Journal of Network and Computer Applications 74 (2016) 66–85

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Review

Reliability and energy efficiency in cloud computing systems: Survey and taxonomy

Yogesh Sharma [a,b,*], Bahman Javadi [a], Weisheng Si [a], Daniel Sun [b]

[a] School of Computing, Engineering and Mathematics Western Sydney University, Australia
[b] Software and Computational Systems, DATA61-CSIRO, Australia

ARTICLE INFO

Article history:
Received 30 April 2016
Received in revised form
4 July 2016
Accepted 12 August 2016
Available online 13 August 2016

Keywords:
Cloud computing
Virtualization
Reliability
Energy efficiency
Resource failure
Failure correlation

ABSTRACT

With the popularity of cloud computing, it has become crucial to provide on-demand services dynamically according to the user's requirements. Reliability and energy efficiency are two key challenges in cloud computing systems (CCS) that need careful attention and investigation. The recent survey articles are either focused on the reliability techniques or energy efficiency methods in cloud computing. This paper presents a thorough review of existing techniques for reliability and energy efficiency and their trade-off in cloud computing. We also discuss the classifications on resource failures, fault tolerance mechanisms and energy management mechanisms in cloud systems. Moreover, various challenges and research gaps in trade-off between reliability and energy efficiency are identified for future research and developments.

Contents

* Corresponding author at: School of Computing, Engineering and Mathematics Western Sydney University, Australia.
E-mail addresses: y.sharma@westernsydney.edu.au (Y. Sharma), b.javadi@westernsydney.edu.au (B. Javadi), w.si@westernsydney.edu.au (W. Si), daniel.sun@data61.csiro.au (D. Sun).

http://dx.doi.org/10.1016/j.jnca.2016.08.010
1084-8045/

1. Introduction

Cloud computing is the ongoing revolution in information and communication technology (ICT) that uses virtualization technology to provide a powerful and flexible computing environment. In a Gartner report published in January 2013, the growth of public cloud services will make it a $155 billion market and by the end of 2016, it is expected to grow to $ 210 billion. Although cloud computing makes the computing reliable, dynamic, fast and easy, it is still facing numerous challenges because of its large-scale and complex architecture. Considering the scale and complexity of cloud data centers, reliability and energy efficiency are two key challenges that need careful attention and investigation. *Reliability of cloud computing systems (CCS) can be defined in the context of security or in the context of resource and service failures.* Due to the complexity of the cloud architecture, failures are inevitable. It has been shown that a system with 100,000 processors experiences a failure every couple of minutes (Engelmann and Geist, 2005). In cloud computing, failures could occur due to multiple reasons such as hardware failure, software failure, etc. (Fig. 3). A failure in the services of a cloud costs significantly for both providers and customers. In a survey of 63 Data Centers done by P. Institute (2016) in 2016, it has been reported that the average down-time cost of each data center rose to $740,357 from $ 500,000 in 2010 (38% increase). Every hour, the business sector is expected to lose around $108,000 and according to the Information week, each year IT outages result in the revenue loss of more than $ 26.5 billion.[1] Provisioning of cloud resources accurately according to the demand of the applications plays a crucial role to make the CCS reliable and energy efficient. In cloud computing, it is hard to predict the requirement of resources accurately before or during submission of an application or task. Sometimes the provisioned resources remain underutilized or become over utilized. The average utilization of resources in cloud based data centers is only between 6% and 12%.[2] In case of underutilized resources, task or virtual machine consolidation is performed by migrating the running virtual machines to other physical resources in order to put the underutilized resources on sleep mode or to turn them off so as to reduce the energy consumption or other running costs (Clark et al., 2005). In the case of overutilization, the running tasks are migrated to other resources to keep the load of over-utilized resources below to a specific threshold to immunise them from failures or crashes.

On the other hand, the energy requirement to operate the cloud infrastructure is also increasing in proportion to the operational costs. Approximately 45% of the total operational expenses of IBM data centers goes in electricity bills (Sams, 2011). According to the Gartner, the electricity consumption by cloud based data centers will increase to 1012.02 Billion kWh by 2020. In 2013, data centers alone in U.S. consumed 91 billion kilowatt-hours, which is enough to power all the households of New York City twice over and if this trend will continue then the consumption will reach 140 billion kWh by 2020, a 35% increase.[3] The energy that the U.S. based data centers are consuming is equal to the electricity produced by 34 power plants each of 500 megawatts capacity and if this can't be reduced then 17 new power plants will need to be established by 2020 to power the data centers.[4] The electricity or energy consumption in cloud infrastructures is very inefficient and there are several types of wastes at different levels such as infrastructure level or system level (Nguyen and Shi, 2010). At the infrastructure level, half of the energy provided to a data center is consumed by the cooling infrastructure and at the system level, 50% of the energy is consumed when systems are in idle state. These types of waste cause financial loss to both providers and users.

Cloud computing infrastructure is a major contributor to the carbon content of the environment. Along with many contributors of carbon emissions in the environment, the contribution of IT infrastructure is equal to the aviation industry. U.S. based data centers emit 100 million metric tonne of carbon content each year and will increase to 1034 metric tonne by 2020 (Cook and Horn, 2011). As the energy consumption, heat release and carbon footprint from large computing infrastructures has increased, researchers are under great pressure to find new ways of decreasing energy consumption. In the last few decades, the primary focus of researchers and designers was on optimizing the performance of the system in terms of speed, space and efficiency. However, concerns about the energy consumption and carbon footprint intensified recently. In January 2015, Amazon has announced the construction of 150 MW wind farm which will produce approximately 500,000 MWh of wind power.[5] The operations of plant are expecting to start in December 2016. The energy generated by the wind farm will be used to power the current and future cloud based AWS (Amazon Web Services) data centers. Microsoft had also made a carbon neutral commitment in 2012 by promising to achieve zero emission of carbon content by their data centers, software development labs etc.[6] Google, IBM and other cloud vendors are also working to make the cloud services and cloud based data centers energy efficient and eco-friendly.

All the above facts and figures of failure and energy consumption lead to the requirement of management of cloud resources in a fault-tolerant and energy-efficient way. In response to this, various researchers worldwide have proposed many

[1] http://www.evolven.com/blog/downtime-outages-and-failures-understanding-their-true-costs.html
[2] http://www.nytimes.com/2012/09/23/technology/data-centers-waste-vast-amounts-of-energy-belying-industry-image.html
[3] http://www.vox.com/2014/12/14/7387945/sony-hack-explained
[4] http://www.computerworld.com/article/2598562/data-center/data-centers-are-the-new-polluters.html
[5] http://aws.amazon.com/about-aws/sustainable-energy
[6] http://blogs.msdn.com/b/microsoft-green/archive/2012/05/08/going-carbon-neutral-and-putting-an-internal-price-on-carbon.aspx

architectures, algorithms and policies to make the cloud computing environment reliable and energy efficient. However, there is very limited research on the trade-off between reliability and energy efficiency in CCS (Section 6). Considering both parameters at the same time would open new opportunities and challenges in the area of resource management and resource provisioning in cloud systems. This paper gives a comprehensive survey of the research done in the field of reliability and energy efficiency followed by an analysis of the trade-off between these two metrics in CCS.

The rest of this paper is organized as follows: Background of cloud computing and virtualization has been explained in Section 2. In Section 3, we introduce the causes of the failures in parallel and distributed computing environments like CCS. Section 4 highlights the research efforts done in the field of reliability and failure management. In Section 5, we present the survey of the research done to make the CCS energy efficient. Finally Section 6 analyse the trade-off between the reliability and energy efficiency followed by the various challenges for determining the suitable equilibrium between them. A taxonomy corresponding to each section has been developed.

2. Background

Cloud computing is a simple concept that has emerged from heterogeneous distributed computing, grid computing, utility computing and autonomic computing. National Institute of Standards and Technology (NIST) has given a very comprehensive and widely accepted definition of cloud computing systems. According to NIST (Mell and Grance, 2011).

> *Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.*

In cloud computing paradigm, end users avail computing as a service or utility from the remote infrastructure just like water, electricity, telephone etc. and pays for the usage. Users and businesses are able to access the computing services according to their requirements with minimum intervention such that without knowing where the services are coming from and how they are getting delivered. For keeping the scope of this survey limited to reliability and energy efficiency in cloud computing, only the term virtualization from cloud computing perspective has been explained briefly because of its intensive use in further sections. A thorough review about the history and trends in cloud computing can be seen in Buyya et al. (2009), Shawish and Salama (2014) and Jula et al. (2014).

Virtualization is the engine of cloud computing paradigm. Virtualization allows the running of multiple virtual machines (a software implementation of a computing node) on a single node simultaneously with different software stacks or configurations such as operating systems and application softwares. Generally, a computing node on which virtual machines (VMs) are running is termed as host machine and the running virtual machine is termed as a guest machine. The number of running VMs on a host depends upon the hardware configuration of the host and the configuration of VMs. A virtual layer called virtual machine monitor (VMM) lies in the middle of hardware and running VMs which ensures the isolation of the running VMs from each other and takes other managerial decisions such as resource scaling, resilience, fault tolerance, power management etc. With the great adoption of cloud computing technology, businesses need to shift

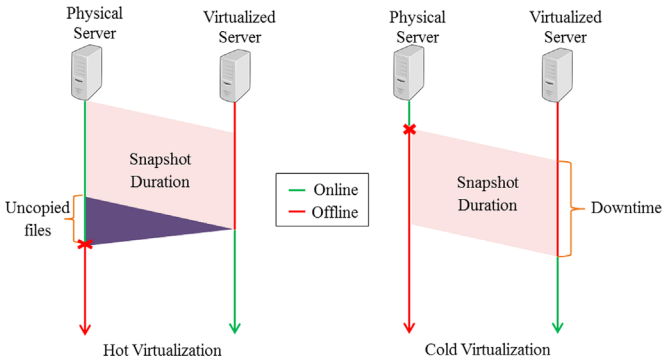

Fig. 1. Virtualization/cloning methods.

their IT operations which were initially running on in-house non-virtualized facility to virtualized environment. This can be done in two ways (Fig. 1): hot virtualization (hot cloning) and cold virtualization (cold cloning) (Portnoy, 2012).

In hot virtualization or hot cloning, the physical machine remains online or running while taking snapshot (creating disk image). Once the snapshot has been taken, the image gets copied on a virtualized machine or server. The benefit of hot virtualization is that we can keep the servers running all the time during the creation of image rather than taking them offline. In this way, the down time can be avoided and loss of business can be prevented. However, while creating the image some of the opened files may be left uncopied, which creates inconsistency between images. The alternate of hot virtualization is cold virtualization, in which the system goes offline and the disk image gets created. In cold virtualization, inconsistency can be avoided, however, the systems need to go offline, which cause loss to businesses (Subbiah, 2012). Although hot virtualization has been preferred over the cold virtualization because of no downtime, the choice matters on the requirements of the organization. In virtualization, the running VMs can be migrated from one server to another targeting different objectives such as fault tolerance, energy efficiency, operational costs, security, environment etc. Uses of VM migration to increase the reliability and to decrease the energy consumption of cloud computing systems have been discussed in Sections 4 and 5, respectively.

3. Failures in cloud and distributed computing environments

In this section, we review the classification of failures in cloud and distributed computing systems. The failure correlations as well as causes for failures are also discussed. According to Javadi et al. (2013).

> *A Failure is defined as an event in which the system fails to operate according to its specifications. A system failure occurs, when a system deviates from fulfilling its normal system function for which it was aimed at.*

According to Google (Barroso et al., 2013), the cost for each repair of failure includes $100 for technician's time and 10% of the total cost of server ($200), which reaches to $ 300 per repair. Therefore the cost of repairing the hardware exceeds its buying cost after only 7 repairs. Sound knowledge of the type of failure and causes of failure will help computer scientists and computer engineers to design more scalable algorithms and to deploy infrastructure in more fault tolerable way. This will help to reduce the repair/replacement cost and engineering expenditures and makes the computing, specifically service computing such as cloud computing, more reliable. Failures in CCS result in loss of business

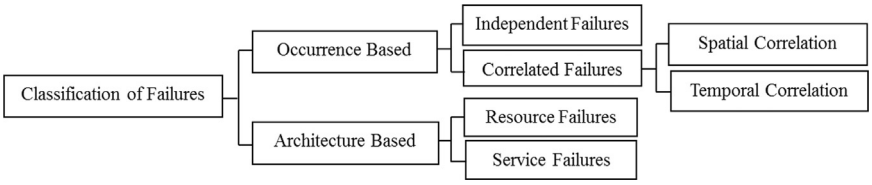

Fig. 2. Classification of failures.

due to the diversion of users to other vendors.

3.1. Classification of failures

Based on the characteristics of the failures in cloud computing, we have generated two different classes of failures: architecture based and occurrence based (Fig. 2). In the architecture based classification, the failures are further divided into two categories, Resource Failure and Service Failure. As name implies, resource failure is caused by the outage of some physical resources like system breakdown, network or power outage, software error etc. Most of the work on the failure tolerance in the literature has focused on resource failures (Javadi et al., 2012; Fu, 2010; Philp, 2005; Vishwanath and Nagappan, 2010). Resource failures could occur at the provider or the client end. Service failure in cloud computing means that the cloud provider is unable to provide, or the user is unable to get, the services promised in the service level agreements (SLAs). Resource failure could lead to a service failure but service could fail even in the presence of working resources during peak loads (Section 3.2.4).

The occurrence based classification of failures is all about the interconnection between the failures, whether or not the occurrence of one failure leads to the occurrence of another in the system. Occurrence based failures are further divided into two categories independent failures and correlated failures. Independent failures occur discretely. This type of occurrence is hypothetical because the literature has demonstrated that there is a correlation between failures (Fu and Xu, 2007; Gallet et al., 2010; Yigitbasi et al., 2010; Schroeder and Gibson, 2010). In correlated failures, the occurrence of a failure leads to the occurrence of other failures in the system. The failures could be correlated in two different ways: spatial correlation and temporal correlation. A complete survey about the correlated failures is discussed in Section 3.3.

3.2. Causes of failures

To make CCS more reliable and available all the time, it is very important to understand the causes of the occurrence of the failures. Various causes of failures in cloud computing are given below in Fig. 3.

