strategy for reliability in cloud data centres," in *Proc. IEEE 9th Int. Conf. Dependable, Autonomic Secure Comput.*, Dec. 2011, pp. 496–502.

[17] N. Mansouri, "Network and data location aware approach for simultaneous job scheduling and data replication in large-scale data grid environments," *Frontiers Comput. Sci.*, vol. 8, no. 3, pp. 391–408, 2014.

[18] Y. Mansouri, A. N. Toosi, and R. Buyya, "Cost optimization for dynamic replication and migration of data in cloud data centers," *IEEE Trans. Cloud Comput.*, vol. 7, no. 3, pp. 705–718, Sep. 2019.

[19] J. Luo, L. Rao, and X. Liu, "Spatio-temporal load balancing for energy cost optimization in distributed Internet data centers," *IEEE Trans. Cloud Comput.*, vol. 3, no. 3, pp. 387–397, Jul./Sep. 2015.

[20] L. D. D. Babu and P. V. Krishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments," *Appl. Soft Comput.*, vol. 13, no. 5, pp. 2292–2303, May 2013.

[21] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Inf. Sci.*, vol. 192, pp. 120–142, Jun. 2012.

[22] J. Luo, Q. Liu, Y. Yang, X. Li, M.-R. Chen, and W. Cao, "An artificial bee colony algorithm for multi-objective optimisation," *Appl. Soft Comput.*, vol. 50, pp. 235–251, Jan. 2017.

[23] Y. C. He, H. Xie, T.-L. Wong, and X. Wang, "A novel binary artificial bee colony algorithm for the set-union knapsack problem," *Future Generat. Comput. Syst.*, vol. 78, pp. 77–86, Jan. 2018.

[24] P. Mansouri, B. Asady, and N. Gupta, "Solve shortest paths problem by using artificial bee colony algorithm," in *Proc. 3rd Int. Conf. Soft Comput. Problem Solving*. New Delhi, India: Springer, 2014, pp. 183–191.

[25] Q. Xu, R. V. Arumugam, K. L. Yong, Y. Wen, Y.-S. Ong, and W. Xi, "Adaptive and scalable load balancing for metadata server cluster in cloud-scale file systems," *Frontiers Comput. Sci.*, vol. 9, no. 6, pp. 904–918, 2015.

[26] S.-Q. Long, Y.-L. Zhao, and W. Chen, "MORM: A multi-objective optimized replication management strategy for cloud storage cluster," *J. Syst. Archit.*, vol. 60, no. 2, pp. 234–244, 2014.

[27] Q. Wei, B. Veeravalli, B. Gong, L. Zeng, and D. Feng, "CDRM: A cost-effective dynamic replication management scheme for cloud storage cluster," in *Proc. IEEE Int. Conf. Cluster Comput.*, Sep. 2010, pp. 188–196.

[28] E. B. Edwin, P. Umamaheswari, and M. R. Thanka, "An efficient and improved multi-objective optimized replication management with dynamic and cost aware strategies in cloud computing data center," *Cluster Comput.*, vol. 22, pp. 11119–11128, Sep. 2019.

[29] J. Wu, B. Zhang, L. Yang, P. Wang, and C. Zhang, "A replicas placement approach of component services for service-based Cloud application," *Cluster Comput.*, vol. 19, no. 2, pp. 709–721, 2016.

[30] C. Afzal, M. A. Khan, and M. Sudhakar, "A novel reliability management technique by reduced replication in cloud storage applications," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 5, no. 6, pp. 128–131, 2017.

[31] I. Casas, J. Taheri, R. Ranjan, L. Wang, and A. Y. Zomaya, "A balanced scheduler with data reuse and replication for scientific workflows in cloud computing systems," *Future Gener. Comput. Syst.*, vol. 74, pp. 168–178, Sep. 2017.

[32] A. Spivak, A. Razumovskiy, D. Nasonov, A. Boukhanovsky, and A. Redice, "Storage tier-aware replicative data reorganization with prioritization for efficient workload processing," *Future Gener. Comput. Syst.*, vol. 79, pp. 618–629, Feb. 2018.

[33] M. Gribaudo, M. Iacono, and D. Manini, "Improving reliability and performances in large scale distributed applications with erasure codes and replication," *Future Gener. Comput. Syst.*, vol. 56, pp. 773–782, Mar. 2016.

[34] L. Zhang, Y. Deng, W. Zhu, J. Zhou, and F. Wang, "Skewly replicating hot data to construct a power-efficient storage cluster," *J. Netw. Comput. Appl.*, vol. 50, pp. 168–179, Apr. 2015.

