Data replication strategy with satisfaction of availability, performance and tenant budget requirements

Said Limam, Riad Mokadem, Ghalem Belalem

Data replication strategy with satisfaction of availability, performance and tenant budget requirements

Said Limam[1] · Riad Mokadem[2] · Ghalem Belalem[1]

Abstract
We propose a dynamic replication strategy that satisfies simultaneously availability and performance tenant requirements while taking into account the tenant budget and the provider profit. The proposed strategy is based on a cost model that aims to calculate the minimum number of replicas required to maintain a high data availability. A replica creation is triggered only when this number of replicas is not reached or when the response time objective is not satisfied. Then, the replication must be profitable for the provider when creating a new replica. Furthermore, data replication and query scheduling are coupled in order to place these replicas in a load balancing way while dealing with the tenant budget. The experiment results prove that the proposed strategy can significantly improve availability and performance while the tenant budget is taken into account.

Keywords Cloud systems · Data replication · Replication cost · Economic aspects · CloudSim

1 Introduction

In recent years, the need for distributed storage has been increasing since application services deal with a large volume of data distributed over Internet when serving a large number of tenants. Consequently, cloud data storage systems usually need to provide a reliable Quality of Service (QoS) by satisfying a Service Level Agreement (SLA), a legal contract between a cloud provider and its tenants, i.e., customers [1]. In this context, data replication is a well-known technique that provides availability and performance. It places multiple copies of data in different locations. This also increases the probability that at least one of these copies is reachable in the face of failures, i.e., fault tolerance. It is therefore no surprise that replication is an important component of cloud computing applications, e.g., Facebook, while services are used by a large number of clients that may access data from different locations with low latencies.

There is a number of works in the literature that deals with data replication in cloud systems [2, 3]. These strategies were proposed for improving performances [4, 5], reducing the bandwidth consumption [6], increasing the level of data availability [7, 8] and load balancing [9]. However, these objectives appear to be conflicting. For example, replicating data ensures the availability. However, this is done on the detriment of communications between sites, which overloads the network and then, affects performances. Furthermore, most of these strategies neglect both the replication cost and the provider profit.

We propose a dynamic data replication strategy that simultaneously satisfies availability and performance requirements while the Profit of the provider is taken into account (DRAPP). We focus on the replication of read only data. Hence, the proposed strategy is used for OLAP purposes.We deal with three issues: (i) when and what data to replicate?

✉ Said Limam
said.limam@gmail.com

Riad Mokadem
Riad.Mokadem@irit.fr

Ghalem Belalem
ghalem1dz@gmail.com

[1] Computer Science Department, Faculty of Exact and Applied Sciences, University of Oran 1 Ahmed Ben Bella, Oran, Algeria

[2] Institut de Recherche en Informatique de Toulouse (IRIT), Paul Sabatier University, Toulouse, France

First, we check the satisfaction of tenant service level objectives, e.g., availability and performance, in order to decide to replicate or not. A replica creation is triggered only if a given availability objective is not reached or a tenant query response time is greater than an agreed response time threshold, specified in SLA. This threshold is previously set in the SLA contract during the negociation phase between the provider and the tenant. In practice, it could be calculated by launching several test queries in the test phase. Then, the maximum response time is selected as a threshold, not to exceed. Elsewhere, penalties are applied for the provider.

The proposed strategy deals with four issues: (i) what data are replicated? We based on data popularity to identify the concerned data. Data popularity constitutes an important parameter that most of replication strategies consider by replicating the most requested data. It can be expressed by the number of requests for this data, which is computed by data access rate. (ii) when to replicate these data? (iii) where to place new replicas? Tenant queries are classified according to different regions. We consider the bandwidth network (BN) level locality [10]. Each region, a separate geographic area, is composed of sub-regions. A sub-region corresponds to a datacenter (DC). The BN between regions is low and the NB within a DC is high with an intermediate NB between DCs. A minimum number of replicas NumberRep is estimated for each region in order to maintain a given high data availability. The replica placement deals with some constraints such as limited budget and cost of replication. The cost of replication should be inferior to the tenant budget. In consequence, replicas are distributed according to this cost. Furthermore, we place them on a load balancing way, and (iv) the number of required replicas is adjusted dynamically. In addition, The provider should have a real profit when triggering a new replica creation. For this aim, we based on its expenditure and revenue estimations when executing a query.

In order to evaluate the proposed strategy, we have widely extended an existing cloud simulator CloudSim [11], already partially extended in [12]. The experiment results show that the proposed strategy can significantly improve both availability and performance simultaneously while the profitability of the cloud provider is taken into account. The rest of this paper is organized as follows: Sect. 2 describes the considered cloud topology andifferent phases of replication through the proposed strategy. Sect. 3 describes the proposed strategy. Section 4 analyzes the experiments in order to validate the proposed strategy. Finally, Sect. 5 contains some related work. Finally, Sect. 6 concludes the paper and gives some future work.

2 Cloud topology and replication scheme

2.1 Cloud topology

A hierarchical cloud system topology that supports replication of data is considered. Cloud providers often establish multiple facilities in separate geographical regions for a multitude of reasons, including providing services that span across the globe. Each region may contain several datacenters that are distributed inside a region. Each datacenter hosts a number of virtual machines that provide computational power, bandwidth network and storage capabilities to several tenants. Throughout this paper, we refer to a virtual machine by a node. All nodes in the cloud are interconnected by network links. Nodes in the same datacenter of a particular geographical region are interconnected via a local high bandwidth and relatively cheaper links. In a similar manner, inter datacenter bandwidth is comparatively less abundant and more expensive and so on with inter-region connections. As the network hierarchy goes from intra-region links (between DCs as shown through the duched lines in Fig. 1) to inter-region links, i.e., the continue lines in Fig. 1, the bandwidth abundance decreases and bandwidth cost increases [4, 10].

