FTCloudSim: support for cloud service reliability enhancement simulation

Ao Zhou, Shangguang Wang*, Chenchen Yang, Lei Sun, Qibo Sun and Fangchun Yang

State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications,
Haidian, Beijing 100876, China
Email: hellozhouao@gmail.com
Email: sgwang@bupt.edu.cn
Email: ccyang@bupt.edu.cn
Email: leisun@bupt.edu.cn
Email: qbsun@bupt.edu.cn
Email: fcyang@bupt.edu.cn
*Corresponding author

Abstract: Recently, an increasing number of companies have begun to deploy their application services in the cloud. However, the cloud data centre downtime has negatively affected the quality of cloud service. Many researchers have studied the problem of cloud service reliability assurance. However, there is a shortage of tools that enable researchers to evaluate their newly proposed cloud service reliability enhancement mechanisms. To fill this gap, in this paper, we propose FTCloudSim as a reliable cloud data centre simulation system based on the basic functionalities of CloudSim. FTCloudSim provides an extensible interface to help researchers implement new cloud service reliability enhancement mechanisms. In addition, FTCloudSim can also study the behaviour of the newly proposed mechanisms. We demonstrate the capabilities of FTCloudSim by using five reliability enhancement mechanisms. The results indicate the benefits of our proposed simulation system.

Keywords: cloud computing; CloudSim; data centre; cloud service; reliability; fault tolerance.

Reference to this paper should be made as follows: Zhou, A., Wang, S., Yang, C., Sun, L., Sun, Q. and Yang, F. (2015) 'FTCloudSim: support for cloud service reliability enhancement simulation', *Int. J. Web and Grid Services*, Vol. 11, No. 4, pp.347–361.

Biographical notes: Ao Zhou is currently an Assistant Professor at Beijing University of Posts and Telecommunications. She received her PhD degree in Computer Science and Technology from Beijing University of Posts and Telecommunications in 2015. Her research interests include cloud computing and service reliability.

Shangguang Wang is an Associate Professor at Beijing University of Posts and Telecommunications. He received his PhD degree in Computer Science and Technology from Beijing University of Posts and Telecommunications in 2011. His research interests include service computing and cloud computing.

He served as Guest Editor for *IEEE System Journal*, *Journal of Computational Science*, *Journal on Wireless Communications and Networking*, etc. He also served as PC Chair for IOV 2014 and SC2 2014.

Chenchen Yang is a Postgraduate student at Beijing University of Posts and Telecommunications, China. His research interests include service selection and cloud service.

Lei Sun is a Postgraduate student at Beijing University of Posts and Telecommunications, China. His research interests include service selection and cloud service.

Qibo Sun is currently an Associate Professor at Beijing University of Posts and Telecommunications, China. He received his PhD degree in Communication and Electronic System from Beijing University of Posts and Telecommunications in 2002. He is a Member of the China Computer Federation. His research interests include services computing, Internet of Things and network security.

Fangchun Yang is currently a Professor at Beijing University of Posts and Telecommunications, China. He received his PhD degree in Communication and Electronic System from the Beijing University of Posts and Telecommunications in 1990. He has published six books and more than 80 papers. His research interests include network intelligence, services computing, communications software, softswitching technology, and network security. He is a fellow of the IET.

1 Introduction

Because of its ability to allocate resources dynamically and instantaneously as needed, cloud computing has become a popular computing model nowadays (Armbrust et al., 2010; Marston et al., 2011). As cloud computing allows the physical resources to be shared by a vast number of cloud service providers (Buyya et al., 2009), companies and other organisations have begun deploy their application services in the cloud to save on the costs of maintaining their own infrastructure (Fox et al., 2009; Calheiros et al., 2011).

However, there are hundreds or thousands of host servers and virtual machines in a cloud data centre where failures are the norm, rather than the exception (Bauer and Adams, 2012; Schwarzkopf et al., 2012). Therefore, this problem is being thoroughly studied. Researchers have begun to study the problem and have proposed many fault-tolerant mechanisms to enhance the reliability of cloud services (Dai et al., 2009; Undheim et al., 2011; Do et al., 2013). However, there is a shortage of tools that enable researchers to evaluate their newly proposed cloud service reliability enhancement mechanisms.

