Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/070634

E^2ARS：An Energy-Effective Adaptive Replication Strategy in Cloud Storage System

Xindong You, Li Zhou[*], *Jie Huang, Jinli Zhang, Congfeng Jiang and Jian Wan*

School of Computer Science, Hangzhou Dianzi University, Hangzhou, China

Received: 26 Mar. 2013, Revised: 28 Jul. 2013, Accepted: 30 Jul. 2013
Published online: 1 Nov. 2013

Abstract: In order to solve the urgent issue of the energy consumption in the cloud storage system. An Energy-effective adaptive replication strategy (E^2ARS) is proposed in this paper, in which data partition mechanism, minimal replicas determining model, replicas placement strategies and the adaptive gear-shifting mechanism are elaborately designed. We try to conserve the energy consumption while satisfying the users' desired response time by our E^2ARS scheme. Mathematical analysis show that our E^2ARS scheme will save energy consumption definitely when the system's workload is light or desired response time is loose. And the simulation experiment results demonstrate that through our E^2ARS scheme energy consumption can be saved while with Qos satisfied and data availability guaranteed, when varying the arrival rate, desired response, replicas number, parallelism degrees.

Keywords: Energy consumption, Energy conservation, replicas management, cloud storage system, gear-shifting mechanism.

1 Introduction and Motivation

Cloud Storage is emerging as a powerful paradigm for sharing information across the Internet, which satisfies people's mobile data demand anywhere and anytime. Many corporations and R&D institutions employ the Cloud Storage as the datacenter to store large amount of the data material. However, a recent industry report reveals that storage devices account for almost 27% of the total energy consumed by a data center [1]. This trend will continue undoubtedly in the near future [2], therefore the energy-consumption problem in data centers will become even serious. Study also found that: on the one hand, data centers consume large amount of the energy, on the other hand, the utilization of the servers of disks of the data centers is lowly to average 25%∼30%. Therefore, how to utilize and manage the servers or the disks of the DataCenter efficiently to conserve the energy consumption is an urgent issue concerned both in research and industry domain. Traditionally, data replication has been widely used in the large distributed system as a mean of increasing the data availability and to balance the workload etc. Recently, more and more researchers employ replication technique to skew workload in order to save energy consumption while the utilization or the workload of Storage System is low. Reasonable replicating strategies make it possible to power off some data nodes to conserve energy consumption while guaranteeing data availability and satisfying the QoS of user. However, Powering down some nodes make the problem of balancing the workload on the remaining active nodes is curial. Therefore, how the placing the replicas in nodes is be off importance. Accordingly, in this paper, we attempt to solve the overall problem of saving energy consumption utilizing the replication strategies by setting the following basic problems:

(1) How to determine the minimal number of the data blocks to achieve essential requirement of storage system: data availability.

(2) How to determine the parallelism degree by partitioning the data to shorten the response time.

(3) How to placing the primary or replicas in the data nodes make it possible to power off some nodes to save energy while guaranteeing data availability as the workload is light.

(4) On the above grounds, try to solve the problem of how to determine the number of nodes to be turned off according to the workload, and how to make the load on the remaining nodes balanced, in order to meet the desired response time.

On the whole, the mismatching between on the energy consumption and the utilization of the large scale storage

[*] Corresponding author e-mail: zhouli@hdu.edu.cn

system is the motivation of our research. An energy-effective adaptive replication strategy (called E^{2}ARS) is present in this paper. We try to employ replication strategies to achieve energy proportional, which means the energy consumption is approximately proportional to the utilization of the storage system while satisfying the user's requirement.

2 Related Work

A great deal of work has done on energy conservation for large-scale storage systems based on data management and DVS (Dynamic Voltage Scaling), which is based on caching[1,3], data placement [4], data migration [5,6], and data replication [5,7,8,9]. These techniques try to prolong disk idle times, so as to make it possible to place idle disks in low-power state, thus saving energy. T. Xie proposes a novel energy-aware strategy, called striping-based energy-aware (SEA), which can be integrated into data placement in RAID-structured storage systems to noticeably save energy while providing quick responses [11], in which they implement two SEA-powered stripping-based data placement algorithms, SEA0 and SEA5, by incorporating the SEA strategy into RAID-0 and RAID-5, respectively. Extensive experimental results demonstrate that compared with non-stripping data placement algorithms, SEA algorithms significantly improve performance and save energy. Rini T. Kaushik propose the Green HDFS using data-classification-driven data placement allows scale-down by guaranteeing substantially long periods of idleness in a subset of servers in the datacenter designated as the Cold Zones [12,13]. There are many other literatures employ data management strategies to conserve energy consumption and achieve the expected performance [14,15,16]. However, among these techniques, exploiting data replication is an attractive practical option, since it is widely employed in server clusters for diverse purpose, including data availability, durability, load balancing, etc. Although replication requires additional storage capacity, it usually comes at a very low cost, since it is well known that storage resources in data centers are often considerably under-utilized at around 1/3 of total available capacity [10,7,8]. C. Weddle etc. [9] built the power-aware RAID (PARAID) based on the elaborated data placement and replication strategies, which reduces energy user of commodity server class disks without specialized hardware. PARAID uses a skewed stripping pattern to adapt to the system load by varying the number of the powered disks. By spinning disks down during light loads, PARAID can reduce power consumption, while still meeting performance demands. Furthermore, J.Kim and D. Rotem based on the PARAID, using replication for energy conservation in RAID Systems [16,17], in which they present a novel approach, called FREP. FREP includes a replication strategy and basic function to enable flexible energy management. The main difference between PARAID and FREP is that PARAID shifts gear within a RAID unit by spinning up/down one or more disks in the array, while FREP do it across multiple RAID arrays. Another difference is the conditions leading to a gear-shift, PARAID relies on disk utilization to make its gear-shift decisions, while FREP monitors the degree of SLA satisfaction for reconfiguration. In their experiments, FREP dramatically reduces energy consumption with a minimal response time penalty. Near recently, M.Nijim etc. develop an adaptive energy-saving scheme (DCAPS) in parallel disk system [18], which consists of a data partitioning mechanism, a response time estimator, and an adaptive energy-conserving mechanism. Experimental results consistently show that DCAPS significantly reduces energy consumption of parallel disk systems in a dynamic environment over the same disk systems without using DCAPS.

