Energy and Expenditure Aware Data Replication Strategy

Morgan Séguela, Riad Mokadem, Jean-Marc Pierson
Institut de Recherche en Informatique de Toulouse (IRIT)
Université de Toulouse, France
Email: {morgan.seguela, riad.mokadem, jean-marc.pierson}@irit.fr

Abstract—Energy saving is a major challenge for Information Technology (IT) companies that aim to reduce their carbon footprint while providing large scale cloud services. These companies often rely on data replication technique in order to satisfy tenant’s objectives, e.g., performance, especially with the increasing volume of data distributed throughout the world. In this paper, we propose a static and multi objective data replication strategy (E2ARS) that aims to reduce both energy consumption and expenditure of the provider. E2ARS leverages on cloud heterogeneity and energy efficient technologies. We first compare different policies of our strategy, from only taking energy consumption into account to only taking expenditure into account. Unsurprisingly, the more you want to reduce the energy consumption, the less you replicate. Then, we compare E2ARS with strategies from the literature. E2ARS reduces both energy consumption and expenditure where those strategies satisfy only one of the two objectives.

Index Terms—Cloud, Data replication, Provider expenditure, Energy consumption, SLA violation

I. INTRODUCTION

Lately, data produced by humanity increased sharply and are heavily accessed. One way to cope with this is to replicate data. Data replication is a well studied technique that aims to satisfy different objectives like availability and performance. In the Cloud, this permits to reduce energy consumption and costs as it is a good leverage to adapt resources on demand.

In a Cloud context, elasticity is a way to limit over-provisioning by adapting resources automatically. This implies a specific economic model called the Pay-as-you-go model where tenants only pay for what they consume. The price to rent resources is specified in a Service Level Agreement (SLA) which also contains Service Level Objectives (SLO) the provider has to respect. If the provider does not, penalties are applied, mainly by a refund of the rent to the tenant.

Reducing carbon footprint is an issue that companies have to consider, e.g. we can consider they have values (being carbon neutral) or attract customer that are aware of environmental issues. To fulfill this objective, many techniques have been studied like sleep state and consolidation to put useless resources in sleep mode and to avoid over-provisioning.

In this paper, we propose a static data replication strategy that aims to reduce both expenditure and energy consumption while considering performance for tenants. Our strategy is based on an optimization algorithm which uses our introduced energy consumption and expenditure models. It leverages on heterogeneity, sleep states and consolidation. Note that we consider read-only data that will not be updated.

The rest of this paper is organized as follows: we begin in Section II by a state of the art of data replication strategies that takes into account energy consumption or expenditure of the provider. Then, in Section III, we describe our data replication strategy. In Section IV, we validate the proposed strategy through the analysis of experimental results, comparing with proposals from the literature in terms of energy consumption, number of violations and expenditure. Finally, we conclude and draw some lines for future works.

II. STATE OF THE ART

We focus on strategies that take into account the provider expenditures, its energy consumption, or both. In [1], replication is triggered by a SLO violation and applied only if it gives profit for the provider. [2] proposed a resilient and cost-effective data replication strategy by classifying data into 2 groups based on their popularity, placing and compressing data item either on primary or backup servers. [3] has a workflow application model giving which task needs which data. This permits to replicate data when they are needed and if it reduces costs compared to doing nothing. [4] proposed a strategy that estimates the benefit made from a new replica or data migration for each time frame.

On the side of energy consumption, [5] proposed a static data replication strategy that tries to answer availability, service time, load balancing, energy consumption and latency issues through an evolutionary algorithm where each objective is weighted. [6] considers a 3-tier fat tree data center architecture and aims to reduce energy consumption and bandwidth consumption by replicating in the lower level of the hierarchy. [7] classifies data according to their popularity and servers by their power consumption. It replicates hot data (popular) in hot servers (that consumes more) and vice versa.

Finally, few strategies considers both objectives. [8] aims to reduce energy consumption in order to maximize the provider profit. However, this strategy does not model all the costs, e.g. while transferring files. [9] considers a linear combination of energy consumption and expenditure and then use optimization algorithm to reduce this objective function. This strategy cannot let the administrator choose its policy with a clear view. Compared to the literature, our strategy aims to reduce both energy consumption and costs to store, read and replicate data.
III. DATA PLACEMENT STRATEGY

A. Notation

Let $F$ be a set of files stored on a set of nodes $N$. Each file $f_i \in F$, $1 \leq i \leq z$ has a size $s(f_i)$. Each node $n_j \in N$, $1 \leq j \leq m$ has a storage capacity $c_{pj}$. Let $\phi$ be a matrix of size $(z, m)$ that denotes the placement of the files on the nodes: $\phi(f_i, n_j)$ is equal to 1 if $f_i$ is on $n_j$, and 0 otherwise.