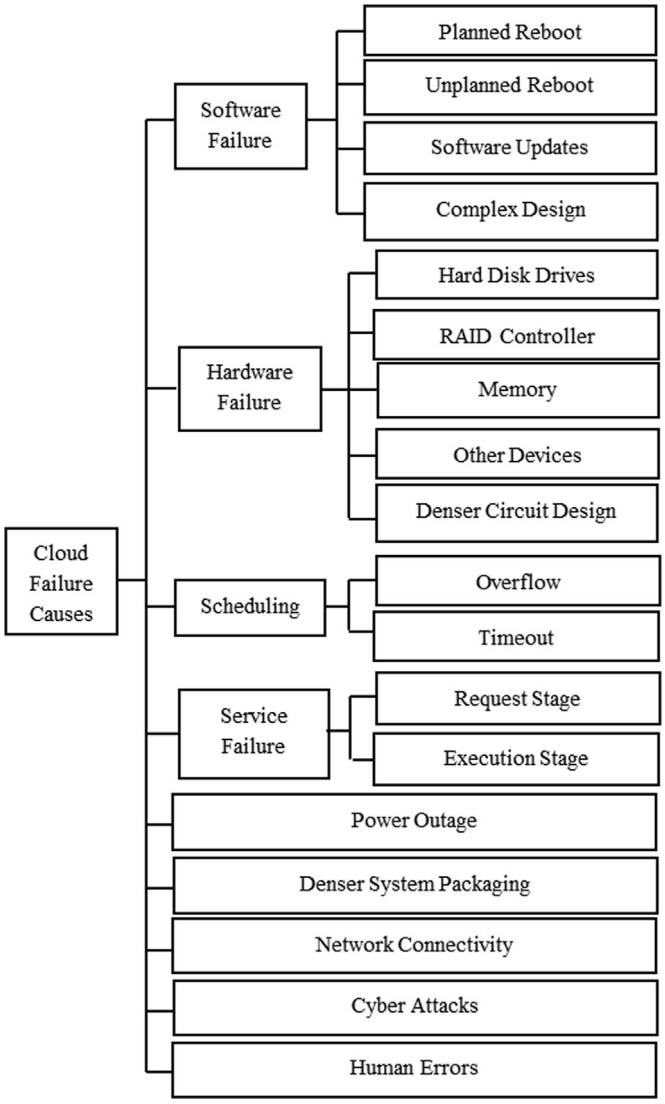

Fig. 3. Causes of failure in cloud computing.

3.2.1. Software failure

As software systems and applications are getting complex day by day, they became a significant reason of system breakdown which causes loss in business and revenue. In October 2013, Knight Capital's[7] cloud based automatic stock trading software went down for 45 min because of an error in trading algorithm which costed $440 million to the company. Sometimes an unexpected error could occur during the process of updating the software, causing the whole system to crash down. In 2013, cloud services of Microsoft were interrupted for 16 h. It was revealed that they were performing a regular process of updating the firmware in a physical region of the data centers. Something went wrong, which brought down the whole system.[8] Another major service outage had seen in January 2015 for 20 min, in which Yahoo Inc. and Microsoft's search engine, Bing, went down during the code update.[9] After the crash, the roll back mechanism of Microsoft did not work, which forced the service to shut down from the linked servers to get the point where the system was operating correctly. After a successful update or due to the system maintenance, sometime reboots are scheduled by the service provider about which the service users are informed in advance.

[7] http://nypost.com/2013/10/26/knight-capital-computer-meltdown-just-waiting-to-happen/

[8] http://www.datacenterdynamics.com/focus/archive/2013/03/overheating-brings-down-microsoft-data-center

[9] http://techcrunch.com/2015/01/02/following-bing-coms-brief-outage-search-yahoo-com-goes-down-too/

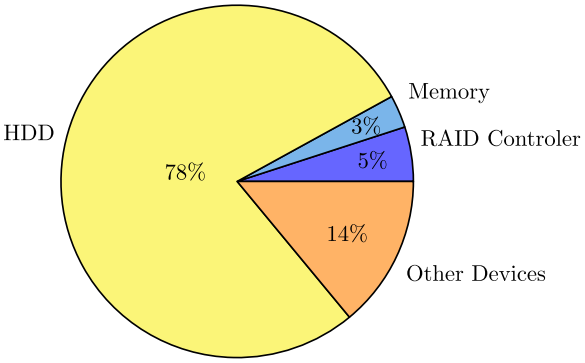

Fig. 4. Percentages of hardware component failures.

Most of the times during planned reboots, service providers consider some backup measures to provide an uninterruptable service to users. On the other hand, unplanned reboots happen after inconsistency in data integration after software or hardware update and the average cost of an unplanned reboot is $9000 per minute. According to Brian Proffitt,[10] up to 20% of attempts are failing in the deployment of software as a service due to the problem of data integration. So it is important to shift application design paradigms from machine-based architecture to cloud-based architectures. Some of the other causes of system failure or performance degradation due to the softwares are memory leakage, unterminated threads, data corruption, storage space fragmentation and defragmentation (Vaidyanathan et al., 2001).

3.2.2. Hardware failure

Hardware failure represents around 4% of all the failures occurred in cloud based data centers. Among all the hardware failures/replacements, 78% are hard disk drives (Fig. 4) (Vishwanath and Nagappan, 2010). In 2007, hard disk drives and memory modules were the two most common hardware components sent by Google for repair (Barroso et al., 2013). Hard disk failures increases as the size and age of the clusters increase. Vishwanath and Nagappan (2010), has shown that with age, failure in hard disk drives (HDD) grows exponentially, but after a saturation point it becomes stable. HDD failures can be reduced by timely replacement, and a increase in system reliability will result.

3.2.3. Scheduling

In the cloud computing architecture, schedulers are responsible for scheduling the requests on the provisioned resources meeting the user requirements. Requests waiting to get scheduled are initially placed on an input queue. On the basis of the current computing and data resource availability, scheduler schedule the requests in the form of tasks or subtasks to the resources. Being a restricted data structure, queue has a limitation to store a specific number of requests. Exceeding the number of requests than the length of queue will cause drop of new requests and service will be unavailable to the users. This is called overflow failure. To avoid the overflow of queues, timeout value is assigned to each request. If the request waiting time in the queue exceeds the specified time out value, then the request will be dropped from the one to make way for fresh requests. This is called timeout failure. This will lead to the service outage in terms of SLA violation due to the delay in cloud computing services. Failure prediction (Salfner et al., 2010) plays an vital role in identifying system resources that are prone to failure. Scheduler can then avoid placing tasks on those resources that are less reliable. The more accuracy of the prediction means less failure in the services.

3.2.4. Service failure

In CCS, service failure can happen with or without resource failure. As stated by Dai et al. (2010), the cause of the cloud service failure depends upon the stage of the submitted job such that request stage and executing stage. During the request stage, all the requests with service requirements submitted by users are kept in the ready queue. During this stage, users may not be able to access the services because of overflow or time-out that happens due to overloading of resources such that during peak hours. In such case, the underlying resources are working fine but they are unable to accommodate more requests and service failure happens. On the other hand, at execution stage, requests are submitted to underlying physical resources. If services get interrupted, it means the cause of service failure is the outage of resources.

3.2.5. Power outage

In cloud based data centers, about 33% of the service degradation has happened due to the power outage. This happens because of natural disasters or war zones. In 2012, out of 27 major outages of cloud computing services, 6 were caused by the hurricane Sandy alone.[11] In 2011, massive tsunami in Japan put the whole country in power crisis for a long time, and all the consumer services were affected. It is estimated that natural disasters contribute around 22% in cloud computing service outage. An another major cause of power outage is UPS system failures, which contributes 25% of total power outage failures and cost around $1000 per incident.

3.2.6. Denser system packaging

Whatever the infrastructure was built ten years ago is now outdated because the data storage has increased exponentially. Designers have begun to design very dense servers like blade servers to keep the storage space low. Total floor space required to setup an IT infrastructure has reduced by 65%,[12] which increased devise density per square feet and outage cost has risen to $99 per square feet. As a result of the high devise density, heat release increases, which causes a rise in temperature and this affects the working of devices. Facebook has revealed that by packing the machines densely, electrical current began to overheat and melt Ethernet sockets and other crucial components. In 2013 data centers of Microsoft faced a severe outage of 16 h that affected its cloud services including Outlook, Hotmail, SkyDrive and Microsoft's image sharing service[13] due to overheating issues.

3.2.7. Network infrastructure

In distributed computing architecture, specifically in the case of cloud computing, all the services are provided by communication networks. The whole information has been stored and exchanged between servers by using the networks. The outage of the underlying network results in the outage of the services of a CCS. For few cloud based applications such as real time applications, performance of networks plays a key role. A small increment in the network delay can be termed as an SLA violation which will be considered as a service failure. The network services could be broken physically or logically. Around 3% of the service failures happened due to the loss of network connectivity. There are

[10] http://readwrite.com/2013/03/05/software-as-a-service-the-dirty-little-secrets-of-saas

[11] http://www.rightscale.com/blog/enterprise-cloud-strategies/lessons-learned-recent-cloud-outages

[12] http://www.emersonnetworkpower.com/documentation/en-us/latest-thinking/edc/documents/white%20paper/energylogicreducingdatacenterenergyconsumption.pdf

[13] http://www.datacenterdynamics.com/focus/archive/2013/03/overheating-brings-down-microsoft-data-center

various challenges corresponding to the networks such as hop count, bandwidth, encryption, etc that need to be taken care of to make cloud computing services reliable.

3.2.8. Cyber attacks

Cyber attacks are the fastest growing reason of the data center outages. According to Ponemon Institute report (P. Institute, 2016), the percentage of data center outages due to cyber attacks was 2% in 2010, which had risen to 18% by 2013 and the latest percentage is 22%. The average downtime cost of outage by cyber attacks is $822,000. IBM's report on cyber security intelligence[14] has argued that 55% of cyber crimes or threats were from people having access to organization's systems, such that employs. Among other technical issues such as trojan attacks and software loopholes, social engineering (Abraham and Chengalur-Smith, 2010) is a major cause of cyber attacks. In social engineering attackers play with human psyche by exploiting them with emotions, fear, greed, etc and manipulate them to leak the confidential information.

3.2.9. Human errors

Along with cyber attacks, human errors also has a big weight (22%) for the causes of failures in CCS with average cost of $489 per incident. But it has been argued by Schroeder and Gibson (2010) that the lack of experience is a main reason of occurrence of human errors. In the survey done by Bianca, it has been seen that the proportion of human errors is higher during the initial days of deployment of infrastructure. This clearly shows that administrators gains more experience with the time, which reduces the occurrence of human errors. Similar to cyber attacks, social engineering is also a reason for human errors.

3.3. Failure correlation

Correlation is all about the interdependency of activities. If a failure has happened in a part of the system that leads to failures in other parts of the system, which could results in the failure of whole system then it can be said that there is some correlation between these failures. In distributed computing systems such as clouds and grids, if multiple computing components are affected by a common failure then that set or group of computing components is called a shared risk group or shared risk domain because they share a common failure risk (Pezoa and Hayat, 2014) just like a communication medium in the network topologies. If the communication medium breaks down then all the data transfer between the nodes using same communication medium will go down. Earlier, most of the research to make cloud environments reliable has been done by considering the independent distribution of failures (Mickens and Noble, 2006), which makes the evaluation simpler but error prone in practice. It has been proved that a single faulty node can influence the working of whole system (Wang and Wang, 2014). Even the co-occurrence of failures reduces the effectiveness of various fault tolerance mechanisms such as encoding schemes, replication and back-ups (Rangarajan et al., 1998). Failure correlation can be based on time (temporal correlation) or space (spatial correlation).

3.3.1. Space correlated failures

Failures are called spatially-correlated if occurs within a short time interval on different nodes of the same system (Fig. 5). Occurrence of failures in a failure burst could be correlated in space and proven empirically or numerically. To prove the correlation between the failures in space, general numerical methods are required. As a result, Gallet et al. (2010), proposed a numerical method or model based on three lognormal distribution based aspects such that downtime due to failures, group arrival and group size so as to find the space-correlation between failures occurring during short time intervals. In the given model, a moving window based method has been used to the find the correlation between the failures in the empirical data. The data was taken from Failure Trace Archive (FTA) (Kondo et al., 2010), a public failure repository. It has been found that seven traces out of fifteen shows a strong correlation between the occurrence of failures which has challenged the assumption that the occurrence of the component failures are independently distributed.

3.3.2. Temporal correlated failures

Temporal correlation is about finding the periodicity in the pattern of occurrence of failures. One of the best methods to find temporal correlation is Auto-Correlation Function (ACF). As shown in Fig. 5, if the value of ACF is near to zero then the occurrence will be considered as random and if value is equal to or nearly equal to 1, it means there is some periodicity. Rangarajan et al. (1998), have identified that the failures occurred in large scale distributed computing systems are not uniformly distributed to all the nodes. Only small number of nodes (less than 4%) are prone to 70% of the failures occurred in the system. They also found a strong time varying failure correlation in the pattern of occurrence of failures on these nodes. Yigitbasi et al. (2010), measure the degree of correlation of the failure information gathered from various failure traces with different time lags by using an autocorrelation function. In their work, they shift the plot generated from the failure information according to different lags such as hours, days and weeks to find a repeated pattern. In their work they measured the behavior of failures by varying the time in large distributed systems. To characterize the repetition pattern of the failures and peaks in failures, a formal method has been proposed by the authors to identify the periods that are responsible for the downtime of the system.

4. Reliable cloud computing services

Reliability in cloud computing is how consistently a cloud computing system is able to provide its services without interruption and failure. Generally the reliability is defined as.

The ability of an item to perform a required function under stated conditions for a stated time period (Quality, 2010).

Cloud computing is a service-oriented architecture so the attributes of the reliability rely on service models such as, Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). To make cloud services reliable, both service providers and service users have their own responsibilities that vary according to the service model. To avoid service failure and to provide resiliency, three different design principles for reliable services (Fig. 6) have been proposed by Mike et al. (2014) from Microsoft Corporation. Good design following the given principles will minimize the effect of failures and enhance system resilience so that there is minimal interruption to services. If a failure event has occurred at a particular instance, then partial or even delayed services need to be delivered. Once the failure has happened, important measures to recover the service from the degradation due to failure also needs attention. The recovery should be done with minimum intervention of human. Various mechanisms such as checkpointing, redundancy, etc. (Section 4.1) have been proposed to recover the services of cloud computing upon failure. During the event of failure and process of recovery from the failure, data integration is a big concern. To avoid inconsistency in the data, mechanisms have to be implemented. On the other hand,

[14] http://public.dhe.ibm.com/common/ssi/ecm/se/en/sew03073usen/SEW03073USEN.PDF?