Above related work show that employing replication technique to conserve energy consumption is feasible and effective. *However, most of the replication strategies are used in the RAID or the parallel disk system, and they can't solve the all of the problems mentioned in our first section. Furthermore, how to employ and design the replication strategy to save energy according to the feature of the cloud system is an urgent problem to be settled. In this paper, the present E^{2}ARS try to conserve the energy consumption through the following consisted parts: data partition mechanism ,the minimal replicas determining model, replicas placement strategy, adaptive gear-shifting mechanism, in which some of the parts are constructed on the basics of the properties of the cloud storage system.*

3 E^{2}ARS: Energy-Effective Adaptive Replication Strategy

We describe the Energy-Effective Adaptive Replication Strategy (E^{2}ARS) in this section. Firstly, we construct the framework of the E^{2}ARS. And then the detailed implement procedures will be described in the following subsections.

3.1 System Architecture

Figure 1 outlines the system architecture of E^{2}ARS. Clients send the disk requests to cloud storage system with a tetrad attributes, according which the present Energy-Effective Adaptive Replication Strategy (E^{2}ARS) determine the minimal replicas number of the file in the request, partitioning the file, and place the replicas in the data nods according to a certain strategy, finally combine with status of the cloud storage system to gear shifting on the data nodes. The constituent parts of E^{2}ARS will be discussed in the following subsections.

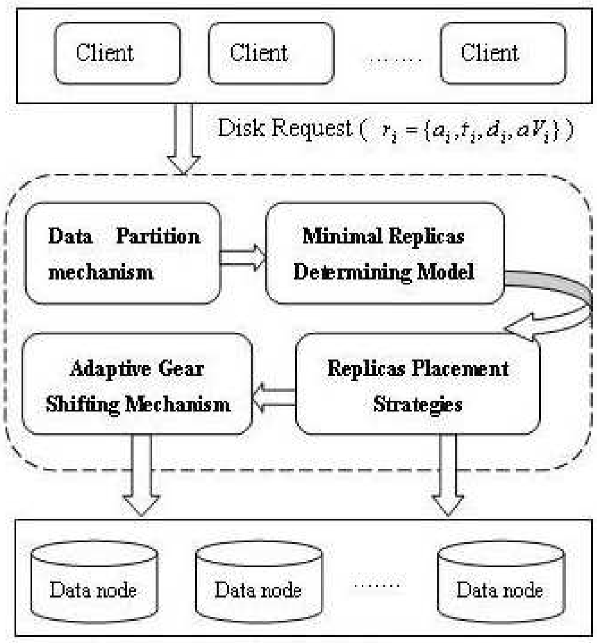
Client
Client
Client
Disk Request ($r_i=\{a_i,t_i,d_i,aV_i\}$)
Data Partition mechanism
Minimal Replicas Determing Model
Adaptive Gear Shifting Mechanism
Replicas Placement Strategies
Data node
Data node
Data node

Fig. 1: Framework of the E^{2}ARS.

3.2 *The Algorithm of E^{2}ARS*

We will describe the E^{2}ARS as the following procedures: Firstly, modeling the system state, disk request and the data nodes characteristic of the cloud storage system. Secondly, according to disk request characteristic, we design the data partition mechanism to optimize the parallelism degree, and then build a mathematical formula to calculate the minimal replicas number of the data blocks. Finally, replicas placing strategy is employed to facilitate gear-shifting mechanism implement on the cloud storage system.

(1) System Status and Energy Consumption Modeling

In this paper, requests to all of the files are modeled as a Poisson process with the overall arrival rate λ.

We consider a sequence of application requests $R=\{r_1,r_2,\ldots\ldots,r_n\}$ will be submitted to the Cloud Storage System. Each request ($r_i\in R$) is a tetrad. That is $r_i=\{a_i,t_i,d_i,aV_i\}$, where a_i,t_i,d_i,aV_i is the arrival time, the desired response time, the data size and the availability of the request r_i, respectively. Furthermore, nodes set in the Cloud Storage System is expressed as: $N=\{N_1,N_2,\ldots,N_m\}$, in which the number of the nodes is m. Each node ($N_j\in N,\ j\in(1\sim m)$) is a six-tuple, that is $N_j=\{\lambda_j,\tau_j,f_j,bw_j,e_j,sm_j\}$, where $\lambda_j,\tau_j,f_j,bw_j,sm_j$ is the arrival rate of request, and the average service time, the failure probability, the network bandwidth, the energy consumption rate, maximal sessions can be connected of the N_j, respectively.

According to the above system modeling, we model the energy consumption of the Cloud Storage System in this subsection. Assume the request r_i to be processed in the jth node, and the energy consumption of request is: $E_{ij}=e_j\cdot\theta_i$, where θ_i is the processing time of the request r_i . Therefore, energy consumption of Node j can be written as below:

$$E_j=\sum_{E_{ij}=}\sum_{r_i\in N_j}e_j.\theta_i \tag{1}$$

The whole system energy consumption can be expressed as:

$$E=\sum_{j=1}^{m}E_j=\sum_{j=1}^{m}\sum_{r_i\in N_j}E_{ij}=\sum_{j=1}^{m}\sum_{r_i\in N_j}e_j.\theta_i \tag{2}$$

Now we can obtain the following non-linear optimum problem formulation to compute the energy consumption of the parallel disk system. Minimize $E=\sum\limits_{j=1}^{m}\sum\limits_{r_i\in N_j}e_j.\theta_i$, subject to $re_i\le t_i$, where re_i is the response time of the ith application request.