$E_o$ denotes the energy consumed to perform the action $a$, $a \in A$. $Pr_b$ represents the price of a node action $b$, $b \in B$. The sets $A$ and $B$ depend on the state of the node. The static state represents the node when it is up but inactive (static). The dynamic states contain actions such as reading a file from a node (read), writing a file to a node (write), transferring data between nodes through the network (network) and storing a file on a node (store). $A$ contains read, write, network and the static states. $B$ contains only store and network actions.

$t$ is the duration the user will rent the nodes, and $nbRds$ is the number of reads the user will do. Table I summarizes the notations.

B. Models and Objective functions

In this part, models are proposed to estimate the provider’s expenditure and energy consumption used in the optimization algorithm. These models are divided into 3 kinds of events: replicating, storing, reading. The expenditure model used in the following estimations is based on [1]. To estimate the energy consumption, we modeled each component that is used in the data management of a node. At first, disks energy consumption is modeled based on [10], also used in [11], to model Solid-State Drive. The memory energy consumption is modeled based on [12], regularly updated by technical reports [13]. Finally, the network energy consumption is considered through the Network Interface Card for the nodes, based on [14], and switches, from [15]. In these models, processors are only considered through their idle energy consumption. However, it is worth to say that these models are only used to build the following strategy and are not meant to precisely describe reality. They can be modified as it will not impact the core of the proposed strategy.

1) Replication: First, the impact of replicating data from their original node to a chosen node is modeled. Let $N_o$ be the set of nodes included in $N$ storing the original files before the placement strategy occurs. $n_{i_o}$ represents the original node where $f_i$ is stored.

If the chosen node is the original node, no replication is activated and the energy consumption and cost for replication are equal to 0. In other cases, to model the energy consumption, transfer and writing energy have to be considered, given in Eq. 1.

\[
EC_{\text{replic}}(F, N, N_o, \phi) = \sum_{i=1}^{z} \sum_{j=1}^{m} \phi(f_i, n_j) \cdot s(f_i)
\]

\[
[EC_{\text{network}}(n_{i_o}, n_j, f_i) + EC_{\text{write}}(n_j, f_i)]
\]  

(1)

Replicating a file implies costs for provider which is mostly represented by the communication cost through the network. The cost is based on the size of the file $s(f_i)$, and the price per megabyte is represented by $Pr_{\text{network}}(n_{i_o}, n_j)$, which depends on the placement: if the nodes are in the same data center it will be cheaper than being in 2 different regions. The expenditure model to replicate files is given in Eq. 2:

\[
Cost_{\text{replic}}(F, N, N_o, \phi) = \sum_{i=1}^{z} \sum_{j=1}^{m} \phi(f_i, n_j) \cdot s(f_i)
\]

\[
Pr_{\text{network}}(n_{i_o}, n_j)
\]  

(2)

2) Storage: After writing all replicas, the impact of storing files for a duration $t$ is modeled. We suppose that: (i) nodes cannot be turned off if they store data in order to retrieve data as quick as possible (ii) Empty nodes can be turned off and cannot be turned off if they store data in order to retrieve data (iii) adding a file to a node as quick as possible (ii) Empty nodes can be turned off and cannot be turned off if they store data in order to retrieve data.

$E_{\text{static}}(n_j, t)$ is the static energy consumption to keep $n_j$ up for a duration $t$. To estimate this energy consumption we sum up the static power consumption of each component multiplied by $t$. The model is described in Eq. 3:

\[
EC_{\text{store}}(F, N, t, \phi) = \sum_{j=1}^{m} E_{\text{static}}(n_j, t)
\]  

\[
[1 - \prod_{i=1}^{z}(1 - \phi(f_i, n_j))]
\]  

(3)

On the other hand, storing files implies a cost for the provider. It depends on the size of the file $s(f_i)$ and on $Pr_{\text{storage}}(n_j)$ (price per second and per megabyte) to store data on $n_j$ for a duration $t$. The model is considered in Eq. 4:

\[
Cost_{\text{store}}(F, N, t, \phi) = \sum_{i=1}^{z} \sum_{j=1}^{m} \phi(f_i, n_j) \cdot s(f_i)
\]

\[
Pr_{\text{storage}}(n_j) \cdot t
\]  

(4)

3) Read: Finally, the energy consumption and costs linked to the number of reads is modeled.