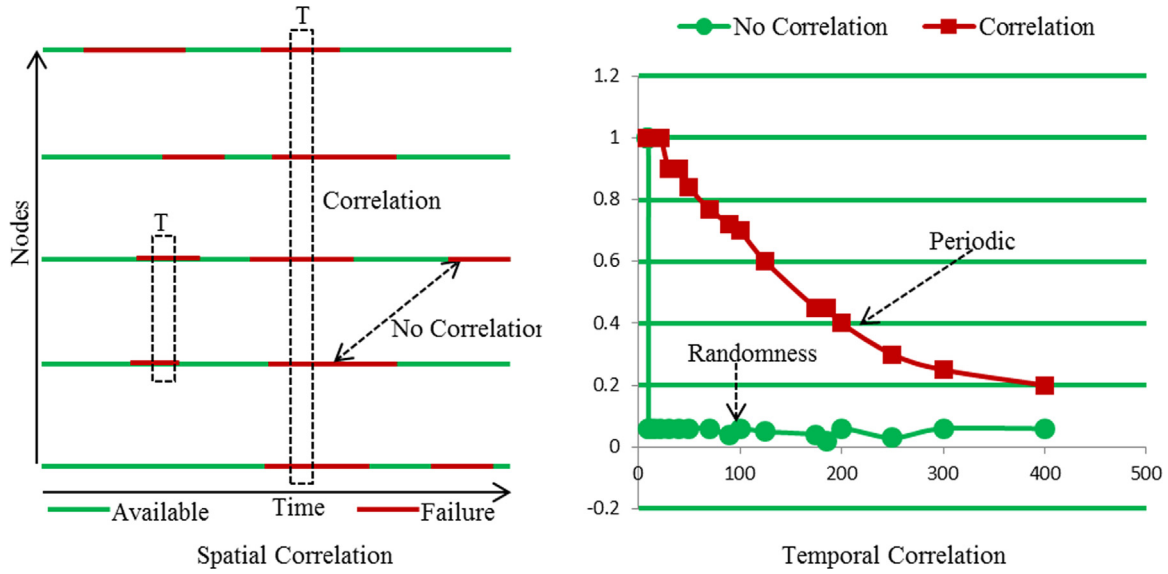

Fig. 5. Spatial and temporal failure correlation.

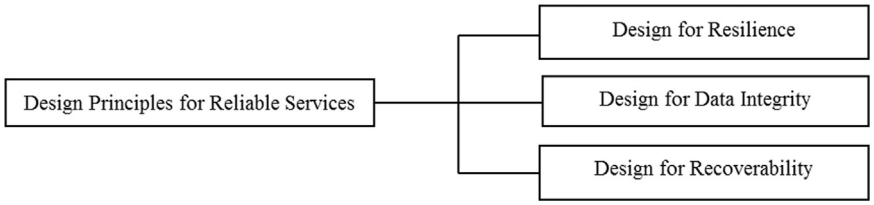

Fig. 6. Design principles for reliable cloud computing services.

data security is also an issue in these days. There are various incidents in history such as the Sony pictures entertainment hack, Dropbox leakage and icloud leakage that highlights the need to preserve the integrity of the data to make the services reliable and trustable.

4.1. Service failure management in cloud computing

To provide reliable services in cloud computing, one needs to manage service failures. All the proposed architectures and techniques designed for well-behaved cloud environment have to be redesigned for a failure-prone cloud environment. To manage resource failures in computing environment for reliability assurance, various techniques and methods have been proposed and implemented (Table 1). Since the service-oriented architecture is used by cloud computing, all the techniques and methods need to be explored from the perspective of service reliability. All the failure management techniques are categorized into two groups (Fig. 7).

4.1.1. Reactive failure management

In reactive failure management, measures are taken after the occurrence of failure. The working of reactive failure management techniques is similar to the working of reactive routing protocols in networks (Sharma et al., 2010). In reactive routing protocols, there are no routing tables. All the routes are created on demand. In the same way, whenever failures have occurred in cloud services, the required measures will be taken by restarting the services from the last execution instance recorded earlier using checkpointing or logging.

Checkpointing is a widely adopted reactive fault tolerance technique, in which the current state of a running process is saved on some backup resources and on the occurrence of failure, the process will be restarted or rolled back by using the last saved state. It has proved that the systems running without checkpointing take exponential time to complete the task (Duda, 1983). By using checkpointing, the exponential time becomes linear. On the basis of the working principle, checkpointing has divided into three different categories (Elnozahy et al., 2002) such as Uncoordinated Checkpointing (Random Checkpointing), Coordinated Checkpointing (Periodic Checkpointing) and Communication Induced Checkpointing (Fig. 7). Various cloud management software suits such as UniCloud by Oracle, Intel's Data Center Manager (DCM) are incorporated with the checkpointing mechanism to provide uninterruptable cloud computing services. It has been argued that in the large-scale systems like clouds, checkpointing mechanisms could create large overheads as well, if performed frequently (Fu, 2010). It has been estimated that the checkpointing creates overhead of 151 h for a job of 100 h in the petaflop systems (Philp, 2005). However, if a running program check pointed infrequently after long intervals, then it will make the re-execution of program lengthy after the failure, which will increase the total execution time of the program. The problem of determining the intervals for checkpointing is called optimal checkpoint interval problem. In the literature, finding the optimal checkpointing interval attracts many researchers (L'Ecuyer and Malenfant, 1988; Daly, 2006).

Replication is another reactive method to provide fault tolerance in which the backup resources are used to run replicas of the running processes. On the basis of updating of running replicas to handle the inconsistency, replication has divided into two categories called Primary Backup (Passive) replication and Active replication (Fig. 7). Various cloud computing providers use replication mechanism to provide fault tolerance at different levels. Microsoft's Azure uses virtual machine replication to provide fault tolerance at the cloud level. In the case of the failure of a virtual

Table 1
Survey of failure management mechanisms in cloud computing.

Authors	Service failure management	Failure management method	Objectives	Architecture	Workload
Meroufel and Belalem (2014)	Reactive	Checkpointing	System Availability	Cloud	Communication Induced
Fu (2010)	Proactive	VM Migration	System Availability, Resource Utilization	Virtualised Clusters	HPC
Walters and Chaudhary (2009)	Reactive	Checkpointing, Replication	System Availability, Resource Utilization	Virtualised Clusters	HPC
Guerraoui and Schiper (1996)	Reactive	Replication	System Availability, Resource Utilization	Clouds	
Jhawar et al. (2013)	Proactive	VM Migration	System Availability, Resource Utilization	Clouds	Multiple
Gao and Diao (2010)	Reactive	Replication	System Availability	Cloud	Web
Sun et al. (2012)	Reactive	Data Replication	System Availability, Resource Utilization	Cloud	
Bala and Chana (2015)	Proactive	VM Migration	System Availability	Cloud	Scientific
Bonvin et al. (2010)	Reactive	VM Replication	System Availability, Resource Utilization	Cloud	
Liu et al. (2009)	Reactive, Proactive	Checkpointing, VM Migration	System Availability, Resource Utilization	Virtualized Clusters	Web
AlZain et al. (2012)	Reactive	Replication		System Security, Resource Utilization	Multi-Clouds
Yu et al. (2015)	Reactive	Replication	System Availability, System Security	Public Cloud	
Nguyen and Shi (2010)	Reactive	Replication	System Availability, Resource Utilization	Cloud	Multiple
Cully et al. (2008)	Reactive	Checkpointing, Replication	System Availability	Virtualised Systems	Web
Jung et al. (2013)	Proactive	VM Migration	System Availability, Resource Utilization	Cloud	Multiple
Javadi et al. (2013)	Reactive	Checkpointing	System Availability, Resource Utilization	Virtualised Clusters	Parallel
Javadi et al. (2012)	Reactive	Checkpointing	System Availability, Resource Utilization	Hybrid Clouds	HPC
Voorsluys and Buyya (2012)	Reactive, Proactive	Checkpointing, VM Migration	System Availability, Resource Utilization	Cloud	Compute-Intensive
Yao et al. (2013)	Proactive	VM Migration	System Availability, Resource Utilization	Cloud	

machine, Azure always keeps replicated VMs to take charge of the failed VM. At Infrastructure as a Service level, OpenStack, an open-source cloud computing platform uses data replication to store data by writing the files and objects at multiple disks spread throughout the servers in the data centers. There are many more examples where the replication is in use like DFS replication, Apache Hadoop, Amazon EBS etc. A complete survey of replication mechanisms has been done by Guerraoui and Schiper (1996). The biggest challenge to run the replicas of a process is to maintain the consistency between the replicas and propagation of update messages. Various methods and mechanisms to handle the challenges and use of replicas in cloud computing environment can be seen in Table 1.

Logging or message logging protocols. Each process is recorded or saved in its present state and messages are sent periodically as the logs at some stable storage. When a process crashes, a new process is created on the place of a crashed process by using the recorded logs. To get the pre-failure state of a crashed process, all the logged messages are evaluated in the same order in which they were generated. Once the new process has created after a crash, the state of the new process should be consistent with other running processes. If the state of the process remains inconsistent then the process will be known as orphan process. To reduce the overhead of logging, checkpointing is incorporated with logging (Table 1). Once the checkpoint has been saved for the state of a process then all the logged messages before the checkpoint can be removed to save storage space. We classify the process of logging into two classes: Orphan process based and Storage based. These are further combined with each other to make more classifications (Meyer et al., 2014) (Fig. 7). In the upper sections, various coordinated methods are used to provide fault tolerance in distributed systems. Because of the overhead generated by the coordination between the processes, they have scalability issues. The uncoordinated methods such as message logging seems to be a good option in terms of application makespan for CCS. Lemarinier et al. (2004), have shown that if the mean time between failures (MTBF) is less than 9 h then messaging logging is a better option than the coordinated checkpoint because of less overheads.

4.1.2. Proactive failure management

Due to the large overhead and expensive implementation of reactive failure management mechanisms, cloud service providers have begun to adopt proactive failure management mechanisms. In proactive failure management, the prevention measures have been taken before the occurrence of failure. The productivity of proactive failure management methods depends upon the prediction of the occurrence of the failures (Fu and Xu, 2007; Islam et al., 2012). On the basis of the failure prediction results, the running processes are migrated from the suspected resource to other healthy resource for an uninterruptable execution. The accurate prediction of the occurrence of failure will make the failure management more efficient and reliable. Failure prediction is classified into two categories: offline failure prediction and online failure prediction. A complete survey about the failure prediction methods has done by Salfner et al. (2010). After the results of the failure prediction methods, suitable actions are taken by proactive fault tolerance mechanisms. Migration is the method that is used to provide fault tolerance by incorporating failure prediction methods. With the introduction of high speed networks and distributed architecture of computing, the migration of running tasks became possible. With the emergence of cloud computing, the migration has divided into the process migration (Milojičić et al., 2000) and virtual machine migration. By considering the dynamic nature of the cloud infrastructure, only virtual machine (VM) migration based fault-tolerance methods have been considered in Table 1. To migrate the running VMs from a faulty server to health

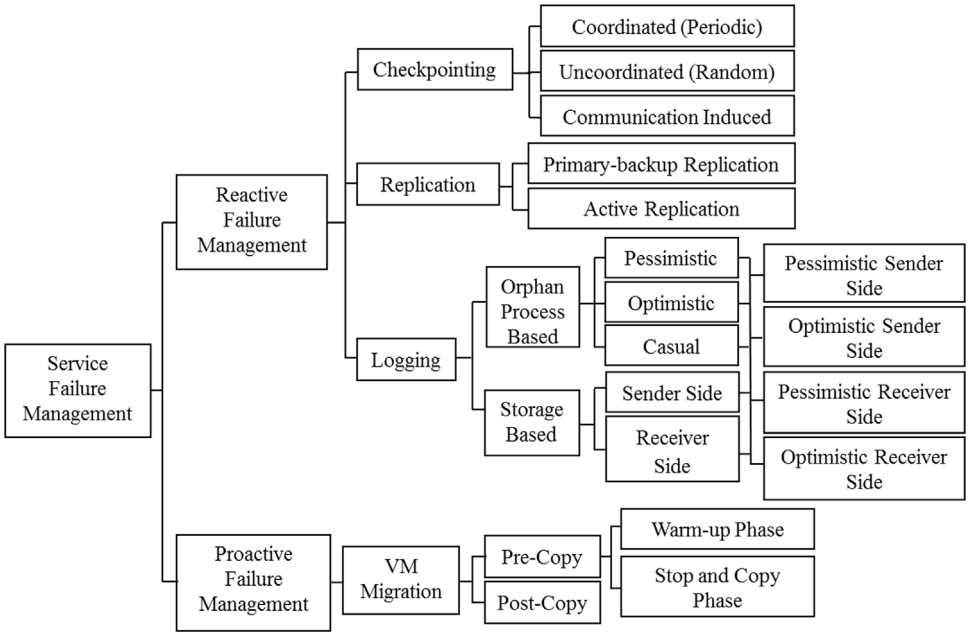

Fig. 7. Failure management in cloud computing.

one, two methods have been proposed in the literature: Pre-Copy and Post-Copy (Fig. 7).

Pre-copy VM migration approach: The pre-copy approach (Shribman and Hudzia, 2013) has two different phases: Warm-up Phase and Stop-and-copy Phase. In warm up phase, hypervisor copies the state of the running VMs such as CPU state, memory state, and state of other devices from a faulty server to the destination server. As the warm-up phase completes, the virtual machine stops at the source machine and stop and copy phase initiates. The stop and copy phase copies the remaining files or pages (if any) in the memory that gets modified (dirty pages) during the warm-up phase. After the transfer of all the pages the virtual machine resumes its execution over the destination machine. The time between the suspension of a virtual machine from the source node and resumption over the destination node is called down-time. Many of the hypervisors such as VMware, Xen, KVM are using pre-copy migration approach (Ma et al., 2010).

Post-copy VM migration approach: In post copy approach (Hines et al., 2009), the running VMs gets suspended at the source nodes and migrated to the destination nodes with partial attributes of the execution state such that CPU state, register usage etc. After getting the destination, the VMs resumes with the execution. In parallel the source machine also stay active serving the migrated VMs. Whenever a VM do not find a page in its local memory, it generates a page fault (network fault). On the generation of a network fault or page fault, destination machine redirects the page request to the source machine which in-turn responds with the faulted page. In general, the memory image can be transferred in the background after execution of VM at destination or it can be transferred on-demand in response of network fault.

As stated earlier, along with providing reliability to the services and optimized resource utilization, virtual machine migration has also been proved as a very promising technique to manage the energy consumption in CCS. Thorough details about the mechanisms used to manage the energy consumption in cloud computing paradigm are discussed in the next section.