(2) Data Partitioning Mechanism Different from the traditional storage system, the Cloud Storage System often strip the application data into blocks for parallel processing. However, too many blocks lead to much communication overhead, too few blocks resulting in poor performance of the application. Therefore, how to partitioning the application data is the first important issue for improve system performance. Our data partitioning mechanism designed to optimize the parallel degree of the application request $r_i=\{a_i,t_i,d_i,aV_i\}$ in order to minimize the response time of the application. We denote the parallelism degree of the application request r_i with p_i , and the p_i can be calculated as the following procedures. Lets first formally derive the disk service time $T_{disk}(d_i,p_i)$ of request r_i . Thus, the disk service time can be computed as:

$$T_{disk}(d_i,p_i)=T_{seek}(p_i)+T_{rot}(p_i)+T_{trans}(d_i,p_i) \tag{3}$$

Where $T_{seek}(p_i)$, $T_{rot}(p_i)$, $T_{trans}(d_i,p_i)$ are the seek time, rotation time, and transfer time of the disk request respectively. $T_{seek}(p_i)$ can be approximately as Eq (4) [22,23]

$$T_{seek}(p_i)=eC(1-a-b\ln(p_i))+f \tag{4}$$

Where C is the number of cylinders on a disk, a and b are two disk-independent constants, whereas e and f are disk-dependent constants. The value of rotation time can be expressed as :$T_{rot}(p_i)=\frac{p_i}{p_i+1}T_{Rot}$, where T_{Rot} is the normal rotation time of a disk. The disk request transfer time can also be given as:$T_{trans}(d_i,p_i)=\frac{d_i}{p_i}.\frac{1}{B_{disk}}$. We can obtain the value of disk service time as:

$$\begin{aligned}T_{disk}(d_i,p_i)&=eC(1-a-b\ln(p_i))\\&+f+\frac{p_i}{p_i+1}T_{Rot}+\frac{d_i}{p_i}.\frac{1}{B_{disk}}\end{aligned} \tag{5}$$

Now we are positioned to calculate the optimal parallelism degree of request $T_{disk}(d_i, p_i)$ by determining the minimum of the function . Thus we can obtain the optimal value of (parallelism degree of) by solving the Eq. (6):

$$\frac{d(T_{disk}(d_i,p_i))}{d(p_i)} = \frac{T_{ROT}}{p_i+1} - \frac{p_i.T_{ROT}}{(p_i+1)^2} - \frac{ecb}{p_i} - \frac{d_i}{p_i{}^2}\cdot\frac{1}{B_{disk}} = 0 \quad (6)$$

(3) Minimal Replicas Determining Model

When we obtained the optimal value p_i, combine with the parameter aV_i: availability of request $r_i = \{a_i, t_i, d_i, aV_i\}$ and the node failure probability parameter f_j, of the $N_j = \{\lambda_j, \tau_j, f_j, bw_j, e_j, sm_j\}$, the minimal replicas number of the file request by r_i can be deduced as[24]: The availability of file F is expressed as (R_i is the number of the replicas):

$$P(FA) = 1 - \overline{P(FA)} = 1 - \sum_{k=1}^{p_i} (-1)^{k+1} c_{p_i}^k (\prod_{c=1}^{R_i} f_j)^k \quad (7)$$

Our E^{2}ARS should be designed to guarantees the availability of the request. That is:

$$\begin{aligned} P(FA) &= 1 - \overline{P(FA)} \\ &= 1 - \sum_{k=1}^{p_i} (-1)^{k+1} c_{p_i}^k (\prod_{c=1}^{R_i} f_j)^k \geq aV_i \end{aligned} \quad (8)$$

$$\begin{aligned} \Rightarrow P(FA) &= 1 - \overline{P(FA)} \\ &= 1 - \sum_{k=1}^{p_i} (-1)^{k+1} c_{p_i}^k (\prod_{c=1}^{R_{\min}} f_j)^k \\ &= aV_i \end{aligned} \quad (9)$$

Where $R_{\min}$ is the minimal replicas number of the file request by r_i. By solving the Eq.(9), we can get the $R_{\min}$. In order to assure to access the file, we make additional restrictions: $if(R_{\min} < 2), R_{\min} = 2$.

(4) Energy-Conserving Replicas Placement Strategies Replicas in Cloud Storage System, not only can guarantee files availability but also can conserve energy consumption when placing appropriately, which means when the workload of Cloud Storage System is light, some of nodes be powered down to save energy consumption, the left replicas can also guarantee files availability and balanced nodes' workload approximately. To obtain the objectives, our Energy-Conserving Replicas Placement Strategies is given as the following procedures:

(a)Primary block (Pb_j) placement From the above mentioned, file F_i requested by request r_i can be partitioned into p_i blocks. The primary blocks (Pb_j) are placed among p_i nodes through the data broker in Cloud Storage. We organize the nodes as a ring, in which node is identified with $NL_j(j \in (0, p_i))$. And the primary blocks are placed according to the following mapping: the primary blocks are placed to the nodes in the ascending order. That is expressed as Eq.(10):

$$Pb_j \rightarrow NL_j(j \in [1, p_i]) \quad (10)$$

(b) First replica (rb_j^1) placement Accordingly, the first replica (rb_j^1) of blocks of the file F_i is placed before the primary blocks, That is given as Eq.(11):

$$rb_j^1 \rightarrow NL_{j-1}\ (j \in [2, p_i]) \quad while \quad rb_0^1 \rightarrow NL_{p_i-1} \quad (11)$$

(c) Second replica (rb_j^2) placement Naturally, the second replica (rb_j^1) of blocks of the file F_i is placed before the first replica, That is written as Eq.(12),Eq.(13):

$$rb_j^1 \rightarrow NL_{j-1}\ (j \in [2, p_i]) \quad (12)$$

$$if(j-2<0), rb_j^2 \rightarrow NL_{j-2+p_i} \quad (13)$$

(d) General replicas placement strategies Without loss of generality, the replica number of F_i is R_i, the replicas placement strategy is expressed in the following Eq.s, where .

$$rb_j^k \rightarrow NL_{j-k} \quad (j-k \geq 0) \quad (14)$$

$$rb_j^k \rightarrow NL_{j-k+p_i} \quad (j-k < 0) \quad (15)$$

(5) Response Time Estimator Based Gear Shifting Mechanism The above replicas placement strategies aim to conserve the energy consumption while guaranteeing the QoS of the user. In this subsection, we describe how to gear shift among the cloud storage nodes according to the response time estimator. Firstly, the time spent by application request r can be deduced by formula (16), in which T_{queue} is the queuing delay at the client side. $T_{partition}$ is the time spent in data partitioning, T_{proc}^i is the system processing delay experienced by the ith stripe unit of the request.

$$T(r,p) = T_{queue} + T_{partition} + \max_{i=1}^{p}\{T_{proc}^i(r,p)\} \quad (16)$$

Eq.16 is designed to be our time response time estimator, and we will describe how to deduce the related parameters in the next section. Secondly, the above replicas placement strategies allow powering down some of the Cloud Storage nodes to save energy consumption while guaranteeing the availability of the files request by r_i [12]. Furthermore, assume the number of file F_i's replicas is R_i that are placed among p_i nodes, the availability of the file can be guaranteed and the load balanced among the nodes can be achieved approximately , only need to meet: R_i +1 nodes are not powered down continuously[12]. That is, the maximal powered off nodes ($\max\_off$) is:

$$\max\_off = \left\lfloor {}^{p_i}\!/_{R_i+1} \right\rfloor \cdot R_i + p_i\%(R_i+1) \quad (17)$$

Therefore, our task is finding how many nodes can be powered off (not exceeding $\max\_off$) to save energy consumption while satisfying the time requirement of the request through the response time estimator. According to the workload of the Cloud System and the response time requirement of the request r, our E^{2}ARS will gear-shifting among the related cloud nodes refer to literature [9].