Actually, a researcher can evaluate his new mechanism in two ways. Firstly, he can conduct the experiment in a real cloud data centre. There are mainly two shortcomings in this. For one, cloud data centres are sometimes not easy to access. Making investments to build one's own infrastructures are very uneconomic, but Amazon or Google may not allow one to modify one's system management facilities. The other thing is that a comprehensive evaluation of the newly proposed mechanism in a real data centre is extremely difficult. It requires the interaction of a large amount of host severs, virtual machines, storage systems and network elements. The complex conditions may go out of the control of researchers, which can impact the quality of evaluation. However, the problem can be solved by using a cloud simulation toolkit.

CloudSim (Calheiros et al., 2011) is a cloud simulation toolkit developed by the CLOUDS Laboratory of the University of Melbourne. CloudSim supports the simulation of a virtualised cloud data centre. It enables researchers to experiment on cloud computing infrastructures. However, CloudSim cannot support the functions related to reliability enhancement currently. Fortunately for the researches, CloudSim is an extensible simulation tool. Researchers can extend the existing functionalities provided by CloudSim, and add new features to CloudSim, including CloudAnalyst (http://www.cloudbus.org/cloudsim/CloudAnalyst.zip), CloudAuction (http://www.cloudbus.org/cloudsim/CloudAuctionV2.0.zip) and DynamicCloudSim (Bux and Leser, 2013), among others. Although DynamicCloudSim can support fault tolerance in some ways, currently it can only determine whether a task succeeds or fails. None of the available current tools can properly aid researchers in evaluating their newly proposed mechanisms.

In this work, to overcome these limitations of current cloud simulation toolkits, we present FTCloudSim, a CloudSim-based system which can model and simulate the cloud service reliability enhancement mechanisms. An extensible interface is provided in FTCloudSim to aid researchers in implementing new mechanisms easily. In addition, FTCloudSim can trigger failure events to test the performance of each mechanism. After execution, it will generate information on the necessary metrics to highlight the advantages and shortcomings of the mechanism.

The rest of the paper is organised as follows. Related work is discussed in Section 2. Section 3 introduces our proposed FTCloudSim. Section 4 gives an overview of our experiments. Finally, we conclude in Section 5.

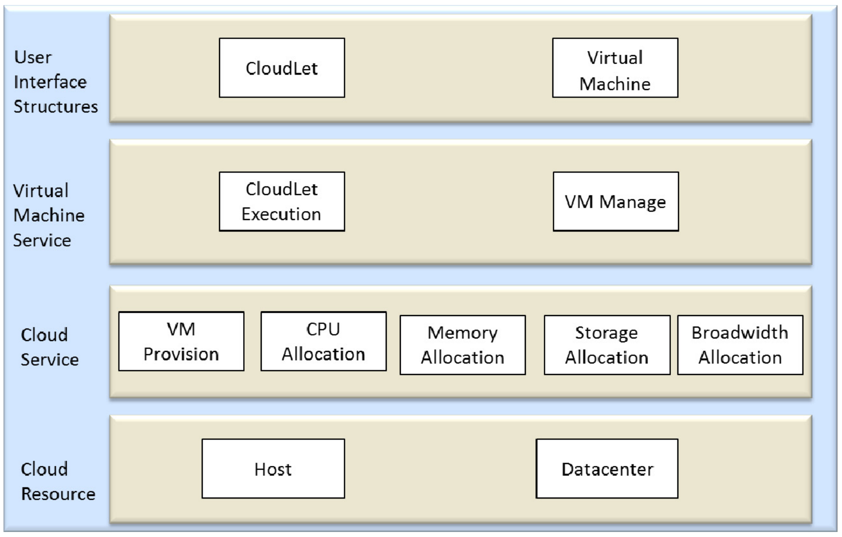
2 Related work

In this section, related simulation toolkits are discussed. There have been many simulation systems to support the research of cloud computing. The most famous simulator is CloudSim. CloudSim enables the simulation of cloud computing infrastructures and cloud applications. Researches can evaluate their resource allocation approaches or task schedule algorithms by using CloudSim.