2.2 Replication scheme

The proposed DRAPP strategy deal with the following issues that must be resolved: (i) what data should be replicated and when to replicate them in order to meet both the tenants' requirements and provider economic benefit. If data replication is performed too early, this does not accelerate access to data and does not reduce the waiting time [8]. (ii) how many suitable new replicas should be created to meet a reasonable system availability requirement. As the number of replicas increases, the cost of creating, maintaining new replicas will significantly increase and making unnecessary expenses. In consequence, the provider has more expenditures and its profit

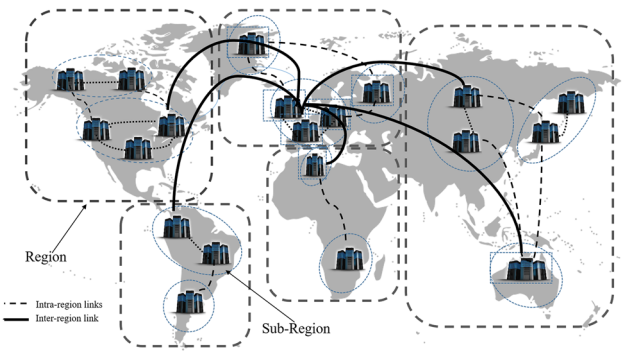

Fig. 1 Cloud topology with regions, sub-regions and datacenters

decreases. Then (iii) determining the number of required replicas should take into account the provider profit, and (iv) where to place new replicas to satisfy bandwidth consumption. In cloud systems, In cloud computing, all resources, e.g., storage, are shared among the tenants. In consequence, an elastic management of these resources permits the satisfaction of both tenant requirements and provider profit. The Quality Of Service (QoS) requirements are extremely important for Cloud Computing. For this aim, a Service Level agreement (SLA), a legal contract between the tenant and the provider, is signed. Mainly, it includes: (i) an amount paid by the tenant to the provider for the processing of his query, (ii) one or more Service Level Objectives (SLO) that are the requirements of the tenant which must be satisfied by the provider, e. g., availability, response time. An example of a response time SLO can be a maximum, response time threshold RespT that can be provided by the provider when executing a tenant query, (iii) a period of validity of the contract, (vi) a monetary penalty paid by the supplier to his tenant in case of violation of the SLA. In our case, the compensation takes the form of an account credit; the service providers reduce the cost of processing of the following requests for the affected tenants.

Our strategy consists of three phases: monitoring, treatment and triggering replication. In consequence, we distinguish three main phases: (i) monitoring phase that deals with the information collecting phase, (ii) treatment phase that deals with the resolving of when and what data to replicate and (ii) triggering replication phase that deals with the new replica placement (where and how many replicas).

2.2.1 Monitoring phase

This phase corresponds to the supervision of the application. It consists in collecting information about existing resources. An example of this information is the data read frequency that is used to calculate the popularity of each data set. Another example is the load of each virtual machine in each datacenter. This parameter is important since new replicas should be created on underloaded nodes.

2.2.2 Treatment phase

The dynamic replication consists in analyzing the information produced by the monitoring phase in order to identify system status. We based on a cost model that aims to estimate the response time of any query before the execution of this query. We consider the response time estimation of each query as in [4]. If the estimated response time is greater than RespT, the replication is considered in order to avoid penalties. The, the provider profit should also be estimated in order to replicate only and only if this replication generates a real economic profit for the provider. This phase also deals with the replica removal. If a replica is unnecessary, it should be removed in order to decrease expenditures of the provider. In consequence, its economic profit is increased.

2.2.3 Triggering replication phase

This phase is based on the results of the treatment phase. When a replication is considered, the next step consists to calculate the number of replicas for each required data in each region. Then, these replicas are placed in a balanced way so that the replication is profitable. The budget of each sub region is taken into account in order to generate a real profit for the provider. In the following Table 1, we describe the role of each element in the three phases of DRAPP.

3 Proposed DRAPP replication strategy

The best way to increase data availability is to replicate data across all sites. However, this solution is not realistic because of storage and bandwidth constraints. Then, a replication strategy is needed. As described above, a replica strategy should deal with the following issues: (i) what data to replicate, (ii) when to replicate, (iii) how many replicas are required to meet the tenants' requirements, (iii) the number of replicas should also take into account the provider' economic benefit and (iv) where to place new replicas. Possible objectives of a replication strategy are to exploit data popularity by replicating the

Table 1 Role for each phase of DRAPP

Functional phase
Monitoring
Collect of informations (data read frequency, budget, etc.)
Treatment
Control the load balancing of each virtual machine
Estimate the response time
Estimate the provider profit
Schedule the access to replicas
Triggering replication
Identify what data files need replication
Determine when to start the replication process
Decide if an unnecessary replica should be deleted
Calculate the minimum number of replicas required
Choose the best location for new replicas

most requested datasets [13], to minimize the update cost [14] that is not the aim of this paper or to maximize economic objectives [15].

This section deals with the four issues presented above. The general idea is to create new replicas according to availability and performance SLO requirements. Then, it should also take into account the provider profit.

Tenant queries are classified according to regions. Available resources are checked in order to verify if they satisfy the requirements of tenant queries. This allows to save the use of resources without creating new replicas unnecessarily. Furthermore, the satisfaction of the provider profit is also taken into account.

3.1 When and what to replicate

An important condition for triggering a replication is the response time objective violation. It occurs when the response time estimation exceeds RespT. We based on the response time estimation given in [4, 16, 17]. Statistical informations are collected, in the monitoring phase, in order to use them when triggering a replication process. An access recorder is assigned to each data, which is used to store the record of tenant access to replicas, including file name, number of access, file size, etc. The data concerned by the replication process are data required by a tenant query that generate the SLA violation dealing with the condition to replicate.

On the other hand, replicas are really created only if the provider profit exceeds replication costs, i.e., the provider's income should exceed the provider's expenditures.

3.1.1 Provider profit estimation

Having a real profit for the provider means that the revenues of the provider are superior than its expenditures.