5. Energy management in cloud computing

Along with the reliability of cloud computing services, energy consumption by the underlying complex infrastructure providing cloud services is also a big concern for cloud service providers. As increasing the reliability of cloud services makes it profitable by attracting more users or clients, decrease in the energy consumption will make it even more profitable by reducing the operational expenses of underlying infrastructure in terms of electricity bills. Besides the construction of data centers by adding temperature monitoring equipments, optimized air vent tiles, putting plates to block cold air passing through the racks, designing of optimized software systems is also very important for the proper utilization of resources of cloud infrastructure to increase the energy efficiency. As shown in Fig. 8, energy consumption can be optimized at the hardware level, software level and intermediate level. In the following sections, we have explored different techniques and methods to regulate the energy consumption in CCS. A complete list of the existing most energy

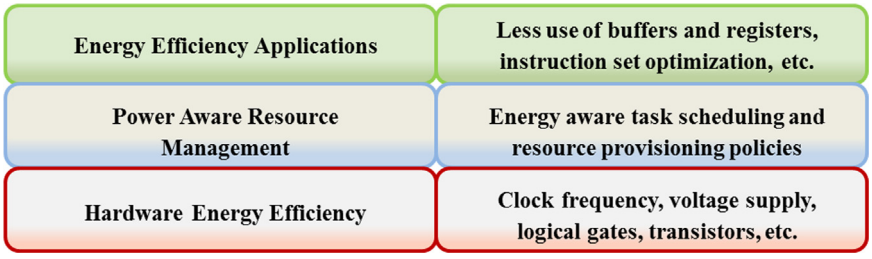

Fig. 8. Levels of energy efficiency enhancement.

efficient distributed computing systems is provided by Green500.[15]

In some studies, problem of high power consumption and high energy consumption has considered separately (Beloglazov et al., 2011). But because of the direct proportional relation between the energy and power consumption (Eq. (1)), both energy and power have been used interchangeably in this study and this has done by many studies in this domain (Faragardi et al., 2013).

$$E = PT \tag{1}$$

5.1. Static power management

Also known as offline energy management deals more with circuitry systems. It is more engineering oriented approach. In static management of power, whole optimization takes place at the system level during the design time. It deals with the geographical distribution of the processing centers, circuit manipulation, redesigning of architectures, instruction sets, transistor sizing, path balancing and factorization (Devadas and Malik, 1995). The main goal of the static power management is to keep the energy consumption or power consumption low by using low power usage components. In this category, the energy consumption is managed at two levels: CPU level and System level. It has been proven that among all the computing components, CPU consumes 35–50% and provides a big scope to optimize energy consumption (Valentini et al., 2013). At CPU level, the optimization could be done at register level or at instruction set level. At the register level, all measures to reduce the energy/power consumption are taken by optimizing the register transfer level (RTL) activities and at the instruction set level, different types of instruction set architectures (ISA) have been proposed to reduce the power consumption such as reduced bit-width ISA. Work has been done on instruction set optimization by various researchers to optimize the power consumption (Lee et al., 2013).

Along with CPU, there are other components who are also big contributors to the overall power consumption of the system such as memory components, network facility and software systems. System level static power management methods have been used to regulate the energy/power consumption by such components. System level power optimization also deals with setup techniques. Questions such as how to choose the right components during the setup phase of cloud systems to minimize the asynchronization between different components, how to place the servers to minimize the delays, choice of operating systems and application softwares are answered using system level power management methods. Architectures such as FAWN (Andersen et al., 2009) and Gordon (Caulfield et al., 2009) have been proposed to couple the low power CPUs with local flash storage and data centric powering systems to balance the computation and I/O activities to make the cloud computing architectures more performance and energy efficient. Geographic distribution of the machines (Tiwana et al., 2010), choosing components with maximum compatibility and network topologies to minimize the power consumption belongs to system level power optimization.

5.2. Dynamic power management mechanisms

Dynamic power management (DPM) deals with the regulation of energy consumption by using software based policies. Each type of server components provides a different dynamic power range such as the difference between the maximum power consumption and minimum power consumption. In the Fig. 10, it has shown that CPUs can consume around 30% of their peak power consumption in the low activity modes which gives the range of 70% to scale up and down. On the other hand, memory and disk drives have the dynamic range of 50% and 25%, respectively followed by the network facilities such as switches or routers, which have the range of only 15% (Beloglazov et al., 2011). On the basis of dynamic range of power consumption, the working of components can be scaled up or scaled down to regulate the power/energy consumption. On the basis of approach used to reduce the power/energy consumption, the classification of DPM methods is done in two levels, Hardware Level (using Power-scalable components) and Software Level (using Power-scalable resource management).

5.2.1. Dynamic power management using power-scalable components

At the component level, all the components supporting low activity modes are considered as the power scalable components such as CPU and can be manipulated using DPM methods. As stated earlier, CPU is the major power consuming component followed by the memory units. So in majority of cases, DPM methods are using two components such that CPU and memory for power/energy regulation.

Power Scalable CPUs use the relation between the power supply, operational frequency and voltage (Eq. (2)) to regulate the power utilization in processors (Fig. 10). Advancement in the processor architectures make CPUs able to run at different activity modes using different voltage and frequency rates.

$$P_{dynamic} = aCfV^2 \tag{2}$$

where a is the logical or switching activity, C is the capacitance, f is the operational frequency and V is the supply voltage. In complementary metal oxide semiconductor (CMOS) circuits, the energy consumption increases quadratically as the supply voltage increases. All the above mentioned power management techniques expolit this factor by reducing the supply voltage (DVS), operational frequency (DFS) or both at the same time (DVFS) (Le Sueur and Heiser, 2010; Zhu et al., 2004). There are many ways to scale down the high voltage supply to decrease the high energy consumption but one of the best is to exploit the stall time. Due to the speed gap between the main memory and the processor, significant amount of clock speed of processor has been wasted whilst waiting to get the required data from the main memory. During the waiting time (stall time), the processor frequency can be brought down by manipulating the supply voltage to the processor to save the excessive energy/power consumption (Kondo and Nakamura, 2005). Many semiconductor chip makers are using given voltage and frequency scaling techniques at different levels and in different devices. Intel's Woodcrest Xeon Processors works at eight different operating frequencies by reducing the maximum operational frequency by 8.3%, 16.5%, 25%, 33.3%, 41.6%, 50.0%, 58.3%, 66.6%, 77.7%, 88.9% and 100% (Gandhi et al., 2009). By using CPU throttling, Intel has developed SpeedStep CPU throttling technology and AMD has developed two CPU throttling technologies: CoolnQuiet and PowerNow!. Along with the frequency scaling of the CPUs, AMD has also implemented frequency throttling in Graphical Processing Units (GPU) as AMD PowerTune and AMD ZeroCore Power.

Power Scalable Storage Systems regulates the activity of storage devices such as disk drives to reduce power consumption. In the distributed computing systems, energy consumption by disk drives is significant. It has been estimated that around one-third of the total electricity supplied to the data centers is required for the mechanical operations of disk storage systems (Kim and Rotem, 2012). Typically, when a disk is in standby, it consumes about one tenth of the power that it consumes during the spinning mode. The energy consumption by storage systems in large data centers

[15] http://www.green500.org/greenlists

need to be considered seriously because the requirement of the storage systems is increasing by 60% annually (Pinheiro et al., 2006). In large cloud based data centers, disk drives usually remains underutilized and use less than 25% of their total storage capacity. This provides large scope to reduce the energy consumption by disk drives by increasing the utilization and by turning off the unnecessary disks (Gurumurthi et al., 2003). Various methods to make storage system power efficient are given in Fig. 9. A thorough survey on the energy efficiency of the disk drives has been done by Bostoen and Mullender (2013).

Power Scalable Memories are addressed least among all the components addressed to minimize the energy consumption in large scale distributed computing systems. According to David et al. (2011), under specific workloads, memory unit can consume 23% on average, of the total power consumption. In Fig. 10, the dynamic range of power consumption of memories is 50%, which provides plenty of scope to increase the power/energy efficiency of memory units. Like CPUs, the concept of low frequency and less voltage for power reduction (DVFS) is also applicable to memory units. In the case of DRAMs, the power consumption of some of the components such as storage arrays of DRAM can be scaled by V and some of the components can be scaled by V^2. Making energy aware memory components and using them in cloud computing environment gives rise to new challenges. Making power efficient memories will be achieved at the price of performance. The power aware techniques used in the memory units should be leveraged to save overall power consumption of the large scale systems without effecting the performance of the systems. In response to this, a software platform called Memory Management Infrastructure for Energy Reduction (Memory MISER) consists of a modified Linux kernel and implementation of a PID controller has proposed by Tolentino et al. (2007). The proposed architecture has been proved to reduce energy consumption of memories by up to 70% and up to 30% for the overall system.

5.2.2. *Dynamic power management using power-scalable resource management*

With the adoption of energy efficient components in the

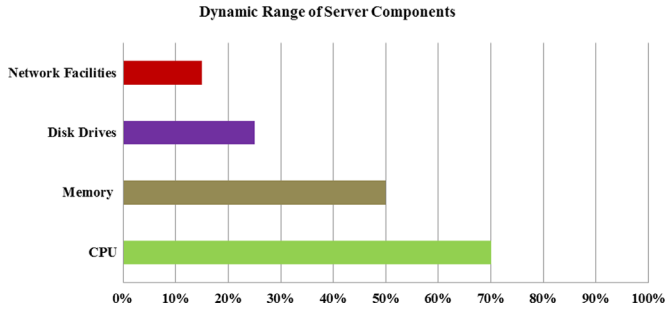

Fig. 10. Dynamic range of power consumption of various server components.

infrastructure of the cloud systems and, due to the vast amount of data for processing, the management and monitoring of the resources is very important. Wise management of the resources including resource provisioning, task scheduling, performance monitoring leads to less energy consumption and more profit-aware computing. Although the management of the resources is a general term for the distributed computing environment but in the context of cloud computing it is more associated with virtualization technology. The employment of the virtualization technology makes it possible to minimize the number of working resources by keeping the utilization of resources high by executing more virtual machines processing different workloads.

In this section, various mechanisms that execute the tasks on cloud computing infrastructure in an energy efficient manner will be highlighted. This section answers several questions such as how to provision the resources in an energy aware manner, How to distribute or schedule the workload among the provisioned computing components in an energy efficient way, When to migrate the running tasks from one underutilized resource to other to save the power consumption, When and how many computing components need to be turned on or turned off to save energy. In the literature, many algorithms, heuristics or architectures are proposed to handle the issues of power/energy consumption in cloud computing environments (Table 2). Mechanisms to reduce the energy consumption by using software techniques are divided into

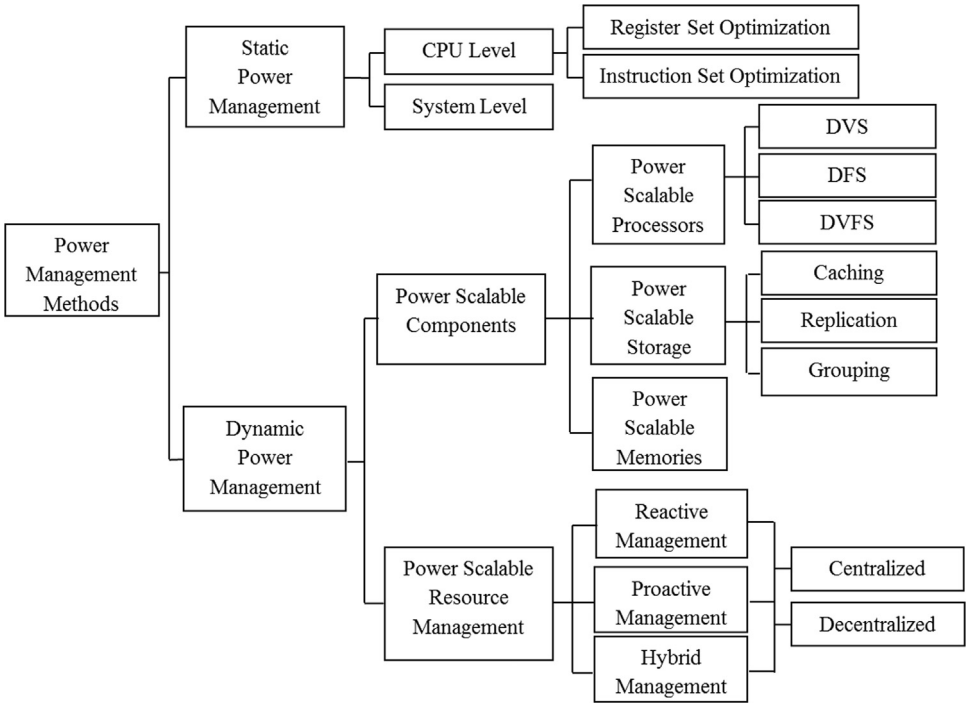

Fig. 9. Energy/power management methods.

Table 2
Survey of energy management mechanisms in cloud computing.

Authors	Power scalable resource management	Objectives	Workload	Components	Power saving method
Beloglazov et al. (2012)	Reactive	Resource Utilization	Web	CPU	DVFS
Garg et al. (2011)	Reactive	Carbon Footprint, Profit	HPC	CPU	DVFS
Burge et al. (2007)	Reactive, Proactive	Profit	Multiple	System	On/Off
Chen et al. (2005)	Reactive, Proactive Hybrid	Profit	Web	CPU	DVS
Lee and Zomaya (2012)	Reactive	Resource Utilization	Multiple	System	
Bradley et al. (2003)	Proactive		Web, Email Database	CPU	On/Off
Wang et al. (2014)	Proactive	Resource Utilization	Multiple	System	On/Off
Ghorbani et al. (2014)	Proactive		Multiple	CPU, Memory	
Subirats and Guitart (2015)	Proactive	Resource Utilization	Batch, Web		
Srikantaiah et al. (2008)	Reactive	Resource Utilization	Multiple	CPU, Disk	
Kord and Haghighi (2013)	Proactive	Resource Utilization, Profit	MapReduce	System	
Gandhi et al. (2009)	Proactive	Resource Utilization	Web	CPU, Memory	DFS, DVFS
Egwutuoha et al. (2013)	Proactive	Resource Utilization	HPC	CPU	
Le and Wright (2015)	Reactive	Carbon Footprint, Profit	HPC	CPU	
Tesfatsion et al. (2014)	Reactive	Carbon Footprint	Video Encoding	CPU	VM Migration, DFS
Wadhwa and Verma (2014)	Reactive	Resource Utilization, Carbon Footprint	Multiple	CPU	DVFS
Khosravi et al. (2013)	Reactive	Carbon Footprint	BoT, Web	System	VM Placement
Lefèvre and Orgerie (2010)	Proactive	Carbon Footprint, Profit	Multiple	System	On/Off
Garg et al. (2011)	Reactive	Carbon Footprint	HPC	CPU	
Mezmaz et al. (2011)	Hybrid	Resource Utilization	HPC	CPU	DVS
Gandhi et al. (2011)	Hybrid	Resource Utilization	Web	System	
Subrata et al. (2010)	Hybrid	Resource Utilization	Web	System	Hibernation
Salfner et al. (2010)	Hybrid	Carbon Footprint	Multiple	CPU	On/Off

three different categories: Reactive, Proactive and Hybrid (Hameed et al., 2014). These can be implemented in centralized and decentralized ways.