CloudSim is a cloud simulation toolkit developed by the CLOUDS Laboratory of University of Melbourne. CloudSim supports the simulation of a virtualised cloud data centre. It enables researchers to experiment on cloud computing infrastructures. In CloudSim simulation, a data centre is composed of host servers and storage servers. A host server can host one or more virtual machines. A data centre assigns virtual machines to host servers based on its policy. Virtual machines process the cloudlets according to the policy of the Cloudlet Scheduler. All policies can be redefined by the user.

As shown in Figure 1, CloudSim consists of four layers. The first lever is the cloud resource level. In this level, a data centre consists of a set of host servers. Host represents a physical machine. The hosts have different configurations, such as CPU type, memory size and disk size. The second level is the cloud service level. The simulator will allocate physical resources to each virtual machine based on the allocation strategies. The virtual machines are scheduled at two levels: host server level and virtual level. This level is extensible. The researchers can implement their own allocation strategies by extending the basic one. The third level is the virtual machine service level. This level manages the execution of applications. The fourth level is the user interface structures level. The virtual machines can share the physical resources among others when they are placed in the same host server. Cloudlet models the application service. CloudSim now can model single application service and workflow application service. Every cloudlet has a pre-assigned length and amount of data needed to be fetched from the database for the successfully execution of the application.

Figure 1 The framework of CloudSim



But CloudSim still has some limitations. For example, functions related to reliability enhancement cannot be supported by CloudSim. Fortunately, Cloudsim is an extensible simulation framework. Hence, many researches extend some of the capabilities of CloudSim and add new features to it.

RealCloudSim (http://sourceforge.net/projects/realcloudsim) is capable of modelling network topologies. Users can construct a data centre network based on the BRITE format. Researches can build a complex data centre network according to their designed network connection methods. However, it cannot distinguish the switches with different configurations. To overcome these limitations, NetworkCloudSim (Garg and Buyya, 2011) is presented to model network topologies and parallel applications. It adds root switches, aggregated switches and edge switches to CloudSim.

CloudAnalyst is also a tool developed at the University of Melbourne. CloudAnalyst is built on top of CloudSim, and it supports location-distributed application. CloudAnalyst can simulate the location of data centres and the traffic between two data centres. Therefore, it helps researches to see how distributed applications are processed in location-distributed data centres.

There are still other CloudSim-based simulation toolkits. CloudAuction implements the auction-based mechanism in CloudSim. CloudReports (https://github.com/ thiagotts/CloudReports) is developed for the purpose of simulating distributed computing environments. CloudMIG Xpress (http://sourceforge.net/projects/ cloudmigxpress/) can show the cost of migrating software to the cloud environment. WorkflowSim (https:// github.com/WorkflowSim/) can support workflow preparation and workflow job scheduler. DynamicCloudSim is a tool that can simulate the dynamic changes to the performance of virtual machines. Although DynamicCloudSim can support fault tolerance in some way, currently, it can only determine whether a task succeeds or fails.

Therefore, none of the current tools can simulate cloud service reliability enhancement mechanism perfectly. For example, currently, the tools cannot trigger the host failure events or virtual machine failure events. But all this can be simulated by FTCloudSim. We will show the design of FTCloudSim in the next section.