4 Analysis Evaluation

In this section, we describe the response time estimator in detail. Then give the performance analysis accordingly. The simulation experiments will be done to verify the effectiveness of our present E^{2}ARS in the next section.

According to Eq.(16), the time spent by the request is determined mainly by the third term: $\max_{i=1}^{p}\{T^i_{proc}(r,p)\}$ (T_{queue} and $T_{partition}$ are relative constant). The system processing delay experienced by the ith stripe unit of the request is T^i_{proc}, which can be expressed as:

$$T^i_{proc}(r,p) = T^i_{network}(r,p) + T^i_{disk}(r,p) \tag{18}$$

$T^i_{network}$ and T^i_{disk} are the delays at the network subsystem and parallel disk subsystem respectively. We assume that when the ith stripe unit of a request arrives at the network queue, there are k stripe units waiting to be delivered to the parallel disk subsystem. $T^i_{network}$ can be written as:

$$T^i_{network}(r,p) = \frac{d/p + \sum_{j=1}^{k} d_j}{B_{network}} \tag{19}$$

Where d_j is the data size of the jth stripe unit in the network queue and $B_{network}$ is the effective network bandwidth, and d is the data size of the request r. Similarly, it is assumed that the ith stripe unit of the request arrives at disk j, there are k disk requests must be processed by disk j before handling the stripe unit. Thus, $T^i_{disk}(r,p)$ is given by the following formula:

$$T^i_{disk}(r,p) = T_{disk,j}(d/p) + \sum_{i=1}^{k} T_{disk,j}(d_i) \tag{20}$$

Where $T_{disk,j}(d)$ is the disk processing time of a request containing d bytes of data. $T_{disk,j}(d)$ can be quantified as follows:

$$T_{disk,j}(d) = T_{seek} + T_{rot} + \frac{d}{B_{disk}} \tag{21}$$

We assume when all of the stripe nodes active. The estimated response time expressed as below:

$$\begin{aligned} T(r,p) &= T_{queue} + T_{partition} + \max_{i=1}^{p}\{T^i_{proc}(r,p)\} \\ &= T_{queue} + T_{partition} \\ &+ \max_{i=1}^{p}\{T^i_{netwrok}(r,p) + T^i_{disk}(r,p)\} \\ &= T_{queue} + T_{partition} \\ &+ \max_{i=1}^{p}\{\frac{d/p + \sum_{j=1}^{k} d_i}{B_{network}} \\ &+ T^i_{disk}(d/p) + \sum_{i=1}^{k}(T_{seek} + T_{rot} + \frac{d}{B_{disk}})\} \end{aligned} \tag{22}$$

From formula (22), we can see T_{queue} and $T_{partition}$ are constant as to a certain request from the client. When employ BC or DC to power off the nodes to conserve energy, the load can be balanced or nearly balanced among the remaining nodes [25]. Therefore, it can be derived that when power off s nodes, the response time can be approximately computed as:

$$\begin{aligned} T(r,p,s) &= T_{queue} + T_{partition} \\ &+ \max_{i=1}^{p}\{T^i_{network}(r,p,s) + \frac{p}{p-s}T^i_{disk}(r,p,s)\} \end{aligned} \tag{23}$$

Where s is the number of nodes powered off when the system served the request r, and s should be meet: $s\max\_off = \left\lfloor p_i/R_i+1 \right\rfloor \cdot R_i + p_i\%(R_i+1)$. Furthermore, according to Eq.(2) , the conserved energy can be calculated as:

$$E = \sum_{j=1}^{m} E_j = \sum_{j=1}^{m}\sum_{r_i\in N_j} E_{ij} = \sum_{j=1}^{m}\sum_{r_i\in N_j} e_j.T(r_i,p_i) \tag{24}$$

Where θ_i is the processing time of the request r_i. Therefore, our strategies aim to save the maximum energy consumption while satisfying the Qos and guaranteeing the availability. Based on the above analysis, all the nodes active, the energy consumption in the fixed period of time is deduced as:

$$\begin{aligned} E &= \sum_{j=1}^{m}\sum_{r_i\in N_j} e_j.(T_{queue} + T_{partition} \\ &+ \max_{l=1}^{p_i}\{T^l_{netwrok}(r_i,p_i) + T^l_{disk}(r_i,p_i)\}) \end{aligned} \tag{25}$$

According to our gear-shifting mechanism, when the workload is light or the Qos level is low, assume s nodes can be powered down to save energy based on the response time estimator. The energy consumption in the fixed period can be deduced in the following formula:

$$\begin{aligned} E(s) &= \sum_{j=1}^{m}\sum_{r_i\in N_j} e_j.(T_{queue} + T_{partition} \\ &+ \max_{l=1}^{p_i}\{T^l_{netwrok}(r_i,p_i,s_i) \\ &+ \frac{p_i}{p_i-s}T^l_{disk}(r_i,p_i,s)\}) \end{aligned} \tag{26}$$

Therefore, the mathematical analysis show that when the average number s nodes are powered down, the energy consumption saved (Es) is deduced as:

$$\begin{aligned} E_s = E - E(s) &= \sum_{j=1}^{m}\sum_{r_i\in N_j} e_j.(T_{queue} + T_{partition} \\ &+ \max_{l=1}^{p_i}\{T^l_{netwrok}(r_i,p_i,s) \end{aligned} \tag{27}$$

According to the Eq.(27), we can see that the E_s is a positive number and increased by the number (s.) of nodes powered down. Then based on our gear-shifting mechanism: the lower is the workload or the Qos level, the bigger is the number s, and the conserved energy consumption is much. Therefore, according to the above mathematical analysis, our E^{2}ARS will save energy definitely in certain condition.