Let $N_r$ be a set of nodes that contains $f_i$ and $N_r^c$ the set of nodes that do not contain $f_i$. These sets are built based on $\phi$. Nodes in $N_r$ are denoted $n_j$ with $1 \leq j \leq m_i$, with $m_i$ being the number of replicas of $f_i$. In $N_r^c$, nodes are denoted

| Notation | Description |
|----------|-------------|
| $F$      | Set of files, $F = \{f_1, \ldots, f_z\}$ |
| $N$      | Set of nodes, $N = \{n_1, \ldots, n_m\}$ |
| $\phi$   | Matrix of files $F$ stored on node $N$ |
| $s(f_i)$ | Size of file $i$ (MB) |
| $c_{pj}$ | Storage capacity of node $j$ |
| $Pr_a$   | Energy consumption of node action (in Joules) |
| $a \in \{\text{read, write, network, static}\}$ |
| $Pr_b$   | Price of a node action (in $/MB$ or $/$MB per second) |
| $b \in \{\text{network, storage}\}$ |
| $t$      | Renting time parameter (in seconds) |
| $nbRds$  | Number of reads parameter |
with $1 \leq j' \leq m_f$. Let $\text{shortest}(n_1, n_2)$ be equal to 1 if node $n_2$ is the closest to node $n_1$ in terms of transfer time and 0 otherwise.

The energy consumption given in Eq. 5 includes the consumption to read a file $f_i$ from the closest node $n_j$ to the node $n_j$ that requested this file ($E_{\text{read}}(n_j, f_i)$), and the energy consumed by the network to transfer this file between those nodes ($E_{\text{network}}(n_j, n_j', f_i)$).

$$E_{\text{read}}(F, N, \phi) = \frac{1}{2} \sum_{i=1}^{n_f} \sum_{j=1}^{m_f} \sum_{f=1}^{m_f} \text{shortest}(n_j, n_j') \times$$

$$[E_{\text{network}}(n_j, n_j', f_i) + E_{\text{read}}(n_j, f_i)]$$ (5)

The reading cost given in Eq. 6 is mainly based on the cost to transfer data to the node $n_j$ that requested the file $f_i$ from the closest node $n_j$ that stores this file. This cost is based on the size of $f_i$ and the price to transfer this file $P_{\text{network}}(n_j, n_j')$ per megabyte.

$$\text{Cost}_{\text{read}}(F, N, \phi) = \frac{1}{2} \sum_{i=1}^{n_f} \sum_{j=1}^{m_f} \sum_{f=1}^{m_f} \text{shortest}(n_j, n_j') \times$$

$$s(f_i) \times P_{\text{network}}(n_j, n_j')$$ (6)

4) Multi-Objective Function: In this part, global models and the objective function used in the optimization algorithm are introduced. As detailed before, $t$ represents the renting time of the user and $\text{nbRds}$ is the number of reads the user will do. These parameters are hypothesis made to balance the number of replicas. In fact, the lower the number of reads and the higher the renting time, the lower the number of replicas to reduce long term energy consumption and cost.

Based on previously defined formulas (1), (3), (5), the global energy consumption model is given in Eq. 7:

$$EC_{G}(F, N, \phi) = EC_{\text{replic}}(F, N, N_0, \phi) +$$

$$EC_{\text{store}}(F, N, t, \phi) + \text{nbRds} \times EC_{\text{read}}(F, N, \phi)$$ (7)

Similarly, the global expenditure model given in Eq. 8 is based on (2), (4), (6):

$$Exp_{G}(F, N, N_0, \phi) = Cost_{\text{replic}}(F, N, N_0, \phi) +$$

$$Cost_{\text{store}}(F, N, t, \phi) + \text{nbRds} \times Cost_{\text{read}}(F, N, \phi)$$ (8)

With $c_p$ being the storage capacity of node $n_j$, the size of all files stored on this node cannot be higher than $c_p$. This constraint is applied to individual nodes, as it will be discussed later it does not occur if the node is considered as a representative of one data center. Following [16], we choose to create a minimum of 2 replicas per file to assure a data availability within a year of more than 99.99%.