Reactive management of resources takes all the measures to manage the energy consumption according to the current state of the system. The reactive mechanisms are based on feedback or monitoring. The continuous monitoring of the system is done and according to the pre-defined constraints such as thresholds the corrective actions are taken by migrating or consolidating the workload to regulate the energy consumption of the system. The productivity of the reactive management of energy consumption depends upon the accuracy of the monitoring procedure. In virtualized computing environments like clouds, when the resources are not fully utilized the migration or consolidation of the running virtual machines to some other resource is possible and promised as the best technique to reduce the energy consumption. Along with regulating the energy consumption, with the efficient utilization of the resources, the carbon emission rate is also a concern. As the energy consumption will be increasing, the temperature will rise and more power will be required to run the cooling infrastructure to keep the temperature low. For the generation of each unit of electricity, fuel has to burn that adds to the carbon emission. This is also a main factor that is under consideration in these days. United Nations and governments of various countries like Japan[16] are imposing the penalties and developing protocols such as UNs Kyoto Protocol[17] to reduce the carbon footprints by the cloud based data centers. A thorough survey of work on the energy consumption by using the reactive resource management mechanisms is given in Table 2.

Proactive Management for Resources also known as predictive management of resources use the information about the average behavior of the system rather than the current state of the system. The decision about choosing the optimized resources in terms of performance, energy consumption and reliability has been taken on the basis of the data collected during the previous runs. By using the collected data, predictions are done about the behavior of the system to make the adequate decisions about the allocation of resources to minimize the energy consumption. In the literature various prediction models are proposed to minimize the energy consumption (Bradley et al., 2003). Similar methods using predictive approach to reduce energy utilization in cloud computing environment is present in Table 2.

Hybrid Management of Resources use both predictive behavior of proactive methods and monitoring behavior of reactive methods to tune the energy consumption and resource utilization. Due to the dependency on the results of the prediction mechanisms, methods in proactive resource management always lags because as mentioned in Section 4.1, it is hard to predict the behavior of the system accurately including the energy consumption as well. On the other hand, due to the large overhead, the reactive energy efficiency resource management methods possess delays, which add to the power inefficiency of the whole system. By combining the merits of reactive and proactive methods, the hybrid methods have been designed. In the literature (Table 2), some work has been done by various authors combining both reactive and proactive methods to reduce the energy consumption of the CCS.

6. Trade-off between reliability and energy efficiency in cloud computing

We have observed in previous sections that most of the research has focused either on service reliability or energy efficiency in cloud computing environments. As analyzed, existing mechanisms do provide reliability to cloud computing services and have proved to be very efficient and optimized (L'Ecuyer and Malenfant, 1988; Clark et al., 2005). By using these methods, cloud computing service providers are claiming on the one hand that their cloud services are more than 99% available in terms of uptime allowing only 80 h of downtime per year, approximately. However, all the given methods require extra back-up and storage resources to store logs and checkpoints to allow last state system recovery in the case of failure or interruption. Adding extra resources to the infrastructure increases the energy consumption at a greater rate than reliability gains and has a direct impact on the profit margins

[16] http://www.jdcc.or.jp/english/
[17] http://unfccc.int/resource/docs/convkp/kpeng.pdf

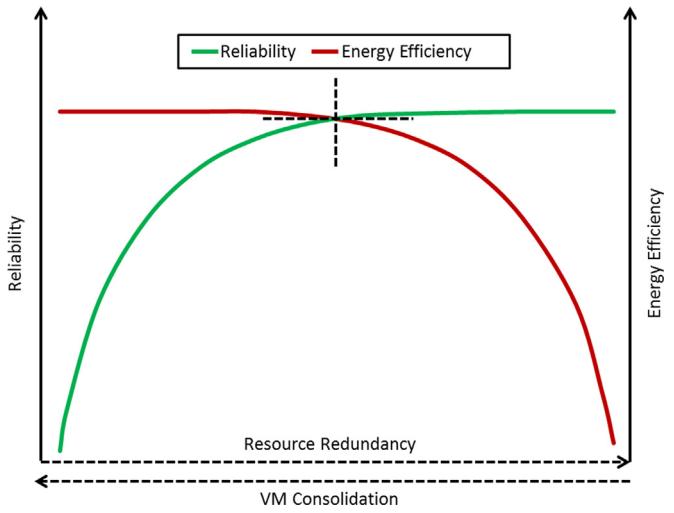

Fig. 11. Reliability and energy efficiency tradeoff in cloud computing systems.

of the service providers and users and negatively impacts natural environment.

Energy management mechanisms that regulate system performance and hardware resources reduce the system energy consumption. The key techniques used to reduce energy consumption is running the resources on low power scaling level or by turning off the idle resources such as back-up, which will reduce the reliability of the system. For example, in the case of virtual machine consolidation (key technique to reduce energy consumption in CCS), if a physical machine fails due to some hardware or software issues before the completion of tasks and there are no recovery resources, then all the virtual machines and their corresponding processes will have to start again. This will dramatically increase overheads such as energy consumption and resource utilization. Service providers will lose a lot of revenue in terms of penalties for SLA violations and most importantly, trust of the users.

In Fig. 11, a crucial trade-off between reliability and energy-efficiency of CCS can be clearly seen. On the one hand, reliability of the system increases as the resource redundancy increases. But increasing the number of redundant resources used to store back-ups or to run replicas has adverse effect on the energy efficiency of CSS. On the other hand, as the frequency of virtual machine consolidation increases, energy efficiency of the system increases. But high VM consolidation has the negative effect on the reliability of the system. Both reliability and energy efficiency of CCS increases asymmetrically. This trade-off opens up new opportunities and challenges in CCS by considering both these elements simultaneously. It is very important to reach equilibrium between these two metrics from different perspectives such as quality of services, revenue, operational cost and environment. There is a distinct need for more research in the area of optimizing the relationship of system reliability and energy efficiency in CCS (Table 3). The following section of this paper seeks to outline the current research into the interplay of reliability and energy efficiency in CCS.

6.1. State of the art in reliability and energy efficiency mechanisms in cloud computing

In this section, current research combining reliability and energy efficiency of cloud computing has highlighted and gaps have been identified. The brief description of this section has provided in Table 3.

Faragardi et al. (2013) have proposed an Integer Linear Programming (ILP) based mathematical model to regulate the reliability and energy consumption of the CCS by taking into

Table 3
Survey of trade-off between reliability and energy management in cloud computing.

Authors	Failure management method	Failure management type	Energy management method	Energy management type	Failure type	Performance evaluation method	Application model
Faragardi et al. (2013)	Imperialist Competitive Algorithm based Failure Aware Resource Allocation	Proactive	On/Off	Proactive	Independent Failures	Mathematical Model, Simulation	Independent Tasks
Egwutuoha et al. (2013)	Failure Prediction based Process Migration	Proactive	Process Migration	Proactive	Independent Failures	Simulation	Message Passing Interface(MPI) Applications
Sampaio and Barbosa (2014)	Failure Prediction based VM Migration	Proactive	VM Consolidation On/Off	Proactive	Independent Failures	Simulation, Real Platform Evaluation	Poission Distribution and Google based Workloads
el Mehdi Diouri et al. (2012)	Checkpointing, Uncoordinated Message Logging, Process Coordination	Reactive	Voltage Scaling	Reactive	Independent Failures	Benchmarking	High Performance Computing Applications
Zhang et al. (2015)	Reliable Resource Allocation	Reactive	Dynamic Voltage and Frequency Scaling (DVS)	Reactive	Independent Failures	Simulation, Benchmarking	Fast Fourier Transformation, LU-decomposition, Gaussian Elimination
Deng et al. (2012)	Genetic Algorithm based Server Cosolidation	Proactive	On/Off	Proactive	Independent Failures	Mathematical Model, Simulation	Light, Normal, Intensive Application Workloads
Lin et al. (2013)	Task Re-execution Policy	Reactive	Load Balancing	Reactive	Independent Failures	Theoretical Simulation	MapReduce Workload

consideration quality of services in terms of service deadlines. On the basis of this model, a swarm intelligence resource scheduling method based on Imperialist Competitive Algorithm (Atashpaz-Gargari and Lucas, 2007) has proposed to allocate the resources in a failure-aware and energy-efficient way. To introduce the failures in the systems, Faragardi has used a Poisson process-based failure model that generates constant and independent failures. Along with the failure model, an energy model has also been proposed based on CPU utilization. By using the equations for reliability and energy consumption, a common ILP-based cost function has been used to balance both energy and reliability. The proposed solution has improved the energy utilization and system reliability significantly by 17% and 9% respectively in comparison to a hybrid genetic algorithm.

In this study the occurrence of failures has been modeled by using Poisson distribution, which has proved to be a poor fit by many researchers (Plank and Elwasif, 1998; Schroeder and Gibson, 2010). Normal and log-normal distributions have proved a better fit for failure generation modeling. Authors have also been modeled independent occurrence of failures, which have been challenged (Rangarajan et al., 1998; Gallet et al., 2010) by showing temporal and spatial correlation between the failures.

Egwutuoha et al. (2013). have developed a generic proactive energy efficient fault tolerance model independent from redundant resources for CCS executing high performance computing (HPC) applications. To provide immunity from task failures, a rule based prediction mechanism has been used to foresee failures using the data gathered by a back end service “FTDeamon” using LM-sensors. A mathematical model has been developed to evaluate the weight of the current state by multiplying the LM-sensor values of all components of systems. After calculating the current weight, a comparison has been done with the critical state threshold value. On the basis of comparison result, decisions about provisioning of new resources, relinquishing of faulty ones and migration of processes has been taken. To make the method less expensive and energy efficient, no extra resources are provisioned initially to provide fault tolerance. On the basis of the results of failure prediction mechanisms, extra resources are provisioned to initialize the virtual machines to migrate the running processes from failing hosts/resources. Process level migration has been used instead of using traditional VM level migration because process level migration has less overheads and makes the migration fast, which further helps to reduce overall energy consumption and make fault tolerance more dynamic and less complex.

The proposed mechanism has been designed for message passing interface applications, which require uninterruptable functioning of resources for a long duration. As no backup resources are used to provide immunity from failures to running processes, this algorithm depends highly upon the accuracy of the failure prediction mechanism. The average accuracy of failure prediction mechanisms is 76.5% (Fu, 2010). This level of accuracy is unsuitable for HPC workload. To make the mechanism more attractive, both reactive and proactive fault tolerance mechanisms should be used simultaneously.

Sampaio and Barbosa (2014) have proposed two algorithms POwer-and Failure-Aware Relaxed time Execution (POFARE) and POwer-and Failure-Aware Minimum time Execution (POFAME). They address the problem of mapping of virtual machines to physical machines, so as to increase the completion rate of the tasks with minimum energy consumption in a private cloud computing environment. Stop and copy VM migration employing failure prediction has been used to make the services available and to execute the tasks by deadline without any interruption. CPU has chosen optimistically on the basis of predicted Mean Time between Failure (MTBF) and according to the capacity required to finish the tasks within their respective deadline. SLA terms are ensured by completing the tasks on time and avoiding penalties. A tentative private cloud architecture has also been designed in which a cloud manager monitors the status of virtual and physical machines. Based on the information, the cloud manager allocates tasks concerning energy consumption improvement, so as to facilitate physical machine fault tolerance. To save energy and provide fault tolerance, virtual machine consolidation or migration has been employed as well as putting free physical machines in sleep mode. Three other algorithms: Common Best-Fit (CBFIT), Optimistic Best-Fit (OBFIT) and Pessimistic Best-Fit (PBFIT) are used to evaluate the performance of proposed algorithms. After the intensive simulations performed by using Poisson distribution-based random workload and Google-based workload, POFARE outperformed all the algorithms and gives the best results.

A limitation of Altino M. Sampaio's energy model lies in the use of only CPU power consumption, without consideration of any other components such as memory and disk-drives and heterogeneity of physical machines. Voltage scaling would have been more energy efficient solution than entering and waking-up the nodes from the sleep state. Similar to Egwutuoha et al. (2013), performance degradation of the system has not been considered. To make the reliability and failure models simple, most of the researchers assume either the system works fine or it fails. This kind of binary behavior is valid for components such as CPU but not for the whole system because in virtualized computing environments, system slowdown or performance degradation occurs because of shared resources between virtual machines. This leads to the system failure. In the given work, running multiple virtual machines on the same node has been implemented but assumes no performance interference, which is not the case in the real world and has to be considered. Failure tolerance in proposed solutions relies completely on failure prediction. If physical machine fails outside of the failure forecasts, then all the virtual machines have to be re-initiated because of a non-forecasted failure then all the running virtual machines have to be re-initiated.

Garraghan et al. (2014) have done an empirical analysis by using google traces to analyse the failure related energy waste in cloud computing environments. This analysis highlights the impact of failures at task level (software level) and server level (hardware failures). All the terminal events taken from Google cluster traces are divided into three categories: Kill, Evict and Task fail. SpecPower2008 benchmark (Benchmarks, 2000) has been used to calculate the energy consumption of per failure event. In the study, it has been noted that Kill and Evict contributes to more energy wastage (48% and 39% respectively) than task failures (13%). The occurrence of kill and evict events have been considered because of scheduling, which is one of the reasons of failures in cloud computing services (Fig. 2). All the tasks are assigned with different priorities from 0 to 9 and occurrence of failures has been scaled on the basis of priorities of the interrupted tasks. The low priority tasks have been found most prone to failures with mean time between failure of only 1 h and vice versa for high priority tasks (48 h and 58 h respectively). 35% of failures occur on tasks with lower priority. At the server level, numbers of failures are calculated on the basis of the architecture type of the underlying servers. The frequency of the occurrence of failures (MTBF) and recovery time (MTTR) is independent of the population of the servers. For energy wastage, priority 0 tasks have only minor energy wastage but priority 8 and 9 tasks waste large amount of energy in comparison to the number of failures because of resubmissions. This means that the energy wastage is independent from the number of task failures. The proportion of energy wastage depends upon the characteristics of the failed tasks such as the task length. The longest running tasks (priority 9) have the greater impact, which wastes a considerable portion of the energy

(up to 65%). From the analysis of all type of terminal events, task failures contribute upto 21% of the total energy wastage because of the resubmission and recomputation of failed tasks.

In conclusion, it has been claimed that the choice of a mechanism to regulate the energy wastage in the presence of failures should be made by considering physical architecture or scenario. Inappropriate mechanism will lead to more energy wastage rather than reduction, for example the adoption of task migration for low priority tasks will lead to high increase in execution time which further increase the energy consumption. Garragham's work is based only on empirical analysis but has not proposed any mathematical model or formal procedure to regulate the energy consumption in the presence of failures.

el Mehdi Diouri et al. (2012) have evaluated the energy consumption by checkpointing and fault tolerance (coordinated and uncoordinated) protocols. In uncoordinated protocols, logs are stored at Hard Disk Drive (HDD) or Random Access Memory (RAM) for message logging. When comparison has been made between the power consumption of RAM logging and HDD logging, power consumption by RAM logging has been found to be less than the consumption of HDD logging. So it has been concluded that to provide fault tolerance in extreme scale distributed computing systems, message logging protocol using RAM to store logs should be preferred over the HDD based message logging and checkpointing. For coordinated protocols, energy consumption patterns are similar to the patterns seen in uncoordinated protocols and checkpointing. The energy consumption by coordinated protocols depends upon the duration of the coordination process, which further depends upon process. Poor synchronization means a longer coordination process and more power consumption. By slowing down the fastest process, extra energy consumption can be minimized.