- Provider's revenues. When executing a tenant query Q, the provider receives a rent from a tenant. This rent depends on different parameters such as the billing period and the objectives to satisfy. In this context, we focus on the availability and performance objectives. The tenant is not billed when the provider triggers a data replication process in order to satisfy the *SLO* requirements. In other terms, data replication is transparent to the tenant. Hence, increasing the number of tenants decreases the per tenant performance. This reduces the overall operating cost of the provider which increases the provider revenue. In consequence, its profit is increased [18].
- Provider's expenditures. The provider has a number of expenditures when executing a query Q as shown in the equation (1). This includes the cost estimation of all resources required to execute Q [15]. Let TT_Q be the estimated total time needed to execute a query Q. Let Nb be the number of nodes required to execute Q during a unit time U_T, e.g. one hour on a node with Amazon. These nodes also include those that hold the replicas created when executing Q [4]. Provider expenditures include the network usage C_N to ship the remote data from other nodes. The network cost also includes the cost of data moving or placement of new replicas on remote nodes which requires additional bandwidth resources [15]. Storage is another cost C_S that the provider should consider when creating new replicas. We also deal with the investment cost C_I that consists to pay the software required for executing tenant queries. The other important cost C_P is the payment of probable penalties to tenants when one or more *SLO* objectives are not satisfied. A penalty is payed from the provider to the tenant when a SLA breach occurs. Finally, provider expenditures include the CPU usage to execute in parallel the sub queries that constitute Q. It is calculated according to TT_Q, U_T, and the number of replicas Nb required for the tenant query execution as shown in Formula (1).

$$Expenditures = C_N + C_S + C_I + C_P + \sum_{1}^{NB} (TT_Q \times U_T) \tag{1}$$

3.2 How many replicas

Knowing the number of replicas, i.e., replica factor, becomes extremely challenging in Cloud, especially with the huge and rapid increase of nodes and resources. To address this issue, the role of the Algorithm 1 is to answer this question: how many copies are needed to be created? Determining the appropriate number of new replicas that should be created in the cloud is important to meet a reasonable system availability requirement. For this aim, it is essential to optimize the replica creation algorithm. With the number of new replicas increasing, the system maintenance cost will significantly increase. However, too many replicas may not increase availability, but bring unnecessary spending instead. In order to enjoy the maximum benefits of replication, an algorithm based on the cost model has been proposed. It calculates the minimum number of replicas that needs to be created for maintaining high data availability while the system performance is improved.

It's better to process the query in DCs close to their regions. For this purpose, we assign queries to the appropriate regions. Then, we estimate the necessary number of replicas for the execution of these queries. To determine

this replica number, we create a new replica as long as the SLA is not satisfied. At the same time, the provider should have a profit. Then, the creation of replicas is stopped if the requirements of the queries are satisfied or if the profit margin is small.

Availability is a critical requirement for many data applications. It can strongly impact the financial cost of a given service. An insufficient replica number may be too costly introducing a heavy performance overhead and reducing data availability. This has a negative impact on the economic benefit of service providers, i.e., it generates significant financial losses because it violates SLA. The proposed algorithm optimizes a profit provider and guarantees a QoS. When a tenant query is submitted, the proposed algorithm considers two options: (i) the first option is to create new replicas to satisfy both availability and performance SLOs which avoids payment of penalties caused by SLA violation. The creation of new replicas is performed only if a minimum profit provider is assured, (ii) the second option is to put the queries that have a low budget in a queue to wait for replicas to be released while considering the payment of penalties if the SLOs of those queries are not satisfied. In the proposed strategy, the second option is chosen so that the provider responds to queries having a high budget. Hence, paying some penalties while receiving high incomes from several tenants do not impact the profit of the provider, i.e., several tenants are serverd by the provider at the same time. Initially, all queries are classified according to the appropriate regions; then we estimate the response time *TR* for each query Q_j. Then we calculate the number of SLA violations. If this number is less than the allowed violation rate, then we launch the execution of these queries without initiating the replica creation process since the current resources can ensure that the QoS delivered can satisfy the tenants' expectations. Before launching queries execution, a scheduling heuristic is used to optimize the execution time of some queries. In the case of SLA violation, the replication process is started. First, we calculate the local budget (Bl_{Ri}) of this region, which equal to the sum of the budgets of the different queries ($BudReq_k$) of tenants. The process of creating new replicas increments the number of replicas ($NumberRep_{R_i}$) as long as the provider's profit is greater than the minimum profit. The creation of new replicas stops if the SLA is guaranteed with the new replicas or if the minimum profit is not assured. In the second situation, the provider may not process certain queries immediately because renting new resources is more expensive than paying penalties to the tenant. The provider informs tenants who have a budget below the minimum cost to postponement the processing of their requests. The tenants decide to either continue waiting, in which case the supplier pays penalties and therefore his profit margin decreases. Otherwise, to look for another provider for the processing of their requests.

Algorithm 1 Determine when to replicate and how many replicas

```
Classify requests by regions
for all region R_i do
  for all Q_j ∈ R_i do
    Estimate the response time TR_j for request Q_j
    if the response time TR_j violates the SLA then
      Nb_sla_Violations + +;
    end if
  end for
  if Nb_sla_Violations < allowable_violation_ratio then
    Scheduling access to replicas;
  else
    Calculate the local budget
    Bl_Ri = Σ_{K∈R_i} BudReq_k ;
    while Profit > Minimum_Profit and SLA not respected do
      NumberRep_Ri + + ;{Increment the number of replicas}
      Check SLA ;{check the quality of services with the new replicas}
    end while
    if SLA is respected then
      Scheduling access to replicas;
    else
      { Here necessarily Profit < Minimum_Profit}
      while SLA is not respected do
        Remove requests that have a low budget;{do not execute them immediately to unload the nodes}
        Add these requests to the waiting list;
        Check SLA;
      end while
    end if
  end if
end for
```

3.3 Replica removal

The proposed strategy adjusts dynamically the number of replicas. To determine which replicas must be removed from the Cloud, we use a counter to calculate the number of visits for each replica. This corresponds to the data popularity during a ceratin period. Then, the replicas with the lowest popularity is considered as a nonperforming replica and will be deleted. Note that the number of replicas maintained after suppression of unnecessary replicas must guarantee the availability agreed upon in the SLA.

On the other hand, we could also remove some replicas when another condition is verified. If the estimated response time of a given query is much smaller than RespT, we remove replicas of data that are required by the corresponding query. We defer this issue to a future work.