Finally, the objective function is given in Eq. 9

$$\min_{\phi} \left( \frac{EC_{G}(F, N, N_0, t, \text{nbRds}, \phi),}{Exp_{G}(F, N, N_0, t, \text{nbRds}, \phi)} + \sum_{i=1}^{n_f} \sum_{j=1}^{m_f} \phi(f_i, n_j) \times s(f_i) \leq c_p \times j \right.$$

$$\sum_{i=1}^{n_f} \sum_{j=1}^{m_f} \phi(f_i, n_j) \geq 3 \times i,$$ (9)

C. Energy and Expenditure Aware Strategy

We propose an Energy and Expenditure Aware Replication Strategy (E2ARS). This static data replication strategy can be considered as an initial data placement for a dynamic data replication strategy.

1) Overview on the static data replication algorithm:

We consider a transnational cloud provider which provides services through different regions with several data centers in each region. All data centers have different prices and parameters and for the sake of simplicity these nodes are homogeneous inside a data center. A multi-objective optimization algorithm is used to reduce both energy consumption and expenditure. However, including all nodes in the search space would make the processing very long. As the objective function considers the reading expenditure and energy consumption for each node between all other nodes, considering a subset of nodes permits to highly reduce the time to compute this part of the objective function. In order to make the algorithm works in a decent amount of time, a two steps decision process is proposed. The first optimization algorithm chooses on which data center replicas are stored. To do so, a node is chosen to represent the data center (referred to as getRepresentatives(N) in Algorithm 1). The second optimization algorithm chooses where the data will be stored inside the chosen data center. The whole process is given in algorithm 1.

Algorithm 1 Static Data Replication Algorithm

Input: $N$: List of nodes
Input: $F$: List of Files
Input: $t$: renting duration
Input: $\text{nbRds}$: number of reads
Input: $\text{listOfDC}$: List of Data Centers
Output: $\text{Final}\_\text{Individual}$: matrix of files and nodes that represents replica placement
1: $\text{Final}\_\text{Individual} = \text{zeros}(F, N)$ // matrix of zeros
2: $\text{DCrep} = \text{getRepresentatives}(\text{N})$
3: $\text{Origines} = \text{getOriginalNodeFile}(F, N)$
4: $\text{DC}_{\text{individuals}} = \text{SPEA2}(\text{F}, \text{DCrep}, \text{Origines}, \text{t}, \text{nbRds})$
5: $\text{DC}_{\text{individual}} = \text{chooseIndividual}(\text{DC}_{\text{individuals}})$
6: for all $\text{DC}$ in $\text{listOfDC}$ do
7: $\text{tempnodes} = \text{getNodesFromDC}(\text{N}, \text{DC})$
8: $\text{tempfiles} = \text{getFileOnDC}(\text{DC}_{\text{individual}}, \text{F})$
9: $\text{DC}\_\text{Individual} = \text{FirstFit}(\text{F}, \text{tempfiles}, \text{tempnodes})$
10: $\text{Final}\_\text{Individual}[\text{tempnodes}] = \text{DC}\_\text{Individual}$
11: end for

2) Data replication between data centers (lines 1-5): The first step is the most important because it has to choose between highly different data centers. It has to replicate and place data where the trade-off between expenditure and energy consumption is the most balanced from the database administrators perspective. We choose to find the Pareto front to let the administrator choose between reducing energy consumption and reducing expenditure, or make a balanced choice. The Improved Strength Pareto Evolutionary Algorithm, also known as
SPEA2 [17] (referred to as SPEA2 in algorithm 1), is efficient to find this Pareto front. SPEA2 returns a group of individuals from the Pareto front that are as far as possible from each other. The administrator can then choose the policy they want to apply (referred to as chooseIndividual in algorithm 1).

Expenditure and energy consumption models are used as objective functions in the fitness calculation. For this first optimization algorithm, we suppose that each data center can store all the data, and consider the constraint of creating at least 2 replicas at this step. [16] highlights the fact that having independently (by reducing correlated node failure) increases data availability. In a performance context, this will reduce a bottleneck induced by an overload of reading and replication requests.

3) Data placement inside chosen data centers (lines 6-10): The second step chooses on which node a replica is stored inside each data center. Before processing this optimization algorithm, it has to gather nodes from the data center (referred to as getNodesFromDC in algorithm 1) and files that will be stored in the data center (getFileOnDC in algorithm 1).