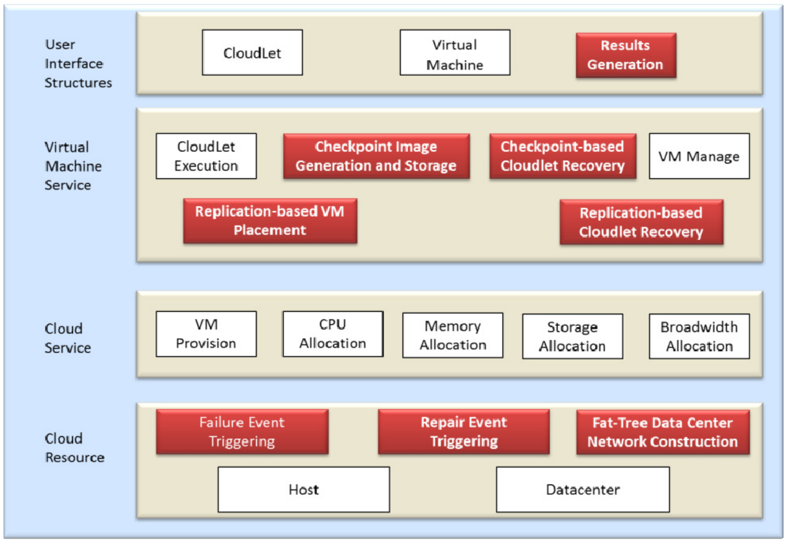
3 Design of FTCloudSim

We will present the design of FTCloudSim in this section.

3.1 System architecture

As shown in Figure 2, FTCloudSim has added eight modules to CloudSim, which will be described in this section. The newly added function modules are divided into three types: fundamental functions, support for checkpoint and support for replication. Almost all the reliability enhancement methods are based on the exploitation of redundancy. Replication and checkpointing are two widely used basic service reliability enhancement mechanisms. FTCloudSim can support checkpointing- and replication-based fault-tolerant mechanism, as of now.

Figure 2 The framework of FTCloudSim



3.2 Fundamental functions

The fundamental functions of FTCloudSim include the following:

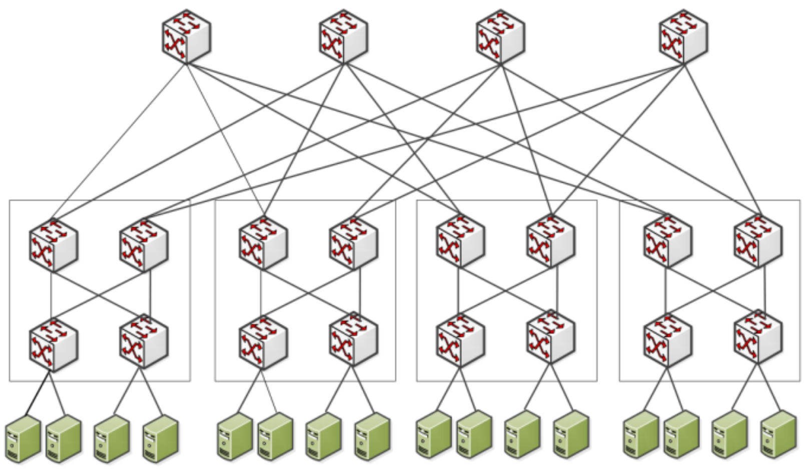
1 *Fat-tree data centre network construction.* Although CloudSim supports the simulation of data centre network topologies, the user needs to construct the network

himself. The process is troublesome if the network is complex. The fat-tree data centre network (Al-Fares et al., 2008; Bilal et al., 2013) is a typical architecture of a current commodity data centre. Hence, we provide the functionality to construct a fat-tree data centre network automatically. The user only needs to set the port number in the configuration file. Figure 3 shows a fat-tree data centre network with four ports.

2 *Failure and repair event triggering*. Failure events and repair events are triggered. The failure events can be generated according to some special distribution (Rausand and Høyland, 2004), Weibull distribution or exponential distribution, etc. The failure event data and the repair event data can be saved to a file so the experiment can be repeated.

3 *Results generation.* This module outputs the simulation results to the user. In cloud computing environments, all resources are commercialised. Therefore, in addition to ensuring reliability, the method should reduce resource consumption based on data centre characteristics. Therefore, FTCloudSim will provide three types of metrics to highlight the advantages and shortcomings of each mechanism. The first metric type is the ability to enhance the reliability of cloud service. The metrics include the total execution time (total time the method takes to complete all tasks) and the average lost time (all time lost because of failure). The second metric is network resource usage. In addition to the total checkpoint image data transferred by all switches, the metric includes the total checkpoint image data transferred by the core switches, the aggregation switches and the edge switches. The third metric is storage resource usage. The metric includes the total disk usage (the disk usage for the storage of checkpoint image).