3.4 Access scheduling

To get the best performance, queries scheduling and data replication have been coupled. A good scheduling strategy will reduce data access time and minimize the total job execution time in the Cloud. In this phase, the treatment manager allows to schedule the access to replicas. It selects the best replica to the tenant query that pays the most. This feature makes possible to reduce the access time and also to satisfy the maximum expectation and requirement of this query. For scheduling access of a query Q to a replica, we have to sort the various queries in descending order according to their budget. The query which has the maximum budget $Q_{\max}$ is selected and the best replica available Rep_{best} is selected for this query.

Algorithm 2 Access Scheduling

```
Classify requests by regions
for each request Q ∈ the list of request of region R_i do
    Sort queries in descending order according to the budget;
    Select the request Q_max that has the maximum budget;
    Select the best replica Rep_best for this request and execute the request;
    Remove Q from the list of requests;
end for
```

3.5 Replica placement

Replica placement consists in finding physical locations for multiple replicas of the desired data efficiently in a largescale Cloud system. The bandwidth consumption constitutes an important issue when placing replicas. The replica manager decides where new replicas should be placed in order to achieve higher system availability. If the replicas and queries are distributed in an optimized way, the replicas may improve data access speeding, waiting time reduction and also decrease the bandwidth consumption.

In order to place new replicas, Algorithms 3 and 4 trie to optimize the placement of these new replicas. First, we use the Algorithm 3 to distribute the minimum number of replicas, which was calculated with the Algorithm 1 for each region.To satisfy the requirements of tenants of regions that have a large budget, we create more replicas in the sub-regions SR with the most budget. The tenants of these sub-regions will benefit from better access to replicas. Then, they will access to replicas in their sub-regions and this is quite normal because these tenants have paid the most.

Algorithm 3 Distribution of replicas in sub-regions

```
Classify requests by regions
for each region R_i do
    Classify requests according to the different sub-region SR;
    for each sub-region SR_j do
        Calculate the local budget of sub-region Bl_{SR_j}
        Estimate the number of replicates for each sub-region SR_j;
```

$$NumberRep_{SR_j} = \left(NumberRep_{Ri} \times Bl_{SR_j}\right) / Bl_{Ri} \tag{2}$$

```
    end for
    Make an approximation such that :
        NumberRep_{SR_j} is an integer &&
```

$$\sum_{SR_j \in R_i} NumberRep_{SRj} = NumberRep_{Ri} \tag{3}$$

```
    for each sub-region SR_j do
        Place replicas;
    end for
end for
```

For each region R_i, the different queries will be classified according to their SR. Then, we calculate the number of replicas in SR. To estimate the number of replicas in SR_j, we have to calculate the local budget Bl_S of SR_j. Then, we deduce the replica number $NumberRep_{SRj}$ from the total replica number in the region R_i, the total budget Bl_{Ri} in the region R_i and also the budget Bl_S of this sub-region SR_j as shown in Eq. (2). We approximate the number of replicas such as the sum of the replica number in all sub-regions equals the total replica number in the region as mentioned in Eq. (3).

After the distribution of replicas in each sub-region, we use Algorithm 4 to find a suitable placement for the different replicas. Finding optimal placement for replicas across all nodes in the Cloud can be time consuming, which leads to a significant increase in the response time. In our strategy, as the number of replicas to create in a sub-region SR_j is known, this makes it possible to reduce the search area. In order to speed up the search, we choose the first node in the sub-region SR_j that has enough storage space and that guarantees the minimum profit for the provider.

Algorithm 4 Replica placement algorithm

```
i ← 0 ;
while i < NumberRep_{SR_j} do
  for each Host h ∈ SR_j do
    if freeSpace(h) > sizeOf(replica) and Profit > MinProfit then
      place replica on H;
      i++;
      break;
    end if
  end for
end while
```

4 Simulation and result

In order to validate DRAPP, we used CloudSim [11], an open source, generalized and extensible simulation framework that enables seamless modelling, simulation, and experimentation of emerging Cloud computing infrastructures and application services [12]. We have extended CloudSim in order to manage data replication and resource cost measurement. To evaluate the behavior of the DRAPP strategy, we used the following metrics: average response time that corresponds to the total execution time of all cloudlets divided by the number of cloudlets. Replica factor that refers to the number of replicas created during the Query execution. Data availability, SLA violation that represents the number of queries that have a response time greater than the time agreed in the SLA. Bandwidth Consumption and (load balance. In what follows, the experiments aim to validate the proposed strategy though the analysis of:

(i) the impact of the budget on the replication factor and data availability,
(ii) the impact of parallel queries on the replication factor, the response time and the SLA violation, and
(iii) the effect of the number of DCs on the load balance and the system performance.

In the following experiments, we simulate 4 different regions, each region contains 2 sub-regions. Then, a sub-region contains two or three DCs; i.e., a total of 20 DCs. Each DC contains 10 hosts. The maximum number of VMs created in this series of simulation is 100. In these experiments, we compare the result of DRAPP to those of the CDRM strategy [9] and PEPR strategy [19].

4.1 Impact of the number of queries

4.1.1 Effects on the replica factor

In first experiments, we measure the impact of the parallel queries on the replica factor. The impact of the budget is also evaluated.

We measure the number of replicas when varying the number of parallel queries. If this number is small, it will increase the query execution time (data transfer). In addition, strategic replica placement can improve performance. For each experiment, we deal with two budgets: 1$ and 1,5$. We vary the number of queries, i.e., Cloudlet number, from 10 to 100 parallel queries that need different data. We note that with both budgets the number of replicas increases with increasing number of Cloudlets. This indicates that DRAPP dynamically adds more replicas if the current replica number does not satisfy availability requirement. The replica factor adapts with the number of queries. It also adapts with the budget value. With a budget of 1,5$ per cloudlet, the replica number stabilizes more quickly. After reaching a certain point, the number of replicas is maintained at a constant level during experiments. This means that the current replica factor is sufficient to satisfy SLO requirements (Fig. 2).

4.1.2 Effects on the availability

In this following experiment, we try to see the budget effect on the availability of replicas. We have set the number of Cloudlets at 60 and we varied the total budget from 10 $ to 80 $. We calculated the availability of replicas for each budget value. Based on the simulation results presented in Fig. 3, we note that availability increases as the budget increases and this is due to the increase in the number of replicas. For example, when the total budget is 40, the availability is 95%.