As most of the cost differences are between data centers, placing data anywhere inside the data center does not have an impact in terms of expenditure. Unlike reducing energy consumption, that can be done by leveraging on sleep state. To do so, we consider a proportion $p$ of nodes that will store data, and other nodes will be put in sleep state to reduce energy consumption. Using a proportion $p$ that is too low would introduce a bottleneck in a performance context, as a high amount of requests will be done on these nodes. Conversely, using a proportion $p$ that is too high increases the number of activated nodes, implying an increase in terms of energy consumption. If the size of all data to store is higher than the storage capacity of all nodes in the subset, we add the same proportion $p$ of nodes in the subset. Based on our experiments, we chose to use 6.25% of the data center nodes as it is a good trade off between performance and energy consumption. This proportion depends on the data center infrastructure and has to be reevaluated in different situations.

The second optimization algorithm takes as input, the set of files $F$, the set of files that will be stored in the chosen data center, and the set of nodes in the current data center. This algorithm sorts chosen files by their size, chooses the proportion of nodes to store data, and then starts to place data with a round-robin method inside the subset of nodes. If a node cannot store the file due to a lack of capacity, it passes to the next node.

IV. EXPERIMENTS

A. Experimental Environment

1) Strategies Parameters: First, we highlight the differences between three kinds of policies based on our optimization algorithm. Those policies represent a range on our objective spectrum based on the Pareto front, from only considering energy consumption ($E2ARSEC$) to only considering expenditure ($E2ARSEX$). We also tried a balanced policy between energy consumption and expenditure, which will be considered on the following experiments ($E2ARS$). To choose this policy, we ordered all the proposed individuals by energy consumption, and we chose the middle one. In following experiments, $t$ is set to 30 days, and $nbRds$ to 100.

Then we compare our strategy with other data replication strategies proposed in the literature, and a control strategy used as a baseline. As control replication strategy, we used a strategy that places 3 replicas randomly called 3Rand. From the literature, we compare our strategy with a static replication strategy called MORM [5] that occurs before the experiment starts, with a multi-objective algorithm that considers availability, latency, load balancing, service time and energy consumption. We also compare our strategy with two dynamic strategies: (i) PEPR [1] takes into account the provider profit (with an income of 0.0205$ per cloudlet) and performance. It is triggered by each violation, replicates if it is still profitable. (ii) Boru et al. [6] consider a hierarchical topology with 3 level of databases (central, data center, rack). The replication is triggered by the number of reads reaching a threshold, and the replication occurs on lower level databases (data center, rack) if they consume less energy and bandwidth than the higher level databases (central, data center).

2) Simulation Parameters: In order to compare those strategies, we implemented them on CloudSim [18]. This simulator has been extended by [1] (replication, monetary cost) and by [19] (energy consumption). We added the capacity for nodes to be turned into sleep mode based on [20] in order to reduce the energy consumption. We supposed that storing data blocks the node from sleeping to keep access to the stored data.

In the experiments, 3Rand, E2ARS, MORM and PEPR are considered using an architecture following a peer to peer topology, and Boru et al.’s architecture is based on a three tier fat tree topology.

We consider two kinds of experiments. Small scale with 30 files and 32 nodes per data center (represented by experiment 1 and experiment 2), and large scale with 1024 files and 128 nodes per data center (represented by experiment 3 and experiment 4). Smaller scale experiments are kept as MORM needed too much memory and could not be processed for large scale ones. Then, when the number of nodes and files are higher, we increased the number of cloudlets from 75,000 to 150,000.

For the workload, [21] shows that when there is a post on a social media with a link to Wikipedia, there is an increased interest about the topic, increasing the number of views on the linked Wikipedia page. Then the interest fades away, decreasing the number of access to the topic page on Wikipedia. [22] also highlights the fact that this interest can fade at different speed. Based on this information, the workload models a short interest for a content, implying a sharp increase followed by a slower decrease. To do so, we choose the gamma probabilistic distribution to model the arrival rate of requests (with parameters $\alpha = 4$ and $\beta = 600$). Then we simulate two different workloads, the first one (represented by experiment 2 and experiment 4) where the increase starts at 1h30, and the second one (represented by experiment 1 and
experiment 3) where the increase starts at 4h. Parameters that are not impacted by the experiment scale are summarized in Table II and differences between experiments are summarized in Table III. Also, experiments are processed 25 times and results are shown through means and standard deviations.

To compare these policies and strategies, we used 4 metrics: (i) the energy consumed (in MJ) by the Cloud at the end of the experiment, (ii) the total cost for the provider (in $), (iii) the number of created replicas, (iv) the proportion of violations.

A GitHub repository is available \(^1\) to replicate the work and explore different experiments and parameters.