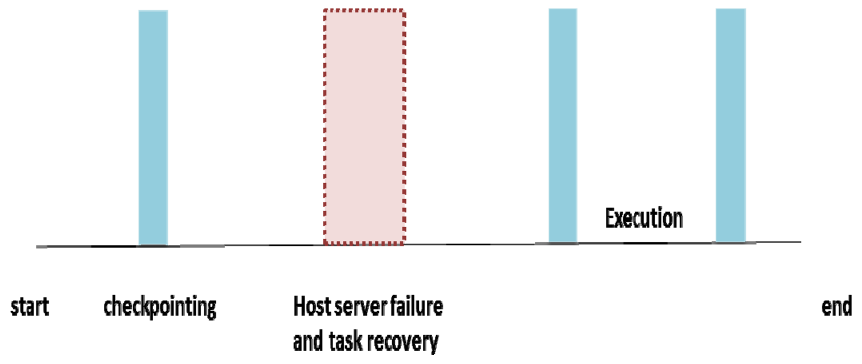
Figure 3 Fat-tree data centre network (see online version for colours)



3.3. *Support for checkpoint*

Checkpoint is an important reliability enhancement mechanism. As shown in Figure 4, in checkpoint mechanism, the running state of a virtual machine is periodically saved as a checkpoint image. The checkpoint image is stored in a persistent storage system. When a virtual machine fails, it can be restarted based on the checkpoint image. We have added two new modules to support checkpoint mechanism in CloudSim: checkpoint image generation and storage, and checkpoint-based cloudlet recovery.

Figure 4 Checkpointing (see online version for colours)



3.3.1 Checkpoint image generation and storage

A checkpoint image is generated, transferred and stored periodically based on the checkpoint mechanism. This module is extensible. The user can design its own checkpoint schedule strategy by extending the default one. The strategy can determine when to generate the image, the content of the image and where to store the image.

Figure 5 Sequence diagram: checkpoint image generation and storage

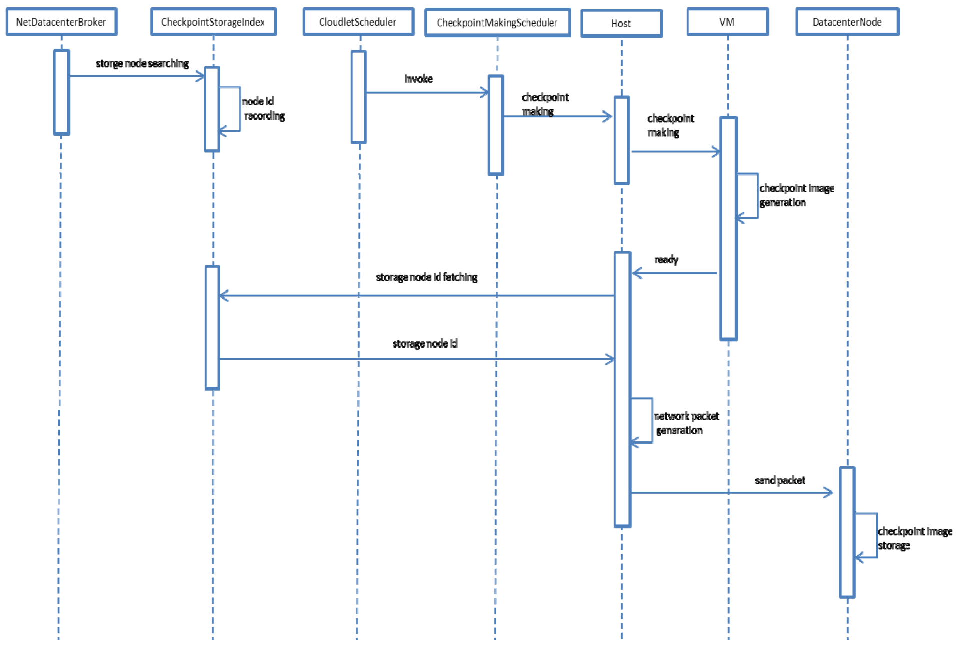


Figure 5 shows the sequence diagram of checkpoint image generation and storage.