[35] B. Spinnewyn, R. Mennes, J. F. Botero, and S. Latré, "Resilient application placement for geo-distributed cloud networks," *J. Netw. Comput. Appl.*, vol. 85, pp. 14–31, May 2017.

[36] N. Mansouri and M. M. Javidi, "A new prefetching-aware data replication to decrease access latency in cloud environment," *J. Syst. Softw.*, vol. 144, pp. 197–215, Oct. 2018.

[37] N. Saadat and A. M. Rahmani, "PDDRA: A new pre-fetching based dynamic data replication algorithm in data grids," *Future Gener. Comput. Syst.*, vol. 28, no. 4, pp. 666–681, 2012.

[38] P. Elango and K. Kuppusamy, "Fuzzy FP-tree based data replication management system in cloud," *Int. J. Eng. Trends Technol.*, vol. 36, no. 9, pp. 481–489, 2016.

[39] N. J. Navimipour and B. A. Milani, "Replica selection in the Cloud environments using an ant colony algorithm," in *Proc. 3rd Int. Conf. Digit. Inf. Process., Data Mining, Wireless Commun. (DIPDMWC)*, Jul. 2016, pp. 105–110.

[40] T. Junfeng and L. Weiping, "Pheromone-based genetic algorithm adaptive selection algorithm in cloud storage," *Int. J. Grid Distrib. Comput.*, vol. 9, no. 6, pp. 269–278, 2016.

[41] L. Wang, J. Luo, J. Shen, and F. Dong, "Cost and time aware ant colony algorithm for data replica in alpha magnetic spectrometer experiment," in *Proc. IEEE Int. Congr. Big Data*, Jun./Jul. 2013, pp. 247–254.

[42] L. Cui, J. Zhang, L. Yue, Y. Shi, H. Li, and D. Yuan, "A genetic algorithm based data replica placement strategy for scientific applications in clouds," *IEEE Trans. Serv. Comput.*, vol. 11, no. 4, pp. 727–739, Jul./Aug. 2015.

[43] M. Bsoul, A. Al-Khasawneh, E. E. Abdallah, and Y. Kilani, "Enhanced fast spread replication strategy for data grid," *J. Netw. Comput. Appl.*, vol. 34, pp. 575–580, Mar. 2011.

RASHED SALEM received the Ph.D. degree in computer science from the University of Lyon 2, France, in 2012. He has been a member of the Complex Data Warehousing and On-Line Analysis Research Group, ERIC laboratory. He is currently an Associate Professor with the Information Systems Department, Faculty of Computers and Information, Menoufia University, Egypt. His current research interests mainly relate to database, business intelligence (BI), data mining, data warehousing, and big data. He has contributed more than 30 technical articles in the areas of distributed databases, data integration and warehousing, software engineering, and big data mining.

MUSTAFA ABDUL SALAM was born in Sharkia, Egypt, in November 1981. He received the B.Sc. degree from the Faculty of Computers and Informatics, Zagazig University, Egypt, the master's degree in information system from the Faculty of Computers and Information, Menoufia University, Egypt, in 2009, specializing in hybrid machine learning and bio inspired optimization algorithms, and the Ph.D. degree in information system from the Faculty of Computers and Information, Cairo University, Egypt, in 2015. He is currently an Assistant Professor with the Scientific Computing and Artificial Intelligence Department, Faculty of Computers and Information, Benha University, Egypt. He has worked on a number of research topics. He has contributed more than 20 technical articles in the areas of neural networks, support vector machines, optimization, time series prediction, extreme learning machine, hybrid CI models in international journals, international conferences, local journals, and local conferences. His majors are machine learning, big data, stream data mining, and deep learning.

HATEM ABDELKADER received the B.Sc. and M.Sc. (by research) degrees in electrical engineering from the Faculty of Engineering, Alexandria University, Egypt, in 1990 and 1995, respectively, and the Ph.D. degree in electrical engineering from the Faculty of Engineering, Alexandria University, Egypt, in 2001, specializing in neural networks and applications. He has been a Professor with the Information Systems Department, Faculty of Computers and Information, Menoufia University, Egypt, since 2004. He has worked on a number of organizations. He has contributed more than 120 technical articles in the areas of neural networks, database applications, information security, and Internet applications.

AHMED AWAD MOHAMED is currently the Ph.D. degree in information system with Menoufia University. He is currently an Assistant Lecturer with the Information Systems Department, Cairo Higher Institute for Languages and Simultaneous Interpretation, and Administrative Science. Also, he was registered a four patent in Egyptian patent office. His major interests are distributed systems, cloud computing, and big data.