### B. Results

1) Comparison between different policies: First, we compare different policies from our static data replication strategy, E2ARSEX, E2ARSEC, and E2ARS.

At first, it can be highlighted that the more energy consumption is considered, the less the strategy creates replicas. Based on Table IV, we can see that E2ARSEC is the one having the lowest number of replicas in all experiments, followed by E2ARS and E2ARSEX. These differences have an impact on the number of violations. Table V shows that linked with the number of replicas, E2ARSEX is the one having the lowest number of violations, followed by E2ARS and E2ARSEC in all experiments.

Tables VI and VII show results related to our main objectives, i.e., energy consumption and expenditure. It highlights the fact that differences between E2ARSEC and E2ARSEX in both energy consumption and expenditure. In fact, E2ARSEC has a lower energy consumption than E2ARSEX, but it is more expensive in all experiments. In the middle of those policies, E2ARS consumes a bit more energy and cost a bit less than E2ARSEX.

However, it should be noted that the number of files and nodes have an impact on these results, which impacts their range of possibility. In fact, some results shows that E2ARS can be closer to E2ARSEX or E2ARSEC in terms of performance, energy consumption and expenditure.

2) Comparison between different strategies: Table IV and table V show the number of replica created and the number of violations per experiment and per strategy respectively. They show that 3Rand keeps a low number of violations in small scale experiments that increases in larger scale ones. Then E2ARS replicates a bit less than 3 times per file with an average computation time of 9.6 seconds and MORM creates a huge amount of replicas and takes 1,164 seconds to compute. This number of replicas had an impact on performance as MORM did not create any violations and E2ARS significantly reduce them compared to 3Rand. For the dynamic group, it can be noted that they start with no replica and create replicas according to their strategy. Tables highlight the fact that in experiment 3, the number of replicas created by PEPR is very low. According to the number of violations, the low number of replicas is due to a lack of profit implied by the absence of

\(^1\)https://github.com/MorganSeguela/Cloud_2021_XPS

### Table II

| Parameters | Values |
|------------|--------|
| File size  | [0.2, 3, 10] GB |
| Number of DCs per region | 4 |
| Cloudlet execution cost | 3.8-4.4*10^6$/MI |
| Transfer cost between regions | 0.094$/GB |
| Transfer cost within a DC | 7.8*10^4$/GB |

### Table III

| Experiments | 1 | 2 | 3 | 4 |
|-------------|---|---|---|---|
| Number of files | 30 | 30 | 1024 | 1024 |
| Number of cloudlets | 75k | 75k | 150k | 150k |
| Number of nodes | 32 | 32 | 128 | 128 |
| Times of increase | 4h | 1h30 | 4h | 1h30 |

### Table IV

| Experience | 1 | 2 | 3 | 4 |
|------------|---|---|---|---|
| Boru et al. | 241 (5.9) | 240 (4.3) | 126 (5.7) | 127 (5.87) |
| 3Rand | 60 (0) | 60 (0) | 2,048 (0) | 2,048 (0) |
| MORM | 6.6k (3.2k) | 6.3k (3.1k) | - | - |
| PEPR | 108 (9.7) | 159 (8.1) | 184 (80.1) | 2,075 (32.1) |
| E2ARS | 85 (2.3) | 85 (2.1) | 2,923 (9.6) | 2,927 (15.7) |
| E2ARSEC | 83 (1.7) | 84 (1.9) | 2,915 (13) | 2,914 (12) |
| E2ARSEX | 86 (1.29) | 86 (1.4) | 2,937 (9.3) | 2,933 (8.6) |

### Table V

| Experience | 1 | 2 | 3 | 4 |
|------------|---|---|---|---|
| Boru et al. | 63k (451) | 70k (190) | 144k (160) | 146k (171) |
| 3Rand | 700 (370) | 610 (137) | 55k (7.3k) | 55k (6.9k) |
| MORM | 0 (0) | 1 (0) | - | - |
| PEPR | 226 (34) | 168 (10) | 109k (5.8k) | 2.1k (32) |
| E2ARS | 510 (164) | 483 (79) | 2.8k (1635) | 2.2k (807) |
| E2ARSEC | 505 (118) | 507 (106) | 3k (1184) | 3.7k (1477) |
| E2ARSEX | 313 (74) | 429 (92) | 2.2k (787) | 2k (781) |