4.1.3 Effects on the response time

To study the impact of the Cloudlet number on the average response time, we varied the query number from 10 to 100 and we measured the processing time of queries as shown in Fig. 4. The number of DCs has been set at 20. The response time includes the specific time of the execution

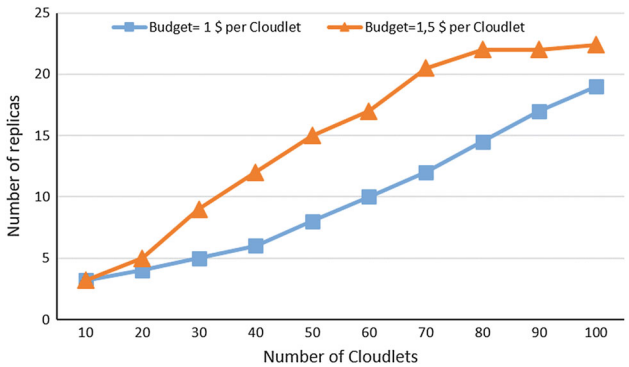

Fig. 2 Impact of the value of the budgets on the replica number

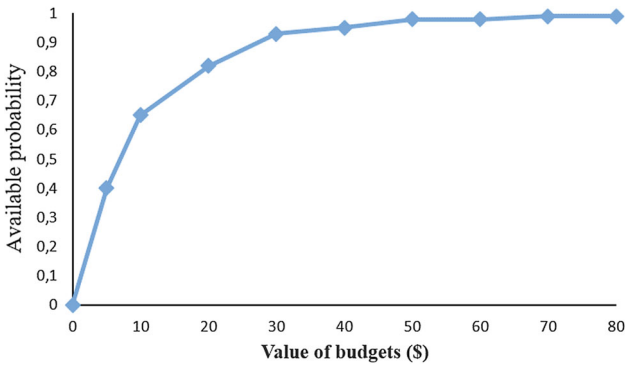

Fig. 3 Impact of the value of the budgets on the availability

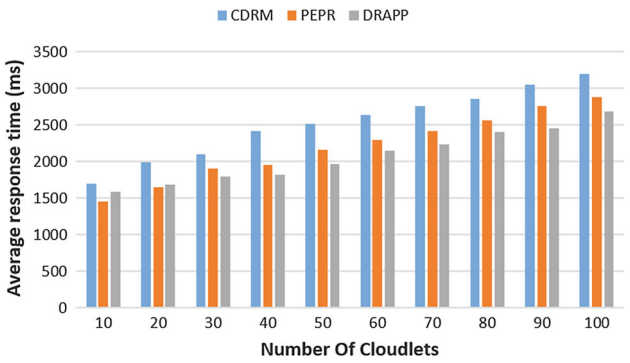

Fig. 4 Impact of the number of cloudlets on the response time

plus the communication time in the network (data access time) and the waiting time.

We notice a significant decrease in the response time for the experimented queries with the DRAPP compared to CDRM and PEPR strategies. DRAPP adapts to the query number. A new replica is created only if the provider gain is real. A replica is placed in the host that receives most queries in opposition of CDRM that uses the migration, which constitutes an overhead that affects the query response time. On the other hand, PEPR strategy places the replicas in the least loaded nodes and therefore not necessarily in the same sub-regions of the tenants. Thus, replicas not close enough to the tenants increase the transfer time. Therefore, the response time will be high.

4.1.4 Effects on the SLA violation

An important characteristic that differentiates the cloud from traditional business models is the penalty mechanism. When an SLA breach occurs, the provider is obligated to pay an agreed upon monetary sum to the tenant [17].

The following experiment aimed to verify the number of SLA violations. We based the same parameters of the previous experiments. To calculate the number of SLA violations, we check the response time of each query. Then, we compare it to the response time threshold defined in the SLA.

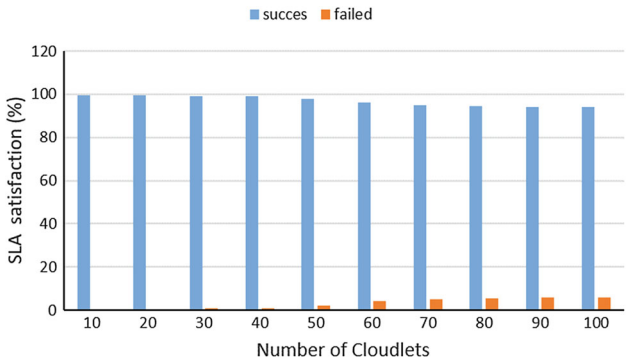

Fig. 5 Impact of the number of cloudlets on the SLA violation

If the response time is greater than RespT, there exist an SLA violation. The results are shown in Fig. 5. We note that the number of SLA violations are very low because DRAPP is dynamic. If the number of replicas at any given time causes an SLA violation, then new replicas are created. Furthermore, when the cost of creating new replicas exceeds the budget, the creation of a replica is canceled. This justifies the existence of some cases of SLA violation.

4.2 Impact of the number of DCs

In the following experiments, we measure the impact of the number of DCs on the system load balance. We deal with five regions. Each region contains two sub-regions. For the first simulation, we created two DCs in each sub-region, i.e., a total of 20 DCs. In other experiments, we deal with three DCs by sub-regions, i.e., a total of 30 DCs. The total number of created VMs in this series of experiments is 60.