- *CheckpointStorageIndex.* An abstract class represents the checkpoint image storage strategy. It serves as an index and stores the checkpoint image storage node for each virtual machine. CheckpointStorageIndex contains an abstract storage node searching method. A user can replace the method with a new checkpoint image storage strategy.
- *Checkpoint-making scheduler.* The class triggers the checkpoint-making event periodically. Based on the pre-defined rules, it alerts a host to save the current state of a special virtual machine as a checkpoint image. This module is extensible. The user can implement a new rule by extending the basic one. The rule can determine the time to make checkpoint, and the content of the checkpoint image.
- *DatacenterNode.* All simentities that can send and receive network packets are all the subclass of DatacenterNode. The hosts, the storage nodes and the switches are all DatacenterNode.

Firstly, the net data centre broker asks the checkpoint storage index to find a checkpoint image storage node for each virtual machine. When the checkpoint storage index receives the message, it searches an image storage node based on pre-defined rules. Before the execution, the scheduler asks the checkpoint-making scheduler to help remind the virtual machine execution state-saving time. When the state-saving time is up, the checkpoint-making scheduler sends the host a message to remind him. When the host receives the message, it saves current execution state of the virtual machine as a checkpoint image. Then, the host generates a network packet which contains the checkpoint image, and sends the packet to the storage node. The storage node stores the checkpoint image to the database when receiving the packet.

3.3.2 *Checkpoint-based cloudlet recovery*

A task is resumed from the failure based on the latest available checkpoint image. If there is no accessible checkpoint image, it will fetch the necessary data from the central database and restart the interrupted task from the beginning.

Figure 6 Sequence diagram: checkpoint-based recovery

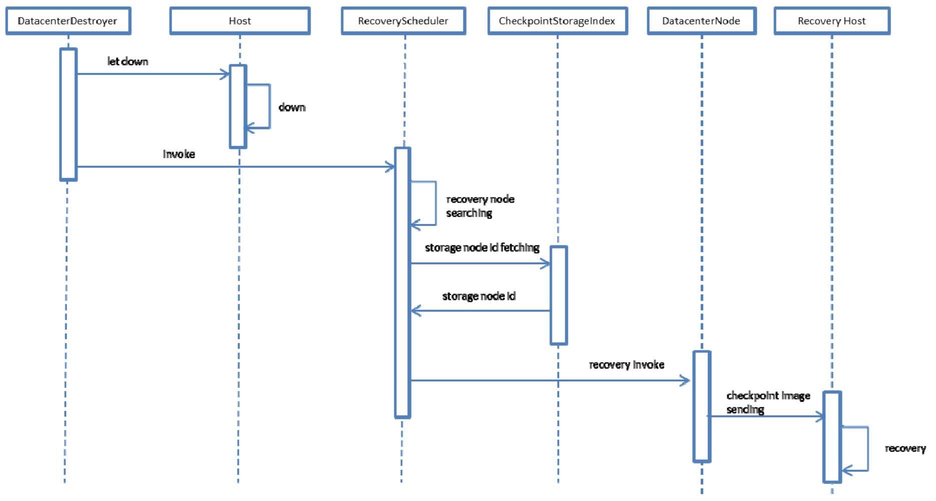


Figure 6 shows the sequence diagram of checkpoint-based cloudlet recovery.

- *Data centre destroyer.* The data centre destroyer destroys the data centre by breaking the hosts and virtual machines. The repair events of hosts and virtual machines are also triggered by the data centre destroyer. We add a bool variable to indicate whether a host or a virtual machine fails.
- *RecoveryScheduler.* An abstract class represents the recovery strategy. Recovery schedule will recover the service from failure event based on pre-defined mechanism. This module is also extensible. A user can replace the method with a new recovery strategy.