5 Experimental Evaluation

To evaluate the performance of our E^{2}ARS scheme in an efficient way, we building our simulation experiments over GridSim toolkits by augmenting the energy related parameters. And four parts of our E^{2}ARS scheme are also implemented in the GridSim. The application requests are submitted to the Gridlets, and the time-optimizing scheduler is employed to distribute the requests to certain set of the cloud storage nodes. Table 1 summarizes important parameters used to resume real world cloud storage system. In addition, the data-partitioning algorithm is implemented in the GridSim to optimize the parallelism degrees of the request, and to get the value of $T_{partition}$. We compare the performance of a cloud storage system with E^{2}ARS against that of another system without employing E^{2}ARS. In this study, a system that does not apply E^{2}ARS is an ordinary cloud storage system with all nodes active all the time. We then evaluate effects of varying arrival rates, the desired response time and the number of the replicas of the data blocks on the performance of the two cloud storage system. Finally the performance impacts of parallelism degrees on the cloud storage system will be analyzed too.

Table 1: Nodes parameters of the simulated Cloud Storage System

Number of nodes	256
Block size	64MB
Average seek time of the disks of the node	6ms
Capacity	128TB
Spindle speed	7200RPM
Energy consumption rate	100w/h
Average bandwidth among nodes	4MB/S

In our simulation experiments, we mainly evaluate the following three performance metrics to demonstrate the effectiveness of the E^{2}ARS scheme.

(1) Average number of nodes powered down (or active) during a certain period time.

(2) Energy consumption is the total energy consumed by the cloud storage systems;

(3) Energy conservation ratio.

Furthermore, the overhead induced by our E^{2}ARS scheme will be discussed in the last subsection.

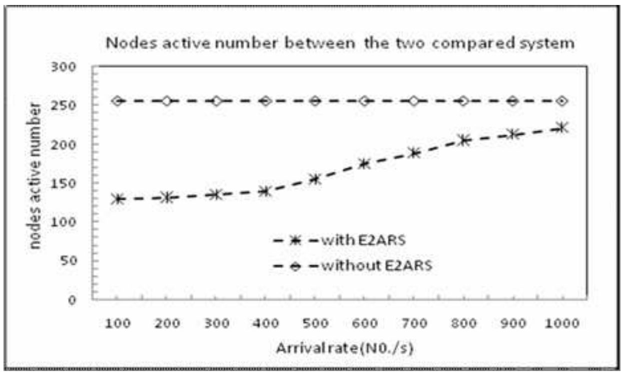

Fig. 2: Impact of arrival rate on the number of nodes active.

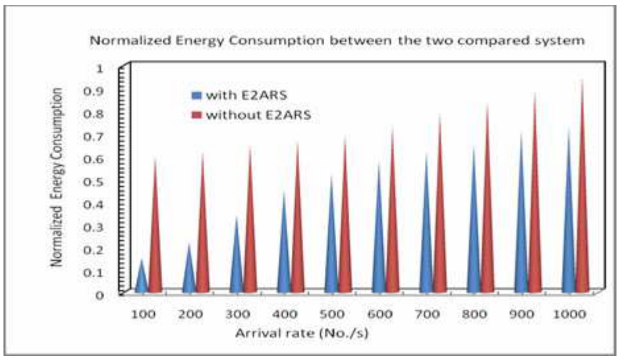

Fig. 3: Impact of arrival rate on the normalized energy consumption.

5.1 Impact of arrival rates

In this experiment, we evaluate the impacts of request arrival rate on the active nodes number, the normalized energy consumption and conservation ratio. We compare an E^{2}ARS-enabled storage cloud system with a non-E^{2}ARS-enbaled one. And vary the arrival rate from 100 to 1000 No./s with an increment of 100No./s, while the desired response time and the data sizes of the requests are constants. Fig.2-Fig.4 demonstrates the simulation experiment results.

Fig.2 verifies our anticipation when the workload is light some of the nodes can be powered down to save energy consumption.

Fig.3 shows that energy consumed by the E^{2}ARS-enable cloud storage system is lower than the non- E^{2}ARS-enable one. Furthermore, the lighter of the workload, the more energy consumption is saved by E^{2}ARS-enable system due to the nodes powered down.

Fig.4 demonstrates the energy conservation ratio of the E^{2}ARS-enable cloud storage system as the arrival rate increase.

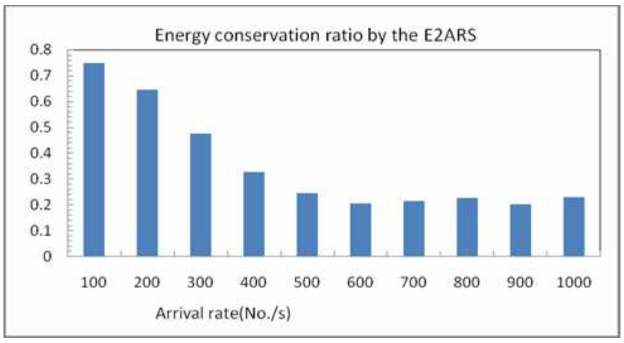

Fig. 4: Impact of arrival rate on the energy conservation ratio.

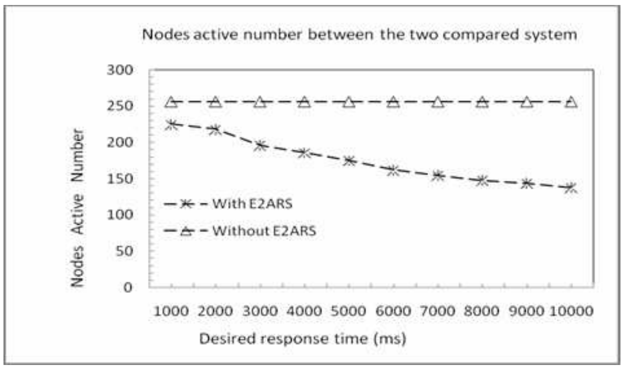

Fig. 5: Impact of desired response time on the number of nodes active.