### Table VI

| Experience | 1 | 2 | 3 | 4 |
|------------|---|---|---|---|
| Boru et al. | 74 (17) | 72 (56) | 8.9k (331) | 8.8k (129) |
| 3Rand | 227 (6) | 194 (1) | 942 (37) | 785 (5.54) |
| MORM | 419 (14) | 457 (9) | - | - |
| PEPR | 216 (6) | 192 (1) | 896 (119) | 498 (2.81) |
| E2ARS | 220 (6) | 187 (1) | 602 (14) | 503 (4.55) |
| E2ARSEC | 220 (7) | 187 (1) | 595 (17) | 501 (3.8) |
| E2ARSEX | 228 (10) | 188 (1) | 606 (17) | 505 (3.2) |

### Table VII

| Experience | 1 | 2 | 3 | 4 |
|------------|---|---|---|---|
| Boru et al. | 6.3k (377) | 6.4k (285) | 40k (485) | 40k (485) |
| 3Rand | 4.8k (305) | 4.7k (195) | 11k (230) | 11k (262) |
| MORM | 594 (102) | 1.8k (1.1k) | - | - |
| PEPR | 4.7k (350) | 3.7k (230) | 22k (2.1k) | 6.8k (106) |
| E2ARS | 4.3k (268) | 4.1k (311) | 8.8k (215) | 8.8k (244) |
| E2ARSEC | 4.3k (209) | 4.2k (389) | 8.8k (210) | 8.9k (209) |
| E2ARSEX | 3.8k (209) | 3.7k (333) | 8.7k (235) | 8.7k (186) |
replicas and violations at the start forcing the provider to make expensive file transfer. Otherwise, PEPR replicates more than E2ARS in small scale experiments and a bit less in larger scale ones and keeps having a lower number of violations. Finally, Boru et al. created fewer replicas in large scale experiments as the number of request per file was reduced but keeps having the highest number of violations due to its topology that makes bottlenecks where data are stored.

Finally, table VI and table VII show results of each strategy for each experiment in terms of energy consumption and expenditure, respectively. As 3Rand is the control strategy, it will be compared to other strategies to show their advantages and drawbacks. Boru et al. is the one that costs and consumes the most compared to other strategies. MORM, in small scale experiments, is the 2nd that consumes the most (2 times the most compared to other strategies. MORM, in small scale expenditure, respectively. As 3Rand is the control strategy, it for each experiment in terms of energy consumption and expenditure (reduction of 16% and 25% compared to 3Rand) as it created replicas in all data centers, the cost to transfer files is highly reduced. We also note (from experiment 3) that PEPR failed to create replicas impacting its energy consumption and expenditure. E2ARS and PEPR are very close both in terms of energy consumption (reduction of both 24% compared to 3Rand) and expenditure (reduction of 16% and 25% compared to 3Rand respectively) even PEPR is a dynamic data replication strategy.

V. CONCLUSION

In this paper, we proposed a static data replication strategy called E2ARS that aims to reduce both energy consumption and expenditure compared to other strategies. The proposed strategy takes into account the trade-off between reducing energy consumption and reducing expenditure that is considered and decided by the administrator. E2ARS uses a 2 steps decision process in order to reduce the search space. The first step uses SPEA2 to consider the cloud heterogeneity followed by a second step which is a heuristic that places data on few nodes to leverage technologies like PowerSleep.

We compared our strategy alongside a control strategy that places 3 replicas placed randomly (3Rand). We also compared our strategy to other existing strategies proposed in the literature, both static (MORM) and dynamic (Boru et al., PEPR). Results show that E2ARS reduces the number of violations compared to 3Rand with only a few more replicas and fulfilled its objective to reduce both energy consumption and expenditure compared to the others strategies. Despite E2ARS is a static strategy, its expenditure and the number of violations are only slightly higher compared to dynamic strategies.

For future work, we will use the proposed static data replication strategy as an initial placement for a dynamic data replication strategy that will use more ‘intelligent’ techniques. Finally, we are currently repeating these experiments on a real architecture.

REFERENCES

[1] U. Tos, R. Mokadem, A. Hameurlain, and T. Ayav, “Achieving query performance in the cloud via a cost-effective data replication strategy,” Soft Computing, vol. 25, no. 7, pp. 5437–5454, 2021.

[2] J. Liu, H. Shen, and H. S. Narman, “Popularity-aware multi-failure resilient and cost-effective replication for high data durability in cloud storage,” IEEE Transactions on Parallel and Distributed Systems, vol. 30, no. 10, pp. 2355–2369, 2018.