4.2.1 Effect on the system load balance

The location of replicas and the scheduling of queries have a significant impact on load balancing. In this experiment, we compare the percentage of load on different DCs. We

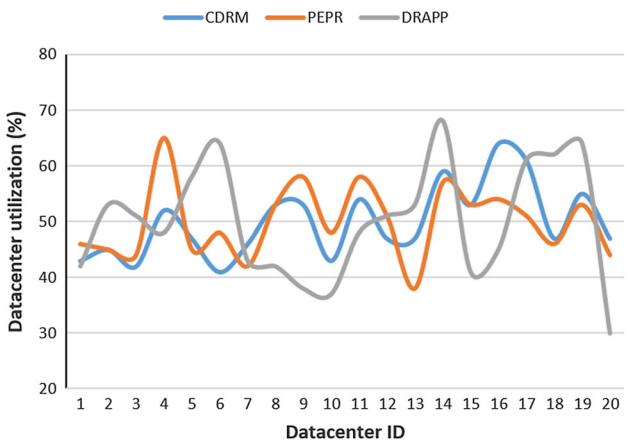

Fig. 6 Impact of the number of DCs on the load balance

use 20 DCs. Then, we calculate the usage percentage of each DC. The test result is plotted in the Fig. 6. We can see that with CDRM, we have a good balance of use between different DCs. For example, the gap between the busiest and least loaded DC is only 23%. This gap reflects good load balancing. This is because the node load criterion is taken into consideration in the replica placement process. Similarly, the gap between the busiest and least loaded DC with PERP is 27%. CDRM evenly distributes imbalanced workload to whole DCs by the use of data migration. Also, it is based on the blocking probability that stop receiving queries in the busiest VM. With DRAPP, we see a larger gap. It exceeds 38%. In fact, queries are sent to DCs closest to their regions which creates an imbalance between the DC loads. satisfying load balancing objective in CDRM is not sufficient to have better response times. The bandwidth consumption in CDRM is more important than our strategy and PEPR that are based on the network level locality.

4.2.2 Effects on the response time

Through this experience, our goal is to see how increasing the number of DCs could affect the average response time of queries. We varied the number of DCs in steps of 5. We fixed the number of queries to 60. The Fig. 7 shows the simulation results of this experience. Initially, we notice a slight advantage of the CDRM strategy compared to DRAPP and PEPR. But when the number of Dcs increases to 10, we remark that DRAPP gives better performance. This advantage becomes more and clearer as the number of DCs increases. We can explain this result by the simultaneous processing of several queries. We estimate the response time of a set of queries at the same time. In PEPR, the estimate is calculated for a single query. In addition, the placement is done for one replica only; this allows introducing a significant additional cost of treatment. In DRAPP, we classify the queries according to their region. Then, we assign these queries to DCs in the same region. This minimizes the response time. Furthermore, we use query scheduling to improve replica access time and maximize vendor profit.

4.2.3 Effects on the bandwidth consumption

In this experiment, we measure the bandwidth consumption for DRAPP, PEPR and CDRM strategies. The use of the network has a significant impact on the provider's expenses and on the data transfer times. On the other hand, the number of replicas and their placements have a big effect on bandwidth consumption. A suitable location can help to reduce network bandwidth consumption and therefore reduce response time. The DC number varies from five to 30 and the number of queries, i.e., number of cloudlet, has been set to 60. Figure 8 illustrates bandwidth consumption with the three compared strategies. These results show that the bandwidth consumption with the CDRM strategy is higher. CDRM uses the migration to balance the load between various DCs, which influences the consumption of the bandwidth. In PEPR, replica placement is done in a way that balances the load across Data Centers (DCs). Some queries will get access to replicas that are in other sub-regions. Therefore, the bandwidth consumption will be increased. The bandwidth consumption with the DRAPP is less important because replicas are closer. Then, less data transfer is required which has the impact of reducing the consumption of bandwidth.

5 Related work

Most of the proposed data replication strategies deal with an individual SLO satisfaction, e.g., availability and performance. Only a few work deal with satisfying simulatenously several SLOs.

Firstly, we can cite strategies that satisfies tenant SLO without takingthe provider profit into account. Sun et al. [6] have proposed Dynamic Data Replication Strategy

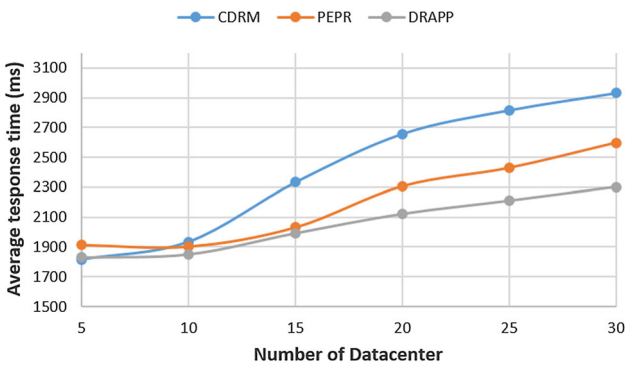

Fig. 7 Impact of the number of DCs on the response time

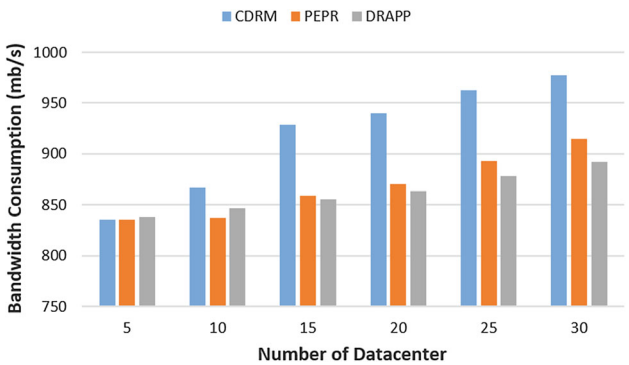

Fig. 8 Impact of the number of DCs the bandwidth consumption

(D2RS). The proposed strategy identifies the most popular data by using historical file access. The creation of new replicas will only be started if a file's popularity is greater than a dynamic threshold; The number of replicates is determined according to the desired data availability.

Wei et al. [9] proposed a strategy called Cost-effective Dynamic Replication Management (CDRM). CDRM calculates and maintains a minimum number of replicas to satisfy a given level of availability. Replica placement is done to balance the load across all sites. Reducing inefficient access ensures that all fragments are served without causing a bottleneck.

Bai et al. [20] have proposed a replica management strategy named Response Time-Based Replica Management (RTRM). The number of created replicas is automatically adjusted and depends on the average response time. The creation of a new replica is done only if the response time exceeds, as in DRAPP, a predefined threshold. This reduces the service response time.

Liu and Shen [21] have proposed a popularity-aware multi-failure resilient and costeffective replication scheme (PMCR). PMCR allows simultaneous management of correlated and independent failures. This management is achieved by replicating the first two replicas of each data segment in the main level, and the third replica in the remote backup level. PMCR uses the similarities and eliminates redundancy between replica chunks. The simulation results show that PMCR guarantees high data availability and durability.