5.2 Impact of the desired response time

In this subsection, we evaluate the impacts of desired response time on the active nodes number, the normalized energy consumption and energy conservation ratio. We compare an E^{2}ARS-enabled storage cloud system with a non-E^{2}ARS-enbaled one. And varying the desired response time from 1000 to 10000ms with an increment of 1000ms, where the arrival rate is set to 500No./s, and the data sizes of the requests is average to 1024MB. Fig.5-Fig.7 demonstrates the simulation experiment results.

Fig.5 verifies our anticipation when the desired response time is loose (that is the Qos level is low) nodes can be powered down to save energy consumption too. Fig.6 shows that energy consumed by the E^{2}ARS-enable cloud storage system is lower than the non-E^{2}ARS-enable one. Furthermore, the looser of the desired response time, the more energy consumption is saved by E^{2}ARS-enable system due to the nodes powered off. And the Fig.7 demonstrates the energy conservation ratio of the E^{2}ARS-enable cloud storage system as the desired response time increase.

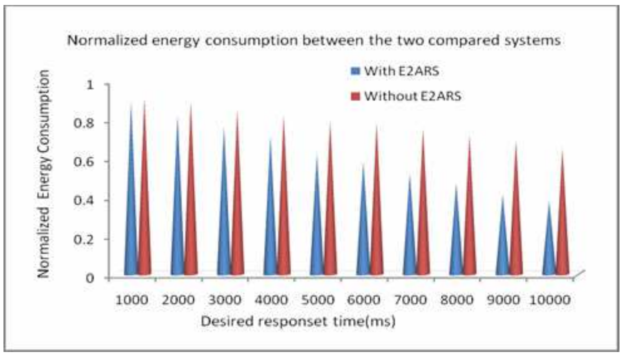

Fig. 6: Impact of desired response time on the normalized energy consumption.

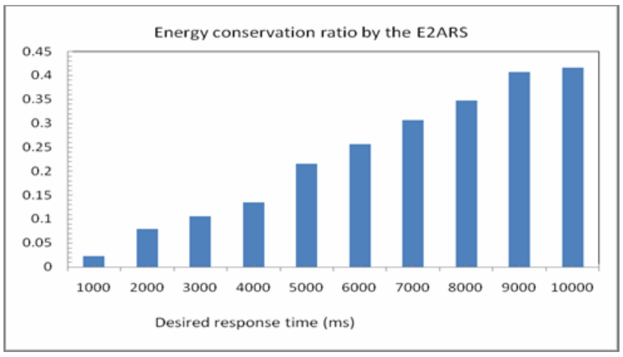

Fig. 7: Impact of desired response time on the energy conservation ratio.

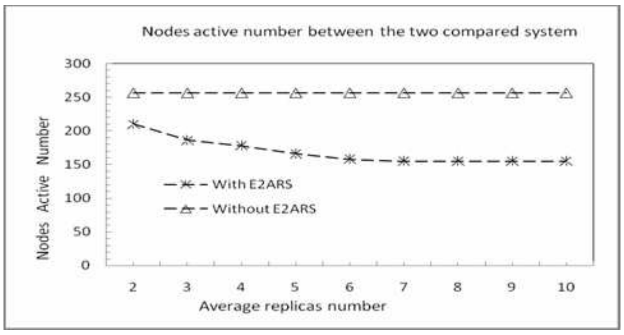

Fig. 8: Impact of average replicas numbers on the active nodes number.

5.3 Impact of average replica numbers

In this subsection, we evaluate the impacts of average replicas number on the number of active nodes, the normalized energy consumption and Energy conservation ratio.

We compare an E^{2}ARS-enabled storage cloud system with a non-E^{2}ARS-enbaled one. And varying the replica numbers from 2 to 10 with an increment of 1, when the

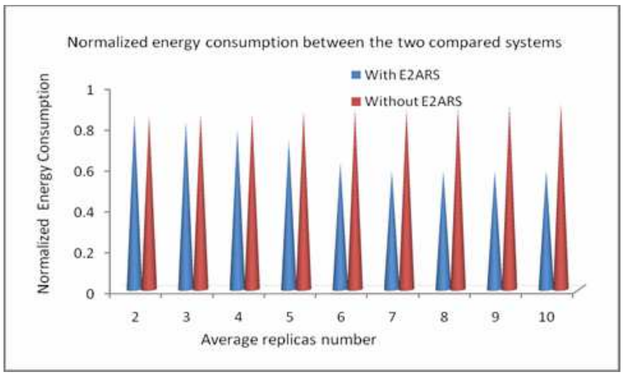

Fig. 9: Impact of average replicas numbers on normalized energy consumption.

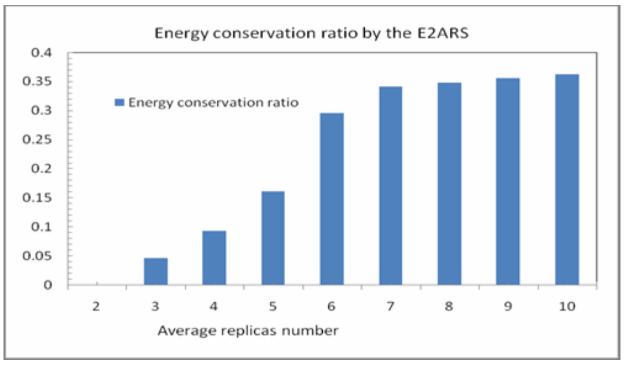

Fig. 10: Impact of average replicas numbers on energy conservation ratio.

arrival rate is set to 500No./s, and the desired response time is set to 2000ms. Fig.8-Fig.10 demonstrates the simulation experiment results. Fig.8 verifies our anticipation when the average replicas number is increased the nodes can be powered down increased, but when replicas number increased to a certain number, the active nodes number is fixed. Fig.9 shows that when the average replicas number varied, energy consumed by the E^2ARS-enable cloud storage system is lower than the non-E^2ARS-enable one. Furthermore, the more is the replicas number, the more energy consumption is saved by E^2ARS-enable system due to the nodes powered down, and the energy consumption saved became constant when the replicas number reach a certain value. Fig.10 demonstrates the energy conservation ratio of the E^2ARS-enable cloud storage system as average replicas number increase.