To make the decision about choosing suitable energy aware fault tolerance methods, an evaluation of coordinated (3 coordinate) and uncoordinated (RAM logging) fault tolerance protocols has been done using 4 NAS parallel benchmarks with 16 and 64 processes. All experiments are conducted on Lyon site of Grid5000 (Cappello et al., 2006) using their energy measuring infrastructure facility. It has been concluded that message logging protocols are more suitable for the applications involving less data exchange and vice versa for coordinated methods.

Zhang et al. (2015) have addressed an optimization problem to maximize the reliability with energy conservation for precedence constraint tasks in heterogeneous clusters by proposing three algorithms. They are: Reliability-aware Heterogeneous Earliest Finish Time (RHFT), Reliability-aware Critical-Path-On-a-Processor (RCPOP) and Reliability Maximization with Energy Constraint (RMEC) algorithm. All the proposed algorithms have three phases: task priority establishment, processor frequency selection and task to processor mapping. In task priority establishment phase, all the tasks are prioritized according to their URank, which is a method to calculate the topological order for directed acyclic graphs (DAG). After calculating the URanks (bottom up approach), all the tasks are pushed in priority queue in decreasing order starting with highest priority. Once tasks are ordered, best frequency and voltage pair are chosen. This consumes less energy in executing tasks ready at the top of the queue. Along with the proposed algorithms, Hierarchical Reliability Driven Scheduling (HRDS) and Reliable Dynamic Level Scheduling (RDLS) algorithms are also used for a comparative evaluation.

To evaluate the performance of given scheduling algorithms a large number of randomly generated DAG with different number of nodes (tasks) and real-world applications are used. For real world applications, three problems i.e. Fast-Fourier Transformation (FFT), LU decomposition and Gaussian elimination are chosen to generate task graphs. The simulation results show that in all cases, RMEC outperforms other algorithms in terms of reliability and energy consumption. Though the proposed algorithms have worked well in the present scenarios, results may vary in the presence of correlated failures. So, to make the solutions more promising and applicable, consideration of models for correlated failures (Yigitbasi et al., 2010; Gallet et al., 2010) should be taken into account.

Deng et al. (2012) have proposed a Reliability-Aware server Consolidation stratEgy (RACE) to address a multi-objective problem with reliability and energy cost factors. A utility model that can estimate the cost of server consolidation in terms of reliability and energy efficiency whilst still mitigating SLA violations occurring due to resource demand and supply mismatch has been formulated. The unified utility model has been used by a genetic algorithm improved grouping genetic algorithm (IG2CA) to provide an optimized solution of the problem by choosing the best among the initial configurations provided by the proposed reliability-aware resource buffering and VM to PM mapping heuristics.

To prove the superiority of the proposed RACE server consolidation strategy, a simulation based analysis has been done by using light, normal and heavy application workloads. The results of the simulation have compared with results of two other server consolidation strategies: pMapper (Verma et al., 2008) and PADD (Lim et al., 2009). With the increase of incoming workload, the occurrence of SLA violations tend to increase due to the fluctuation in the workload and resource shortage. In the proposed method, the value of utility function has been assessed before accommodating any request and performing VM consolidation. If the value of utility function is positive, only then consolidation will be considered valid. Because of the common utility function has unified SLA violation, energy costs and reliability, the proposed strategy has outperformed all other methods. This kind of constraint is not available for other consolidation strategies and they tend to accept all the requests which lead to more SLA violations and energy saved by them has outweighed because of penalties of SLA violations.

Lin et al. (2013) have studied the job completion reliability (JCR) and job energy consumption (JEC) for general map reduce infrastructure (GMI). The probabilistic models for worst case and best case have been formulated to represent the reliability of slave nodes performing map and reduce tasks and master nodes running job tracker and name node instances. The best case corresponds to the execution of job without any interruption and worst case corresponds to the execution of job on every cold-standby node (redundant nodes) at the slave end. Along with the formulation of reliability of master and slave nodes to finish a job, corresponding energy consumption has also been formulated as the function of time taken to finish. All the nodes at master end and at slave end are homogeneous and occurrence of failures has been assumed following Poisson distribution. The influence of different number of cold-standby slave nodes (varies from 1 to 4 in this study) has been evaluated on job completion reliability and job energy consumption. 10 jobs with different lengths of map task execution time are considered and each of them are divided into 4096 map tasks and 1 reduce task. It has been seen that increasing the number of cold-standby nodes from 1 to 2 increases the JCR but further increase does not make any difference because of the absence of any redundancy measure at the master end. This means that increasing the number of backup resources does not increase reliability as long as no measure has been taken at the master node end. For the best case scenario, energy consumption is less and linear with respect to map task execution time and independent from the number of cold-standby nodes such that energy consumption remains same for all the number of backup nodes. For the worst case, energy consumption is linear with respect to map-task execution time but varies according to the number of cold-standby nodes. When the

best case occurs, increase in the number of cold standby nodes does not affect JEC of GMI.

After the analysis, it has been concluded that General Mapreduce Infrastructure (GMI) is energy efficient but for long executing jobs, it is not reliable because of the absence of redundancy measure at the master end. We propose to improve the reliability of the system by using redundancy measures at the master end of the system.

6.2. New challenges and future research directions

Many solutions have been developed either to increase the reliability of the system (Table 1) or decrease the energy consumption of the system (Table 2). Some of the work done jointly in the field of reliability and energy efficiency of CCS is highlighted in the Table 3. To the best of our knowledge, this list includes all the research to make the CCS both reliable and energy efficient at the same time. Finding a solution to achieve both objectives at the same time poses new fundamental challenges, which are discussed in the following sections.

Impact of Energy Efficiency Techniques on Reliability: though a lot of work has done to optimize the energy management (Table 2) by exploiting power regulation techniques in order to make CCS energy efficient but reliability of cloud systems has left an open challenge to look at along with energy efficiency of the systems. To make the CCS energy efficient, all the energy-aware resource management techniques are usually based on the manipulation of underlying resources which can be done either by running the resources at low-scaling mode or by turning them off. Though these methods have proved very efficient from the perspective of energy management, they have adverse effect on the reliability of the systems. Switching the resources between low scaling modes and high scaling modes using frequency and voltage scaling techniques (DVFS) causes an increase in the response time and decrease in the overall throughput of the system. This can result in a service delay and be considered as a service failure due to SLA violations. On the other hand, turning servers on/off or putting resources in sleep mode more frequently makes them more failure prone than running the resources all the time. Just as the lifetime of a car brake-pads decrease with each slowdown, the reliability of server components, specifically disk drives, also decrease with each power modulation. Thats why many disk manufacturers limited the start/stop power cycles of disk drives to 50,000 for their entire lifetime and also propose to keep the power cycles limited to 10 times/day to keep the overall system reliability high (Zhu and Zhou, 2005). So the optimal solution is to make the CCS energy efficient and reliable at the same time and thus help to make the paradigm stable and acceptable.

Impact of Failures on Energy Consumption: many solutions are provided in the literature (Table 3) to evaluate the impact of utilization, energy consumption etc on the occurrence of failures but how much energy consumption of the system will be effected with the occurrence of failures remains unclear. It is necessary to use optimized fault tolerance methods to reduce the occurrence of failures in CCS. But to make the current fault tolerance methods more optimized in terms of energy consumption, it is important to study the relation between the failures and energy consumption. Defining this relationship will help to simultaneously increase the reliability and energy efficiency of CCS.

Multi-Objective Resource Provisioning Methods and Techniques: most existing research has on either the reliability or the energy efficiency aspect of cloud task scheduling (Table 3). The resource or task scheduling can be formulated by using different optimization problems such as bin packing problem in which the available resources are assigned to the incoming tasks according to certain conditions. The resources provisioning is like a bin-making problem such that adequate number of resources need be reserved first and, after the reservation, bin-packing solutions can be used to do the optimization. In the case of under-provisioning of resources, the scheduler will not get enough resources to schedule the tasks which can lead to the service failure. On the other hand, in the case of overprovisioning, reserved resources will remain underutilized which will increase the cost of service in terms of energy consumption and other operation expenses. Rather than considering the resource and task mapping problem as a single-layer problem, it is better to consider it as a two-layer problem consisting of resource provisioning and resource scheduling. For each layer, different solutions need to be proposed to make the CCS both reliable and energy efficient.

Prediction Algorithms to Estimate both Fault Occurrence and Energy Consumption: if the occurrence of the failure or fault in the system is predictable, then important measures can be taken before the occurrence such that the checkpoints can be saved with less overhead, running virtual machines or tasks can be migrated to more reliable physical machines. By doing this, we can save unnecessary wastage of power/energy that will be required to restart all the running process that were interrupted during the failure. The prediction will help to adopt the reactive and proactive failure management and energy management mechanisms wisely. Suppose the occurrence of failure can be known in advance, then the checkpointing or logging of the current state of the system will start just before the occurrence of the failure. Therefore, we can reduce overhead occurred due to the checkpoints or logs of the running system. If the overhead will be reduced then less number of backup resources will be required and energy consumption of the system will be reduced without compromising the reliability of the system.

Federated Clouds and their Standardization: interconnected clouds or Federated clouds is the collection of clouds analogus to the Internet (collection of networks). Giacobbe et al. (2015) have defined the cloud federation as an ecosystem of different cloud providers that are interconnected in a cooperative decentralized computing environment. With the inter-cloud computing the reliability and energy efficiency of the cloud services will be increased by making them more dynamic and scalable. Being in the early stage, cloud computing is lacking in standardization. As the reference models and standards are available for other deployments such as the Internet, cloud deployment has not yet have any confirmable reference models and standards. As a result, most of the cloud providers have designed their own proprietary standards and interfaces. To avail the services of such clouds, applications need to get tailored according to the specific standards and interfaces. This gives rise to another problem called vendor lock-in Toosi et al. (2014). The existence of the reference models (TCP/IP model in case of Internet) and standards for cloud computing paradigm will help the developers to implement the generic solutions following the similar attributes. The standardization will also help to regulate the energy consumption of cloud infrastructure by making the migration of running virtual machines easy from one cloud vendor to another, which is yet only been done between the resources of same or different sites of the same cloud provider. With the proper set of standards or rules, the concept of inter-cloud computing will be realized more efficiently, which will make the cloud technology more reliable, affordable and eco-friendly. It is observed that the majority of time, the resources of data centers providing cloud services remains underutilized but still the providers keep extending and upgrading their infrastructure to house the future needs for example Microsoft is adding 10,000 servers per month to its data centers.[18] With the

[18] http://www.datacenterknowledge.com/archives/2008/08/14/218000-servers-in-microsoft-data-centers/

proper realization of inter-cloud computing architectures, this over spending can be avoided by sharing the resources between the different cloud providers to serve the unexpected service requests in a reliable manner without violating the service level agreements.

Real Cloud Failure Traces: although at the physical level cloud computing services are deployed at the infrastructure of the clusters or other distributed computing systems, the working paradigm for the CCS is different from the rest of the distributed computing architectures. In most of the research literature, the empirical or statistical analysis about failures and energy consumption of the CCS has been done by using traces or log files of grids or clusters mounted within cloud computing services. For example, Garraghan et al. (2014) have done an empirical analysis to evaluate the effect of failures on the energy wastage of the cloud systems. The whole work was done by using Google traces generated during the occurrence of failures in Googles clusters. The occurrence of the failures was deduced according to the behavior or changes in the log data because no information has been shared regarding the occurrence of these failures. Although, there are different types of failure traces present such as Failure Trace Archive (Kondo et al., 2010), Google cluster traces, Computer Failure Data Repository (CFDR)[19] for different types of distributed computing architectures such as grids, clusters, volunteer computers etc, there are no failure traces present for a real cloud. A big gap exists in the analytical studies of cloud behavior done by using non-cloud based traces or logs that specifically trace failures and energy consumption. To make the research more attractive, the cloud computing service providers must disclose the real cloud traces for the occurred failures and energy consumption and must build public repositories or help the researchers to do so.

6.3. Reliable and energy-efficient cloud computing architecture: a conceptual model

To resolve aforementioned issues, there is a need for optimized energy-aware and failure-aware resource provisioning policies, which is the focus of our research. To realize these policies, a cloud computing architecture is required. Fig. 12 depicts an extended version of a layered view of our tentative cloud architecture that was proposed earlier (Sharma et al., 2015) which incorporates reliability-aware and energy-aware resource provisioning policies.

Cloud Service Users/Brokers: cloud service users or brokers providing services to other users reside in this layer. Users submit requests and attains services according to the terms and conditions of service level agreements (SLA).

Cloud Management Center (CMC): this layer is the heart of whole architecture on which our research is focused. All the management decisions about providing the services will be made here. This layer includes Business, Provisioning and Monitoring components.

1. **Business:** this part is used to manage the expenses of a CCS. Challenges like billing of services, cost of services, cost of ownership etc. will be handled by the solutions provided by this module.
2. **Monitoring:** the monitoring section will help to make decisions for other layers by providing them feedback. The main job of this section is to monitor the activity of the under-lying infrastructure so as to ensure uninterruptable services. The solutions provided are also responsible to monitor the activities of users such as, their requirements and operations.
3. **Reliable and energy-efficient resource provisioning:** this module is responsible for the cloud resource provisioning to customers in a reliability-aware and energy-aware manner. All the decisions regarding the optimization of cloud services will be taken here. This module will provide solutions such as, energy management, virtual machine management, SLA management and fault management. The main focus of our research deals with reliability-aware and energy-aware resource provisioning policy that we will incorporate in this layer.
   - *SLA Management:* includes SLA contract definition and utilization of SLA schemas with associated QoS parameters, SLA monitoring and reliability and energy efficiency policies.
   - *Fault Management:* keeps track of systems and other faults and uses this information to statistically compute future potential failures, and the mechanisms and processes to mitigate the likelihood of such errors and their impact.
   - *Energy Management*: includes the energy management mechanisms that will be responsible to regulate the energy consumption of the under lying hardware resources by lowering the operating frequency or turning them off according to the current utilization or workload.
   - *VM Management:* monitors the availability of VMs and provides migration/replication services on behalf of the cloud provider on the basis of our proposed common cost function for reliability and energy efficiency. The value of the common cost function will be calculated using the outputs of the energy management module and fault management module. A key part of the work to be undertaken is to support live migration of VMs from active physical machines to passive physical machines and to preemptively deal with failures seamlessly and transparently from the cloud customer perspective, and so to provide undisrupted cloud services.
4. **Virtual Layer**: on the basis of procedures and policies implemented at Cloud Management Center layer, virtual machines providing services to users will run on the top of physical architecture. Virtual machine migration or consolidation to ensure fault tolerance and energy efficiency will take place at this layer according to the results of resource or service management algorithms or policies executing at cloud management center (upper layer),
5. **Physical Infrastructure**: this layer deals with actual hardware infrastructure upon which the cloud computing services rely. It consists of different types of physical machines such that low utilized passive physical machines and active physical machines that are providing services to the users.