Mansouri et al. [22] have designed algorithms with full and partial future workload information. By exploiting dynamic programming, they formulate offline cost optimization problem in which the optimal cost of storage, Get, Put, and migration is calculated where the exact future workload is assumed to be known a priori. Furthermore, they propose two online algorithms to find near-optimal cost. The proposed algorithms make a trade-off between residential and migration costs and dynamically select storage classes.

Zhang et al. [23] present an auction model to implement a replica placement policy. Proposed work aims to satisfy only availability in a large-scale cloud storage environment. If the desired availability level cannot be maintained, a bidding is held to determine the placement for a new replica. Bidding price is dependent on several properties of the nodes including failure probability, network bandwidth and available space. However, the response time is not included as an objective function.

Boru et al. [24] introduce a data replication strategy that focuses on improving the energy efficiency of cloud datacenters. Their strategy optimizes energy consumption, bandwidth use and network delays at both inter-datacenter and intra-datacenter levels. The authors modeled datacenter power usage and bandwidth consumption of database operations. The simulation study proves that power consumption and response time are improved. Economic benefit is not a focus of this study.

Most of the above strategies do not take into account the provider profit. Only few works consider the cost of data replication and the provider profit as we do in this paper.

Sakr and Liu [25] introduced a SLA customer centric strategy for cloud databases. Servers are scaled in/out according to tenant SLOs. In the proposed strategy, cloud system is closely monitored and cloud providers declaratively define application specific rules to adaptively scale the resources. However, this work neglect the cost of replication and economic benefit of the provider.

Gill and Singh [26] have proposed an algorithm named Dynamic Cost-aware Rereplication and Rebalancing Strategy (DCR2S) with the concept of knapsack problem to optimize the cost of replication. The proposed algorithm determines which file to replicate and to calculate the appropriate number of replicas. Placement of replicas ensures that the cost does not exceed the budget. The simulation results show that DCR2S improve the cost of replication.

Sousa and Machado [27] proposed an elastic multi-tenant database replication strategy. It takes the performance SLA into account while the number of replicas is adjusted by monitoring the system utilization. It also reacts to the workload changes. The results indicate that the proposed strategy satisfies tenant objectives with minimal SLA violations.

The proposed strategy by Tos et al. in [4] ensures performance tenant objective while taking into account the economic benefit of the cloud provider. In replica placement, new replicas are placed on the cloud node that is closest to the most amount of queries. However, the parallelism is not considered when executing queries. Furthermore, the authors compare the proposed strategy to a strategy that is proposed for grid systems. In the PErformance and Profit oriented data Replication Strategy (PEPR) [19] that extends the strategy proposed in [4], the parallelism is taken into account. However, only a minimum number of replicas is kept in order to satisfy a minimum availability objective.

Table 2 summarize different data replication strategies. For each strategy, we specify the main objective defined by the authors, the simulator used to evaluate the proposed strategy as well as the evaluation metrics.

6 Conclusion

We have proposed a data replication strategy that aims to simultaneously satisfy both the availability and performance requirements while taking into account the tenant

Table 2 Comparisons between different data replication strategies in cloud

References	Objective(s)	Economic aspect	Simulator	Evaluation metrics
[4]	Performance	X	CloudSim	Average response time,monetary expenditure, amount of transferred data
[7]	Performance	–	PeerSim	Bandwidth utilization, GET durations
[8]	Availability, bandwidth	–	CloudSim	Successful execution rate, Response time
[9]	Availability, load balancing	–	Hadoop	Average Latency, System utilization, Replica Number, Availability
[26]	Availability	X	CloudSim	Cost of replication,availability of block, availability of File
[20]	Performance	–	OptorSim	Average job time,network utilization
[23]	Availability	X	CloudSim	Response Time,Load Balance, Network bandwidth
[24]	Energy	–	GreenCloud	Energy, Bandwidth, Data access delay
[19]	Performance	X	CloudSim	Average response time,bandwidth consumption, monetary expenditure
[27]	Performance	–	OLTPBenchmark framework	Response Time, SLA violations
[25]	Cunsumer centric performance	–	Berkeley Cloudstone benchmark	SLA Satisfaction
[28]	Availability	–	CloudSim	Response time, Number of replicas

budget and the profit of the provider. In order to preserve resources, the replication is triggered only if the availability is less than to a given desired availability or when the response time is greater than a threshold response time, both defined in SLA. Furthermore, the replication is performed only if the creation of the new replica is profitable to the provider. DRAPP consists of four points: (i) we estimate the minimum number of replications required to satisfy availability requirements, (ii) query scheduling and data replication have been coupled in order to improve the system performance, (iii) we have optimized the placement of a new replica by its distribution in a load balancing way according the budget of each tenant, and (iv) the provider revenues must be superior to its expenditures when replicating data. We validate DRAPP by the achievement of several series of experiments through varying several parameters. The results show the superiority of DRAPP compared to CDRM, an already proposed strategy in clouds. DRAPP significantly reduces the query response time, increases the availability while the tenant budget and provider profit are taken into account.

Some research tracks are possible to continue this work. We can project to: (i) extend experiments and comparisons by studying the influence of replication on the energy consumption and (ii) validate our proposal through experiments on a real cloud platform.

References

1. Kouki, Y., Ledoux, T.: CSLA : a Language for improving cloud SLA management. In: Proceedings of the International Conference on Cloud Computing and Services Science, CLOSER 2012, pp. 586–591. Porto, Portugal (2012)
2. Milani, A., Navimipour, N.J.: Comprehensive review of the data replication techniques in the cloud environments: major trends and future directions. J. Netw. Comput. Appl. **64**, 229–238 (2016)
3. Tabet, K., Mokadem, R., Laouar, M.R., Eom, S.B.: Data replication in cloud systems: a survey. IJISSC **8**(3), 17–33 (2017)
4. Tos, U., Mokadem, R., Hameurlain, A., Ayav, T., Bora, S.: A performance and profit oriented data replication strategy f]or cloud systems. In: Proceedings of the International IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), vol. 2016, pp. 780–787. Toulouse (2016)
5. Xiong, R., Luo, J., Song, A., Liu, B., Dong, F.: QoS preference-aware replica selection strategy using MapReduce-based PGA in data grids. In: 2011 International Conference on Parallel Processing, Taipei City, pp. 394–403 (2011)
6. Kloudas, K., et al.: PIXIDA: optimizing data parallel jobs in wide-area data analytics. PVLDB **9**(2), 72–83 (2015)
7. Silvestre, G., Monnet, S., krishnaswamy, R., Sens, P.: AREN: a popularity aware replication scheme for cloud storage. In: IEEE International Conference on Parallel and Distributed Systems (ICPADS), pp. 189–196, IEEE, Singapore (2012)
8. Da-Wei, S., Gui-Ran, C., Shang, G., Li-Zhong, Jin, Xing-Wei, Wang: Modeling a dynamic data replication strategy to increase system availability in cloud computing environments. J. Comput. Sci. Technol. **27**(2), 256–72 (2012)
9. Wei, Q., Veeravalli, B., Gong, B., Zeng, L., Feng, D.: CDRM: a cost effective dynamic replication management scheme for cloud