[3] F. Xie, J. Yan, and J. Shen, “A data dependency and access threshold based replication strategy for multi-cloud workflow applications,” in International conference on service-oriented computing. Springer, 2018, pp. 281–293.

[4] Y. Mansouri and R. Buyya, “Dynamic replication and migration of data objects with hot-spot and cold-spot statuses across storage data centers,” Journal of Parallel and Distributed Computing, vol. 126, pp. 121–133, 2019.

[5] S.-Q. Long, Y.-L. Zhao, and W. Chen, “MORM: A Multi-objective Optimized Replication Management strategy for cloud storage cluster,” Journal of Systems Architecture, vol. 60, no. 2, pp. 234–244, Feb. 2014.

[6] D. Boru, D. Klizovitch, F. Granelli, P. Bouvry, and A. Y. Zomaya, “Energy-efficient data replication in cloud computing datacenters,” Cluster Computing, vol. 18, no. 1, pp. 385–402, Mar. 2015.

[7] Y. Lin and H. Shen, “Eafrr: An energy-efficient adaptive file replication system in data-intensive clusters,” IEEE Transactions on Parallel and Distributed Systems, vol. 28, no. 4, pp. 1017–1030, 2016.

[8] M. Alghamdi, B. Tang, and Y. Chen, “Profit-based file replication in data intensive cloud data centers,” in 2017 IEEE International Conference on Communications (ICC), Paris, France: IEEE, May 2017, pp. 1–7.

[9] Y. Ebadi and N. Jafari Navimpour, “An energy-aware method for data replication in the cloud environments using a tabu search and particle swarm optimization algorithm,” Concurrency and Computation: Practice and Experience, vol. 31, no. 1, p. e4757, 2019.

[10] A. Hyllick and R. Sohan, “A methodology for generating disk drive energy models using performance data,” Energy (Joules), vol. 80, p. 100, 2009.

[11] M. Song, “Minimizing Power Consumption in Video Servers by the Combined Use of Solid-State Disks and Multi-Speed Disks,” IEEE Access, vol. 6, pp. 25737–25746, Jun. 2018.

[12] M. Rhu, M. Sullivan, J. Leng, and M. Erez, “A locality-aware memory hierarchy for energy-efficient gpu architectures,” in 2013 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO). IEEE, 2013, pp. 86–98.

[13] Micron, “TN-40-07: Calculating Memory Power for DDR4 SDRAM,” 2017. [Online]. Available: https://www.micron.com/support

[14] R. Basmadijan, H. De Meer, R. Lent, and G. Giuliati, “Cloud computing and its interest in saving energy: the use case of a private cloud,” Journal of Cloud Computing: Advances, Systems and Applications, vol. 1, no. 1, pp. 1–25, 2012.

[15] A. Vishwanath, K. Hinton, R. W. Ayre, and R. S. Tucker, “Modeling energy consumption in high-capacity routers and switches,” IEEE Journal on Selected Areas in Communications, vol. 32, no. 8, pp. 1524–1532, 2014, publisher: IEEE.

[16] W. Li, Y. Yang, and D. Yuan, “A Novel Cost-Effective Dynamic Data Replication Strategy for Reliability in Cloud Data Centres,” in 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, Sydney, Australia: IEEE, Dec. 2011, pp. 496–502.

[17] E. Zitzler, M. Laumanns, and L. Thiele, “SPEA2: Improving the strength Pareto evolutionary algorithm,” TIK-report, vol. 103, 2001.

[18] R. N. Calheiros, R. Ranjan, A. Belegagazov, C. A. F. De Rose, and A. Y. Zomaya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” Software: Practice and Experience, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[19] M. Seguela, R. Mokadem, and J.-M. Pierson, “Comparing energy-aware vs. cost-aware data replication strategy,” in 2019 Tenth International Green and Sustainable Computing Conference (IGSC), Oct. 2019, pp. 1–8.

[20] S. Wang, J. Liu, J.-J. Chen, and X. Liu, “Powersleep: a smart power-saving scheme with sleep for servers under response time constraint,” IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 1, no. 3, pp. 289–298, 2011.

[21] D. Moyer, S. L. Carson, T. K. Dye, R. T. Carson, and D. Goldbaum, “Determining the influence of Reddit posts on Wikipedia pageviews,” in Ninth international AAAI conference on web and social media. AAAI Press Oxford, UK, 2015, pp. 75–82.

[22] K. Lerman and R. Ghosh, “Information contagion: An empirical study of the spread of news on digg and twitter social networks,” in Fourth international AAAI conference on weblogs and social media, 2010.