7. Conclusion

Although cloud computing platforms are widely used today, there are still plenty of research gaps to be addressed. Due to the large infrastructure of clouds, energy efficiency, reliability and scalability are among the foremost concerns in cloud computing. In this paper, we have explored various types of failures that drive researchers to design the mechanisms to make the CCS highly reliable. This paper has surveyed and critiqued a variety of methods aimed at increasing the reliability of CCS. The increase in the size and design complexity of clouds, is resulting in huge energy consumption and enormous carbon footprints. This paper also presented a comprehensive survey of all the energy management techniques used in CCS. We observed that the adoption of mechanisms to provide reliability in cloud computing services has impacted the energy consumption of the system. Adding back-up resources, running replicated systems, storing logs etc. provide strong fault tolerance but also increase the energy consumption. There is a critical trade-off between service reliability and energy consumption that urgently needs to be investigated. We have

[19] https://www.usenix.org/cfdr

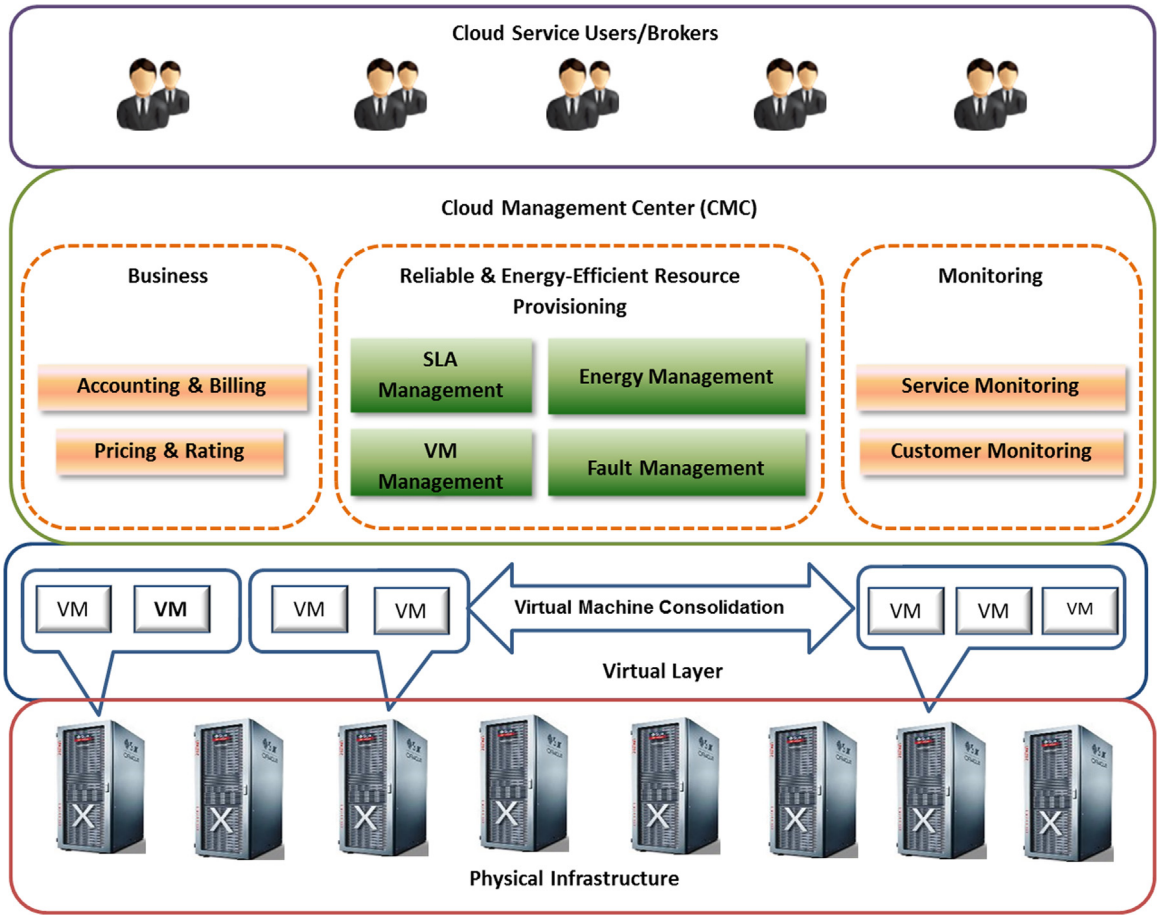

Fig. 12. Reliable and Energy-Efficient Cloud Computing Architecture.

identified the need for a reliability-aware and an energy-aware resource provisioning policy to improve the availability of cloud services whilst simultaneously reducing its energy consumption.

Acknowledgments

The authors would like to show their gratitude to Rodrigo N. Calheiros from The University of Melbourne, David Di Lenno and Ramesh K. Dixit for sharing their constructive comments and suggestions on improving the survey. Authors are also thankful to two anonymous reviewers for their comments that greatly improved the manuscript.

References

Abraham, S., Chengalur-Smith, I., 2010. An overview of social engineering malware: trends, tactics, and implications. Technol. Soc. 32 (3), 183–196.
AlZain, M., Pardede, E., Soh, B., Thom, J., et al., 2012. Cloud computing security: from single to multi-clouds. In: Proceedings of the 45th Hawaii International Conference on System Science (HICSS). IEEE, Maui, HI, USA. pp. 5490–5499.
Andersen, D.G., Franklin, J., Kaminsky, M., Phanishayee, A., Tan, L., Vasudevan, V., 2009. FAWN: a fast array of wimpy nodes. In: Proceedings of the 22nd ACM Symposium on Operating Systems Principles. ACM, Big Sky, MT, USA. pp. 1–14.
Atashpaz-Gargari, E., Lucas, C., 2007. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. In: Proceedings of the Congress on Evolutionary Computation (CEC). IEEE, Singapore. pp. 4661–4667.
Bala, A., Chana, I., 2015. Autonomic fault tolerant scheduling approach for scientific workflows in cloud computing. Concurr. Eng. 23 (1), 27–39.
Barroso, L.A., Clidaras, J., Hölzle, U., 2013. The datacenter as a computer: an introduction to the design of warehouse-scale machines. Synth. Lect. Comput. Archit. 8 (3), 1–154.
Beloglazov, A., Buyya, R., Lee, Y.C., Zomaya, A., et al., 2011. A taxonomy and survey of energy-efficient data centers and cloud computing systems. Adv. Comput. 82 (2), 47–111.
Beloglazov, A., Abawajy, J., Buyya, R., 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Gener. Comput. Syst. 28 (5), 755–768.
Benchmarks, S., 2000. Standard Performance Evaluation Corporation.
Bonvin, N., Papaioannou, T.G., Aberer, K., 2010. A self-organized, fault-tolerant and scalable replication scheme for cloud storage. In: Proceedings of the 1st ACM symposium on Cloud computing. ACM, Indianapolis, IN, USA. pp. 205–216.
Bostoen, T., Mullender, S., Berbers, Y., 2013. Power-reduction techniques for data-center storage systems. ACM Comput. Surv. (CSUR) 45 (3), 33.
Bradley, D.J., Harper, R.E., Hunter, S.W., 2003. Workload-based power management for parallel computer systems. IBM J. Res. Dev. 47 (5.6), 703–718.
Burge, J., Ranganathan, P., Wiener, J.L., 2007. Cost-aware scheduling for heterogeneous enterprise machines (cash'em). In: Proceedings of the International Conference on Cluster Computing, IEEE, Austin, TX, USA. pp. 481–487.
Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I., 2009. Cloud computing and emerging it platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst. 25 (6), 599–616.
Cappello, F., Desprez, F., Daydé, M., Jeannot, E., Jegou, Y., Lanteri, S., Melab, N., Namyst, R., Primet, P.V.-B., Richard, O., et al., 2006. Grid5000: a nation wide experimental grid testbed. Int. J. High Perform. Comput. Appl. 20 (4), 481–494.
Caulfield, A.M., Grupp, L.M., Swanson, S., 2009. Gordon: using flash memory to build fast, power-efficient clusters for data-intensive applications. ACM Sigplan Not. 44 (3), 217–228.
Chen, Y., Das, A., Qin, W., Sivasubramaniam, A., Wang, Q., Gautam, N., 2005. Managing server energy and operational costs in hosting centers. ACM SIGMETRICS Perform. Eval. Rev. 33 (1), 303–314.
Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I., Warfield, A., 2005. Live migration of virtual machines. In: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2. NSDI'05, USENIX Association, Berkeley, CA, USA. pp. 273–286.
Cook, G., Horn, J.V., 2011. How Dirty is Your Data? A Look at the Energy Choices that Power Cloud Computing, 1–36.
Cully, B., Lefebvre, G., Meyer, D., Feeley, M., Hutchinson, N., Warfield, A., 2008. Remus: High availability via asynchronous virtual machine replication. In: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation, San Francisco. pp. 161–174.
Dai, Y.-S., Yang, B., Dongara, J., Zhang, G., 2010. Cloud Service Reliability: Modeling