storage cluster. In: Proceedings of the IEEE Cluster Computing, pp. IEEE, 188–196. (2010)
10. Park, S.M., Kim, J.H., Ko, Y.B., Yoon, W.S.: Dynamic data grid replication strategy based on internet hierarchy. In: Li, M., Su, X.-H. (eds.) Grid and Cooperative Computing, pp. 838–846. Springer, Berlin (2004)
11. Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A.F., Buyya, R.: Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software **41**(1), 23–50 (2011)
12. Belalem, G., Limam, S.: Towards improving the functioning of cloudsim simulator. Int. Conf. Dig. Inf. Process. Commun. **189**, 258–267 (2011)
13. Lamehamedi, H., Szymanski, B., Shentu, Z., Deelman, E.: Data replication strategies in grid environments. In: Proceedings of the Fifth International Conference on Algorithms and Architectures for Parallel Processing, pp. 378–383 (2002)
14. Bell, W.H., Cameron, D.G., Carvajal-Schiaffino, R., Millar, A.P., Stockinger, K., Zini, F.: Evaluation of an economy-based file replication strategy for a data grid. In: Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGrid 2003, pp. 661–668 (2003)
15. Uras, Tos: Réplication de données dans les systè mes de gestion de données à grande échelle. PhD manuscript. (2017) (In French)
16. Özsu, M.T., Valduriez, P.: Principles of Distributed Database Systems, 2nd edn. Prentice-Hall Inc., Upper Saddle River, NJ (1999)
17. Lang, W., Shankar, S., Patel, J.M., Kalhan, A.: Towards multi-tenant performance SLOs. IEEE Trans. Knowl. Data Eng. **26**(6), 1447–1463 (2014)
18. Kouki, Y., Ledoux, T., Sharrock, R.: Cross-layer SLA selection for cloud services. In: Proceedings of the 1st International Symposium on Network Cloud Computing and Applications. IEEE, pp. 143–147 (2011)
19. Tos, U., Mokadem, R., Hameurlain, A., Ayav, T., Bora, S.: Ensuring performance and provider profit through data replication in cloud systems. Clust. Comput. **21**, 1479–1492 (2017)
20. Xiaohu, Bai, Hai, Jin, Xiaofei, Liao, Xuanhua, Shi, Zhiyuan Shao: RTRM: a response time-based replica management strategy for cloud storage system. In: Proceedings of the grid and pervasive computing. GPC, Springer, pp. 124–33 (2013)
21. Liu, J., Shen, H.: A popularity-aware cost-effective replication scheme for high data durability in cloud storage. In: Proceedings of the IEEE International Conference on Big Data (Big Data), Washington, DC, pp. 384–389, (2016)
22. Mansouri, Y., Nadjaran Toosi, A., Buyya, R.: Cost optimization for dynamic replication and migration of data in cloud data centers. IEEE Trans. Cloud Comput. **99**, 1–14 (2017)
23. Zhang, H., Lin, B., Liu, Z., Guo, W.: Data replication placement strategy based on bidding mode for cloud storage cluster. In: Proceedings of the 11th International Conference on Web Information System and Application, pp. 207–212 (2014)
24. Boru, D., Kliazovich, D., Granelli, F., Bouvry, P., Zomaya, A.Y.: Energy-efficient data replication in cloud computing datacenters. Clust. Comput. **18**(1), 385–402 (2015)
25. Sakr, S., Liu, A.: SLA-based and consumer-centric dynamic provisioning for cloud databases. In: Proceedings of the IEEE 5th International Conference on Cloud Computing, IEEE, pp. 360–367 (2012)
26. Kaur, Gill Navneet, Sarbjeet, Singh: Dynamic cost-aware re-replication and rebalancing strategy in cloud system. In: Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) pp. 39–47, (2014)
27. Sousa, F.R.C., Machado, J.C.: "Towards elastic multi-tenant database replication with quality of service. In: Proceedings of the IEEE/ACM 5th International Conference on Utility and Cloud Computing, UCC 2012, pp. 168–175 (2012)
28. Hussein, M.-K., Mousa, M.-H.: A light-weight data replication for cloud data centers environment. Int. J. Eng. Innov. Technol. **1**(6), 169–175 (2012)

Said Limam is a Ph.D. candidate in the Department of Computer Science Faculty of Exact and Applied Sciences, University of Oran1 Ahmed Ben Bella, Algeria. He received his M.S. Degree in 2012 from the University of Oran. His research interests are: distributed system, grid computing, cloud computing, fault tolerance, data replication and consistency.

Riad Mokadem is an Associate Professor in Computer Science at Paul Sabatier University (IRIT Lab.), Toulouse, France. His main research interests are query optimization in large-scale distributed environments and database performance. Recently, he was invited as a guest editor for a special issue on 'Elastic Data Management in Cloud Systems' in IJCSSE.

Ghalem Belalem Graduated from Department of computer science, Faculty of exact and applied sciences, University of Oran1 Ahmed Ben Bella, Algeria, where he received Ph.D. degree in computer science in 2007. His current research interests are distributed system; grid computing, cloud computing, replication, consistency, fault tolerance, resource management, economic models, energy consumption, Big data, IoT, mobile environment, images processing, Supply chain optimization, Decision support systems, High Performance Computing.