5.4 Impact of the parallelism degrees

In this subsection, we evaluate the impacts of parallelism degrees on the number of active nodes, the normalized

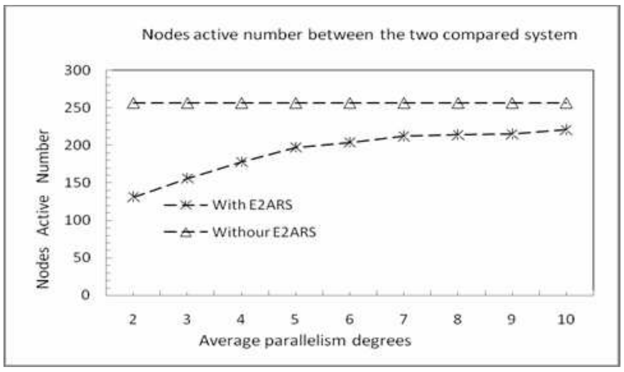

Fig. 11: Impact of the average parallelism degrees on active nodes number.

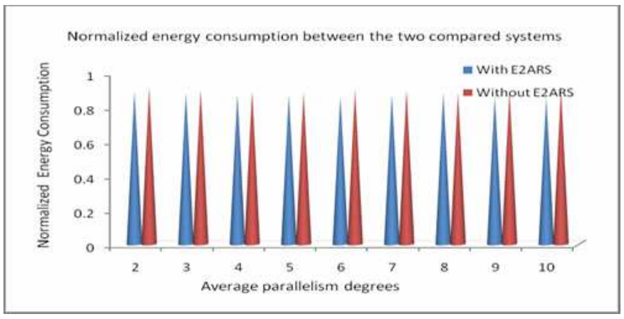

Fig. 12: Impact of the average parallelism degrees on normalized energy consumption.

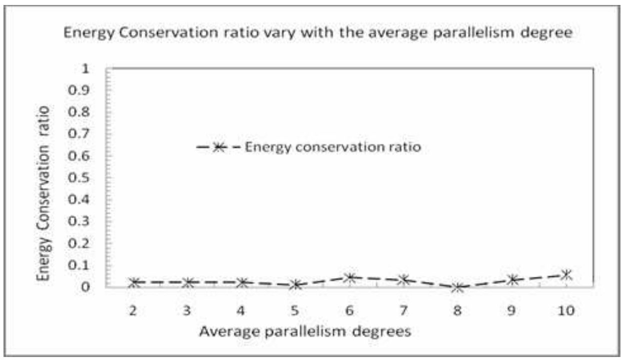

Fig. 13: Impact of the average parallelism degrees on the energy conservation ratio.

energy consumption, Energy conservation ratio, and the normalized response time. We vary the parallelism from 2 to 10 with an increment of 1, where the arrival rate is set to 500No./s, and the desired response time is set to 2000ms, and the average replicas number is set to 3. Fig.11-Fig.14 demonstrates the simulation experiment results Fig.11 shows that with the average parallelism degrees increased the number of nodes powered down

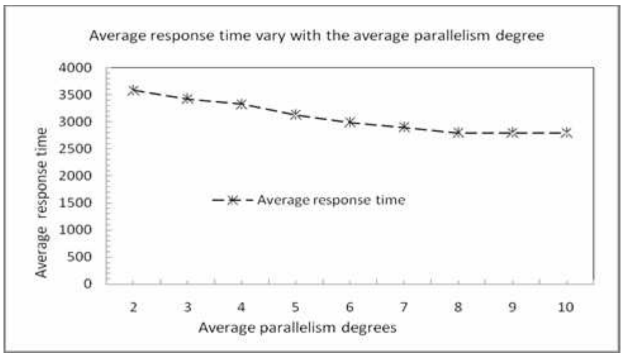

Fig. 14: Impact of the average parallelism degrees on the average response time.

decreased due to the related nodes of the request increased. The remaining figures (Fig.12-Fig.14) show that as the average parallelism degrees increase, energy consumption saved by our E^{2}ARS scheme is neglectful, but the average response time is shorten due to the parallel processing.

5.5 Overhead analysis

From the above subsections, we found that our E^{2}ARS scheme can save energy consumption almost at all of the related factors. However, in our simulation experiments, we also found the overhead induced by our E^{2}ARS. For example, when we increase the replicas number to save energy and guaranteeing the data availability, more storage capacity and the additional time used to manage the replicas are required. What is more, when employing data partition mechanism for parallel processing to shorten the average response time, the additional of partition time and the blocks managing overhead occur. Though the overhead will be induced by our E^{2}ARS, when come to energy consumption, our E^{2}ARS scheme is also valuable. Through our E^{2}ARS, energy conservation can attain 75% at some time (Fig.4) while satisfying the users' desired response time and the data availability.

6 Conclusion and Future Work

In this paper, we present an Energy-effective adaptive replication strategy (E^{2}ARS), in which data partition mechanism, minimal replicas determining model, replicas placement strategies and the adaptive gear-shifting mechanism are elaborately designed to maximize the energy conservation. Mathematical analysis is done exhaustively to confirm that our E^{2}ARS scheme will save energy consumption when the system's workload is light or desired response time is loose. And the simulation experiment results demonstrate that through our E^{2}ARS scheme energy consumption can be saved with Qos satisfied and data availability guaranteed when varying the arrival rate, desired response, replicas number and parallelism degrees. However, our E^{2}ARS scheme obtains the effective energy performance by inducing some storage capacity overhead. Our further work will be focused on minimizing the overhead while maximizing the energy consumption conservation.

Acknowledgement

This work is supported by Natural Science Foundation of Zhejiang Province (LY13F020045, LY13F020047, LQ12F02002), Natural Science Foundation of China (Grant No.61202094,6110004, 61003077,61100193), Important Science & Technology Specific Projects of Zhejiang Province of China (2010C13022) and National Key Technology Research and Development Program of the Ministry of Science and Technology of China (2012BAH24B04).

References

[1] Zhu, Q. and Zhou, Y. 2005. Power-aware storage cache management. IEEE Trans. Computer, **54**, 587-602 (2005).

[2] Kim, J. and Rotem, D. Energy proportionality for disk storage using replication. In Proceedings of the 14th International Conference on Extending Database Technology. EDBT/ICDT'11. ACM, New York, NY, USA, 81-92 (2011).