and Analysis, 1–17.
Daly, J.T., 2006. A higher order estimate of the optimum checkpoint interval for restart dumps. Future Gener. Comput. Syst. 22 (3), 303–312.
David, H., Fallin, C., Gorbatov, E., Hanebutte, U.R., Mutlu, O., 2011. Memory power management via dynamic voltage/frequency scaling. In: Proceedings of the 8th ACM international conference on Autonomic computing. ACM, Karlsruhe, Germany. pp. 31–40.
Deng, W., Liu, F., Jin, H., Liao, X., Liu, H., Chen, L., 2012. Lifetime or energy: Consolidating servers with reliability control in virtualized cloud datacenters. In: Proceedings of the 4th International Conference on Cloud Computing Technology and Science (CloudCom). IEEE, Taipei, Taiwan. pp. 18–25.
Devadas, S., Malik, S., 1995. A survey of optimization techniques targeting low power vlsi circuits. In: Proceedings of the 32nd Annual ACM/IEEE Design Automation Conference. ACM/IEEE, San Francisco, CA, USA. pp. 242–247.
Duda, A., 1983. The effects of checkpointing on program execution time. Inf Process Lett 16 (5), 221–229.
Egwutuoha, I., Chen, S., Levy, D., Selic, B., Calvo, R., 2013. Energy efficient fault tolerance for high performance computing (HPC) in the cloud. In: Proceedings of the Sixth International Conference on Cloud Computing (CLOUD). IEEE, Santa Clara, CA, USA. pp. 762–769. ⟨http://dx.doi.org/10.1109/CLOUD.2013.69⟩.
el Mehdi Diouri, M., Glück, O., Lefevre, L., Cappello, F., 2012. Energy considerations in checkpointing and fault tolerance protocols. In: Proceedings of the IFIP International Conference on Dependable Systems and Networks Workshops (DSN). IEEE, Boston, MA, USA. pp. 1–6.
Elnozahy, E.N., Alvisi, L., Wang, Y.-M., Johnson, D.B., 2002. A survey of rollback-recovery protocols in message-passing systems. ACM Comput. Surv. (CSUR) 34 (3), 375–408.
Engelmann, C., Geist, A., 2005. Super-Scalable Algorithms for Computing on 100,000 Processors, 3514, 313–321.
Faragardi, H.R., Rajabi, A., Shojaee, R., Nolte, T., 2013. Towards energy-aware resource scheduling to maximize reliability in cloud computing systems. In: Proceedings of the 10th International Conference on High Performance Computing and Communications (HPCC). IEEE, Dalian, China. pp. 1469–1479.
Fu, S., Xu, C.-Z., 2007. Exploring event correlation for failure prediction in coalitions of clusters. In: Proceedings of the Conference on Supercomputing (SC'07). ACM/IEEE, Reno, NV, USA. pp. 1–12.
Fu, S., 2010. Failure-aware resource management for high-availability computing clusters with distributed virtual machines. J. Parallel Distrib. Comput. 70 (4), 384–393.
Gallet, M., Yigitbasi, N., Javadi, B., Kondo, D., Iosup, A., Epema, D., 2010. A model for space-correlated failures in large-scale distributed systems. In: Euro-Par 2010-Parallel Processing. Springer, Ischia, Italy. pp. 88–100.
Gandhi, A., Harchol-Balter, M., Das, R., Lefurgy, C., 2009. Optimal power allocation in server farms. ACM SIGMETRICS Perform. Eval. Rev. 37 (1), 157–168.
Gandhi, A., Chen, Y., Gmach, D., Arlitt, M., Marwah, M., 2011. Minimizing data center sla violations and power consumption via hybrid resource provisioning. In: Proceedings of the International Green Computing Conference and Workshops (IGCC), IEEE, Orlando, FL, USA, pp. 1–8.
Gao, A., Diao, L., 2010. Lazy update propagation for data replication in cloud computing. In: Proceedings of the 5th International Conference on Pervasive Computing and Applications (ICPCA). IEEE, Maribor, Slovenia. pp. 250–254.
Garg, S.K., Yeo, C.S., Anandasivam, A., Buyya, R., 2011. Environment-conscious scheduling of hpc applications on distributed cloud-oriented data centers. J. Parallel. Distrib. Comput. 71 (6), 732–749.
Garg, S.K., Yeo, C.S., Buyya, R., 2011. Green cloud framework for improving carbon efficiency of clouds. Eur-Par 2011 Parallel Process. 6852, 491–502.
Garraghan, P., Moreno, I.S., Townend, P., Xu, J., 2014. An analysis of failure-related energy waste in a large-scale cloud environment. IEEE Trans. Emerg. Top. Comput. 2 (2), 166–180.
Ghorbani, M., Wang, Y., Xue, Y., Pedram, M., Bogdan, P., 2014. Prediction and control of bursty cloud workloads: a fractal framework. In: Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis. ACM, New Delhi, India, pp. 1–9.
Giacobbe, M., Celesti, A., Fazio, M., Villari, M., Puliafito, A., 2015. Towards energy management in cloud federation: a survey in the perspective of future sustainable and cost-saving strategies. Comput. Netw. 91, 438–452.
Guerraoui, R., Schiper, A., 1996. Fault-tolerance by replication in distributed systems. In: Proceedings of Reliable Software Technologies-Ada-Europe'96. Springer, Montreux, Switzerland. pp. 38–57.
Gurumurthi, S., Sivasubramaniam, A., Kandemir, M., Franke, H., 2003. Reducing disk power consumption in servers with DRPM. Computer 12, 59–66.
Hameed, A., Khoshkbarforoushha, A., Ranjan, R., Jayaraman, P.P., Kolodziej, J., Balaji, P., Zeadally, S., Malluhi, Q.M., Tziritas, N., Vishnu, A., et al., 2014. A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems. Computing, 1–24.
Hines, M.R., Deshpande, U., Gopalan, K., 2009. Post-copy live migration of virtual machines. ACM SIGOPS Oper. Syst. Rev. 43 (3), 14–26.
P. Institute, 2016. Cost of Data Center Outages. pp. 1–21.
Islam, S., Keung, J., Lee, K., Liu, A., 2012. Empirical prediction models for adaptive resource provisioning in the cloud. Future Gener. Comput. Syst. 28 (1), 155–162.
Javadi, B., Abawajy, J., Buyya, R., 2012. Failure-aware resource provisioning for hybrid cloud infrastructure. J. Parallel Distrib. Comput. 72 (10), 1318–1331.
Javadi, B., Thulasiraman, P., Buyya, R., 2013. Enhancing performance of failure-prone clusters by adaptive provisioning of cloud resources. J. Supercomput. 63 (2), 467–489.
Jhawar, R., Piuri, V., Santambrogio, M., 2013. Fault tolerance management in cloud computing: a system-level perspective. IEEE Syst. J. 7 (2), 288–297.
Jula, A., Sundararajan, E., Othman, Z., 2014. Cloud computing service composition: a systematic literature review. Expert Syst. Appl. 41 (8), 3809–3824.
Jung, D., Chin, S., Chung, K.S., Yu, H., 2013. Vm migration for fault tolerance in spot instance based cloud computing. Grid Pervasive Comput. 7861, 142–151.
Khosravi, A., Garg, S.K., Buyya, R., 2013. Energy and carbon-efficient placement of virtual machines in distributed cloud data centers. Eur-Par 2013 Parallel Process 8097, 317–328.
Kim, J., Rotem, D., 2012. FREP: energy proportionality for disk storage using replication. J. Parallel Distrib. Comput. 72 (8), 960–974.
Kondo, M., Nakamura, H., 2005. Dynamic processor throttling for power efficient computations. Lecture Notes in Computer Science 3471. pp. 120–134.
Kondo, D., Javadi, B., Iosup, A., Epema, D., 2010. The failure trace archive: Enabling comparative analysis of failures in diverse distributed systems. In: Proceedings of the 10th International Conference on Cluster, Cloud and Grid Computing (CCGrid). IEEE/ACM, Melbourne, Victoria, Australia. pp. 398–407.
Kord, N., Haghighi, H., 2013. An energy-efficient approach for virtual machine placement in cloud based data centers. In: Proceedings of the 5th Conference on Information and Knowledge Technology (IKT). IEEE, Shiraz, Iran. pp. 44–49.
L'Ecuyer, P., Malenfant, J., 1988. Computing optimal checkpointing strategies for rollback and recovery systems. IEEE Trans. Comput. 37 (4), 491–496.
Le, T., Wright, D., 2015. Scheduling workloads in a network of datacentres to reduce electricity cost and carbon footprint. Sustain. Comput.: Inform. Syst. 5, 31–40.
Le Sueur, E., Heiser, G., 2010. Dynamic voltage and frequency scaling: the laws of diminishing returns. In: Proceedings of the international conference on Power Aware Computing and Systems. USENIX Association, Vancouver, BC, Canada. pp. 1–5.
Lee, Y.C., Zomaya, A.Y., 2012. Energy efficient utilization of resources in cloud computing systems. J. Supercomput. 60 (2), 268–280.
Lee, J., Youn, J.M., Cho, D., Paek, Y., 2013. Reducing instruction bit-width for low-power vliw architectures. ACM Trans. Des. Autom. Electron. Syst. (TODAES) 18 (2), 25.
Lefèvre, L., Orgerie, A.-C., 2010. Designing and evaluating an energy efficient cloud. J. Supercomput. 51 (3), 352–373.
Lemarinier, P., Bouteiller, A., Herault, T., Krawezik, G., Cappello, F., 2004. Improved message logging versus improved coordinated checkpointing for fault tolerant mpi. In: International Conference on Cluster Computing. IEEE, San Diego, CA, USA. pp. 115–124.
Lim, M.Y., Rawson, F., Bletsch, T., Freeh, V.W., 2009. Padd: Power aware domain distribution. In: Proceedings of the 29th International Conference on Distributed Computing Systems (ICDCS'09). IEEE, Montreal, Quebec, Canada. pp. 239–247.
Lin, J.-C., Leu, F.-Y., Chen, Y.-p., 2013. Analyzing job completion reliability and job energy consumption for a general mapreduce infrastructure. J. High Speed Netw. 19 (3), 203–214.
Liu, H., Jin, H., Liao, X., Hu, L., Yu, C., 2009. Live migration of virtual machine based on full system trace and replay. In: Proceedings of the 18th ACM international symposium on High performance distributed computing. ACM, Munich, Germany. pp. 101–110.
Ma, F., Liu, F., Liu, Z., 2010. Live virtual machine migration based on improved pre-copy approach. In: IEEE International Conference on Software Engineering and Service Sciences. IEEE. pp. 230–233.
Mell, P., Grance, T. The Nist Definition of Cloud Computing, 2011.
Meroufel, B., Belalem, G., 2014. Adaptive time-based coordinated checkpointing for cloud computing workflows. Scalable Comput.: Pract. Exp. 15 (2), 153–168.
Meyer, H., Rexachs, D., Luque, E., 2014. Hybrid message logging. Combining advantages of sender-based and receiver-based approaches. Procedia Comput. Sci. 29, 2380–2390.
Mezmaz, M., Melab, N., Kessaci, Y., Lee, Y.C., Talbi, E.-G., Zomaya, A.Y., Tuyttens, D., 2011. A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. J. Parallel Distrib. Comput. 71 (11), 1497–1508.
Mickens, J.W., Noble, B.D., 2006. Exploiting availability prediction in distributed systems. In: Proceedings of the (NSDI'06), 3rd Symposium on Networked Systems Design and Implementation, San Jose, CA, USA, pp. 73–86.
Mike, A., Shannon, B., David, B., Sean, F., Margaret, L., Tim, R., Michael, R., Dan, R., Frank, S., Sian, S., Jason, W., 2014. An introduction to designing reliable cloud services 2–14.
Milojičić, D.S., Douglis, F., Paindaveine, Y., Wheeler, R., Zhou, S., 2000. Process migration. ACM Comput. Surv. (CSUR) 32 (3), 241–299.
Nguyen, T., Shi, W., 2010. Improving resource efficiency in data centers using reputation-based resource selection. In: Proceedings of the International Green Computing Conference, Chicago, IL, USA, pp. 389–396. ⟨http://dx.doi.org/10.1109/GREENCOMP.2010.5598290⟩.
Pezoa, J.E., Hayat, M.M., 2014. Reliability of heterogeneous distributed computing systems in the presence of correlated failures. IEEE Trans. Parallel Distrib. Syst. 25 (4), 1034–1043.
Philp, I., 2005. Software failures and the road to a petaflop machine. In: HPCRI: 1st Workshop on High Performance Computing Reliability Issues, in Proceedings of the 11th International Symposium on High Performance Computer Architecture (HPCA-11). San Francisco, California, USA. pp. 125–128.
Pinheiro, E., Bianchini, R., Dubnicki, C., 2006. Exploiting redundancy to conserve energy in storage systems. ACM SIGMETRICS Perform. Eval. Rev. 34 (1), 15–26.
Plank, J.S., Elwasif, W.R., 1998. Experimental assessment of workstation failures and their impact on checkpointing systems. In: Proceedings of the Twenty-Eighth Annual International Symposium on Fault-Tolerant Computing. IEEE, Munich, Germany. pp. 48–57.

Portnoy, M., 2012. Virtualization Essentials. vol. 19. John Wiley & Sons, United States.
Quality Excellence for Suppliers of Telecommunications Forum (QuEST Forum), 2010.
Rangarajan, S., Garg, S., Huang, Y., 1998. Checkpoints-on-demand with active replication. In: Proceedings of the Seventeenth Symposium on Reliable Distributed Systems. IEEE, West Lafayette, Indiana, USA. pp. 75–83.
Salfner, F., Lenk, M., Malek, M., 2010. A survey of online failure prediction methods. ACM Comput. Surv. (CSUR) 42 (3), 10.
Sampaio, A.M., Barbosa, J.G., 2014. Towards high-available and energy-efficient virtual computing environments in the cloud. Future Gener. Comput. Syst. 40, 30–43.
Sams, S.L., 2011. Discovering Hidden Costs in Your Data Centre - A CFO Perspective. pp. 1–4.
Schroeder, B., Gibson, G., et al., 2010. A large-scale study of failures in high-performance computing systems. IEEE Trans. Dependable Secur. Comput. 7 (4), 337–350.
Sharma, Y., Sharma, A., Sengupta, J., 2010. Performance evaluation of mobile ad hoc network routing protocols under various security attacks. In: Proceedings of the International Conference on Methods and Models in Computer Science (ICM2CS). IEEE, New Delhi, India. pp. 117–124.
Sharma, Y., Javadi, B., Si, W., 2015. On the reliability and energy efficiency in cloud computing. In: Proceedings of the 13th Australian Symposium on Parallel and Distributed Computing (AusPDC 2015). Vol. 37, Sydney, NSW, Australia. pp. 111–114.
Shawish, A., Salama, M., 2014. Cloud computing: paradigms and technologies. In: Inter-cooperative Collective Intelligence: Techniques and Applications. Springer. pp. 39–67.
Shribman, A., Hudzia, B., 2013. Pre-copy and post-copy vm live migration for memory intensive applications. In: Euro-Par 2012: Parallel Processing Workshops. Springer. pp. 539–547.
Srikantaiah, S., Kansal, A., Zhao, F., 2008. Energy aware consolidation for cloud computing. In: Proceedings of the Conference on Power Aware Computing and Systems. ACM, San Diego, California. pp. 1–10.
Subbiah, S. Clonescale: Distributed Resource Scaling for Virtualized Cloud Systems, 2012.
Subirats, J., Guitart, J., 2015. Assessing and forecasting energy efficiency on cloud computing platforms. Future Gener. Comput. Syst. 45, 70–94.
Subrata, R., Zomaya, A.Y., Landfeldt, B., 2010. Cooperative power-aware scheduling in grid computing environments. J. Parallel Distrib. Comput. 70 (2), 84–91.
Sun, D.-W., Chang, G.-R., Gao, S., Jin, L.-Z., Wang, X.-W., 2012. Modeling a dynamic data replication strategy to increase system availability in cloud computing environments. J. Comput. Sci. Technol. 27 (2), 256–272.
Tesfatsion, S., Wadbro, E., Tordsson, J., 2014. A combined frequency scaling and application elasticity approach for energy-efficient cloud computing. Sustain. Comput.: Inform. Syst. 4 (4), 205–214.
Tiwana, B., Balakrishnan, M., Aguilera, M.K., Ballani, H., Mao, Z.M., 2010. Location, location, location!: modeling data proximity in the cloud. In: Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks. ACM, Monterey, CA, USA. pp. 1–6.
Tolentino, M.E., Turner, J., Cameron, K.W., 2007. Memory-miser: a performance-constrained runtime system for power-scalable clusters. In: Proceedings of the 4th international conference on Computing Frontiers. ACM, Ischia, Italy. pp. 237–246.
Toosi, A.N., Calheiros, R.N., Buyya, R., 2014. Interconnected cloud computing environments: challenges, taxonomy, and survey. ACM Comput Surv (CSUR) 47 (1), 7:1–7:47.
Vaidyanathan, K., Harper, R.E., Hunter, S.W., Trivedi, K.S., 2001. Analysis and implementation of software rejuvenation in cluster systems. ACM SIGMETRICS Perform. Eval. Rev. 29 (1), 62–71.
Valentini, G.L., Lassonde, W., Khan, S.U., Min-Allah, N., Madani, S.A., Li, J., Zhang, L., Wang, L., Ghani, N., Kolodziej, J., et al., 2013. An overview of energy efficiency techniques in cluster computing systems. Clust. Comput. 16 (1), 3–15.
Verma, A., Ahuja, P., Neogi, A., 2008. pMapper: power and migration cost aware application placement in virtualized systems. Middleware 5346, 243–264.
Vishwanath, K.V., Nagappan, N., 2010. Characterizing cloud computing hardware reliability. In: Proceedings of the 1st ACM symposium on Cloud computing, ACM, Indianapolis, IN, USA. pp. 193–204.
Voorsluys, W., Buyya, R., 2012. Reliable provisioning of spot instances for compute-intensive applications. In: Proceedings of the 26th International Conference on Advanced Information Networking and Applications (AINA). IEEE, Fukuoka, Japan. pp. 542–549.
Wadhwa, B., Verma, A., 2014. Energy and carbon efficient vm placement and migration technique for green cloud datacenters. In: Proceedings of the Seventh International Conference on Contemporary Computing (IC3). IEEE, Noida, India. pp. 189–193.
Walters, J.P., Chaudhary, V., 2009. A fault-tolerant strategy for virtualized HPC clusters. J. Supercomput. 50 (3), 209–239.
Wang, S.-S., Wang, S.-C., 2014. The consensus problem with dual failure nodes in a cloud computing environment. Inf. Sci. 279, 213–228.
Wang, C.-F., Hung, W.-Y., Yang, C.-S., 2014. A prediction based energy conserving resources allocation scheme for cloud computing. In: Proceedings of International Conference on Granular Computing (GrC). IEEE, Noboribetsu, Hokkaido, Japan. pp. 320–324.
Yao, L., Wu, G., Ren, J., Zhu, Y., Li, Y., 2013. Guaranteeing fault-tolerant requirement load balancing scheme based on vm migration. Comput. J. 56 (2), 1–8.
Yigitbasi, N., Gallet, M., Kondo, D., Iosup, A., Epema, D., 2010. Analysis and modeling of time-correlated failures in large-scale distributed systems. In: Proceedings of the 11th International Conference on Grid Computing (GRID). IEEE/ACM, Brussels, Belgium. pp. 65–72.
Yu, X., Ning, P., Vouk, M.A., 2015. Enhancing security of hadoop in a public cloud. In: Proceedings of the 6th International Conference on Information and Communication Systems (ICICS). IEEE, Amman, Jordan. pp. 38–43.
Zhang, L., Li, K., Xu, Y., Mei, J., Zhang, F., Li, K., 2015. Maximizing reliability with energy conservation for parallel task scheduling in a heterogeneous cluster. Inf. Sci. 319, 113–131.
Zhu, Q., Zhou, Y., 2005. Power-aware storage cache management. IEEE Trans. Comput. 54 (5), 587–602.
Zhu, D., Melhem, R., Mossé, D., 2004. The effects of energy management on reliability in real-time embedded systems. In: Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD-2004). IEEE. pp. 35–40.