[3] X. Yao and J. Wang. Rimac: a novel redundancy-based hierarchical cache architecture for energy efficient, high performance storage systems. In EuroSys '06: Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems. ACM, New York, NY, USA, **2006**, 249-262 (2006).

[4] T. Xie. Sea: A striping-based energy-aware strategy for data placement in raid-structured storage systems. IEEE Trans., **57**, 748-761 (2008).

[5] E. Pinheiro and R. Bianchini. Energy conservation techniques for disk array-based servers. In ICS '04: Proceedings of the 18th annual international conference on Supercomputing, ACM, New York, NY, USA, 68-78 (2004).

[6] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes. Hibernator: helping disk arrays sleep through the winter. In SOSP '05: Proceedings of the twentieth ACM symposium on Operating systems principles, ACM, New York, NY, USA, 177-190 (2005).

[7] W. Lang, J. Patel, and J. Naughton. On energy management, load balancing and replication. Univ. of Winsonsin Technical Report, **1670**, (2010).

[8] D. Li and J. Wang. eraid: A queueing model based energy saving policy. In MASCOTS '06: Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation,Washington, DC, USA. IEEE Computer Society, 77-86 (2006).

[9] C. Weddle, M. Oldham, J. Qian, A.-I. A. Wang, P. Reiher, and G. Kuenning. Paraid: A gear-shifting power-aware raid. Trans. Storage, **3**, 13 (2007).
[10] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke. Reducing disk power consumption in servers with drpm Computer, **36**, 59-66 (2003).
[11] Tao Xie, SEA: A Striping-Based Energy-Aware Strategy for Data Placement in RAID-Structured Storage Systems. IEEE Transaction on Computers, **57**, (2008).
[12] Kaushik, R. T. and Bhandarkar, M. Greenhdfs: towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster. In Proceedings of the 2010 international conference on Power aware computing and systems. HotPower ’10. USENIX Association, Berkeley, CA, USA, 1-9 (2010).
[13] Kaushik, R. T., Cherkasova, L., Campbell, R., and Nahrstedt, K. Lightning: self-adaptive, energy-conserving, multi-zoned, commodity green cloud storage system. In Proceedings of the 19th ACM International Symposium on, (2010).
[14] H. Amur, J. Cipar, et al. Robust and Flexible Power-Proportional Storage. 2010. In the Proceeding of SoCC’10, Indianapolis, Indiana, USA, 10-11 (2010)
[15] Lakshmi Ganesh, Hakim Weatherspoon, Ken Birman. 2011. Beyond Power Proportionality: Designing Power-Lean Cloud Storage. IEEE 10th International Symposium on Network Computing and Applications (2011).
[16] Kim, J. and Rotem, D. Energy proportionality for disk storage using replication. In Proceedings of the 14th International Conference on Extending Database Technology. EDBT/ICDT’11. ACM, New York, NY, USA, 81-92 (2011).
[17] J. Kim and D. Rotem. Using replication for energy conservation in raid systems. In PDPTA’10, (2010).
[18] M. Nijim, X. Qin, M. Qiu, etc. An adaptive energy-conserving strategy for parallels disk systems. Journal of Future Generation Computer Systems, **29**, 196-207 (2013).
[19] T. Sumner, M. Marlino, Digital libraries and educational practice: a case for new models, in: Proc. ACM/IEEE Conf. Digital Libraries, 170-178 (2004).
[20] P. Scheuermann, G. Weikum, P. Zabback, Data partitioning and load balancing in parallel disk systems, VLDB Journal, **7** 48-66(1998).
[21] Qingsong Wei, Bharadwaj Veeravalli, Bozhao Gong, et al. CDRM: A Cost-effective Dynamic Replication Management Scheme for Cloud Storage Cluster. IEEE International Conferece on Cluster Computing, (2010).
[22] Hui-I Hsiao, David J. DeWitt. Chained Declustering: A New Availability Strategy for Multiprocessor DataBase machines. ICDE, (1990).

Xindong You is a lecturer of School of Computer Science and Technology, Hangzhou Dianzi University, China. She is with the Grid and Service Computing Lab in Hangzhou Dianzi University.Before joining Hangzhou Dianzi University, she was a PhD candidate in Northeastern University from 2002 to 2007. She received her PhD degree in 2007. Her current research areas include virtualization, distributed computing, etc.

Li Zhou is an associate professor of School of Computer Science and Technology, Hangzhou Dianzi University, China. She is with the Grid and Service Computing Lab in Hangzhou Dianzi University. Her current research areas include cloud computing, virtualization, etc.

Jie Huang is a lecturer of School of Computer Science and Technology, Hangzhou Dianzi University, China. He is with the Grid and Service Computing Lab in Hangzhou Dianzi University .Before joining Hangzhou Dianzi University, he was a PhD candidate in Zhejiang University from 2003 to 2008. He received his PhD degree in 2008. He current research areas include cloud computing, virtualization, geo-computation, etc.

Jilin Zhang is a lecturer of School of Computer Science and Technology, Hangzhou Dianzi University, China. He is with the Grid and Service Computing Lab in Hangzhou Dianzi University.Before joining Hangzhou Dianzi University, he was a PhD candidate in Beijing University of Technology from 2005 to 2009. He received his PhD degree in 2009. His current research areas include parallel processing, complier optimization, virtualization, distributed computing, etc. Dr. Zhang is a member of China Computer Federation (CCF) and the IEEE. Dr. You is a member of China Computer Federation (CCF)

Congfeng Jiang is an associate professor of School of Computer Science and Technology, Hangzhou Dianzi University, China. He is with the Grid and Service Computing Lab in Hangzhou Dianzi University. Before joining Hangzhou Dianzi University, he was a PhD candidate in Huazhong University of Science and Technology from 2002 to 2007. He received his PhD degree in 2007. His current research areas include power aware computing system, virtualization, grid computing, etc. Dr. Jiang is a member of China Computer Federation (CCF), IEEE and IEEE Computer Society.

Jian Wan is the director of Grid and Service Computing Lab in Hangzhou Dianzi University and he is the dean of School of Computer Science and Technology, Hangzhou Dianzi University, China.Prof.Wan received his PhD degree in 1996 from Zhejiang University, China. His research areas include parallel and distributed computing system, virtualization, grid computing, etc. Prof. Wan is a member of China Computer Federation (CCF).