Firstly, the data centre destroyer breaks a host or a virtual machine. After the host or the virtual machine fails, the destroyer invokes recovery scheduler to recover the service. The recovery scheduler searches for a host with enough free resources for each interrupted virtual machine. Then, it asks the checkpoint storage index for the ID of the checkpoint image storage node. After receiving the information, the recovery scheduler sends a message containing the recovery host ID to the checkpoint image storage node. The checkpoint image storage node sends the related checkpoint image to the recovery host after receiving the message. Now, the recovery host recovers the service based on the checkpoint image.

3.4 Support for replication

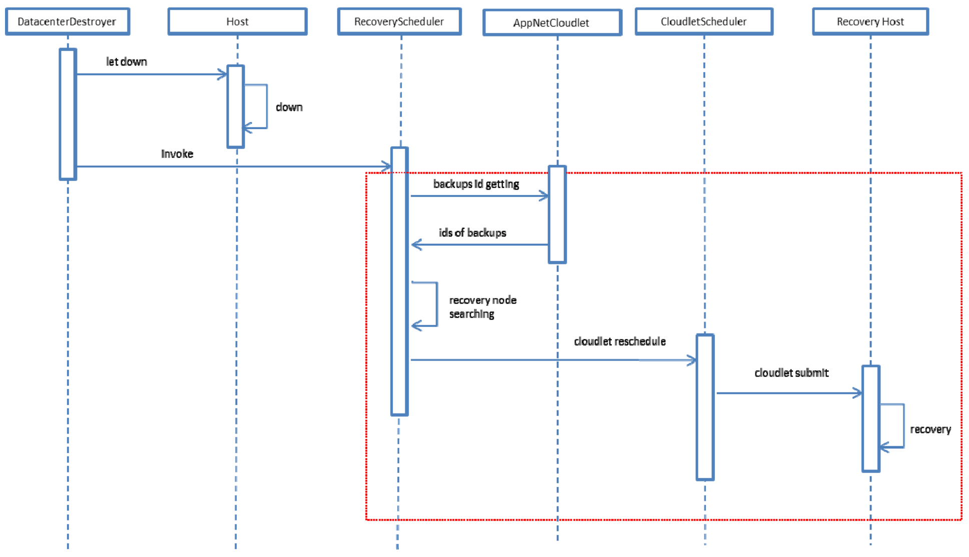
Replication is a reliability enhancement mechanism that relies on the exploitation of redundancy. For example, the *k*-fault tolerance replication mechanism provides more virtual machines than needed to ensure that all applications can be maintained, while any *k* virtual machines fail at the same time. In the standby mechanism, several virtual machines synchronously or asynchronously process the same task. We have added two new modules to support the replication mechanism in CloudSim: replication-based virtual machine placement and replication-based cloudlet recovery.

1. *Replication-based virtual machine placement*. To support replication in CloudSim, we have the extended class AppNetCloudlet. Information such as how many virtual machines are providing the application service, the location of the primary virtual machines and the location of the backup virtual machines are stored in this class.

   A user can extend the virtual machine schedule mechanism of cloudsim to determine the physical location of each primary or backup virtual machine. We have just implemented a random location selection mechanism. A user can replace the module with a new mechanism.

2. *Replication-based cloudlet recovery*. After the failure of a host server or a virtual machine, the affected cloudlets need to be reassigned to the backup virtual machines. Figure 7 shows the sequence diagram of replication-based cloudlet recovery. Firstly, the data centre destroyer breaks a host or a virtual machine. After the failure of the host or the virtual machine, the destroyer invokes the recovery scheduler to recover the service. The recovery scheduler fetches information of the backup of the virtual machine's location from the class AppNetCloudlet. When the recovery scheduler receivers the information, it searches for the recovery host for each cloudlet. All cloudlets are re-submitted to their recovery hosts. This module is extensible. A user can implement a new mechanism to determine how to reassign the cloudlets.

Figure 7 Sequence diagram: replication-based image storage (see online version for colours)



4 Experiments

We present the new functions of FTCloudSim by using five replication-based cloud service reliability enhancement mechanisms in this section. We have implemented the FTCloudSim system and published the source code (http://sguangwang. com/ftcloudsim. html). The following sections first describe the experimental setting. Then, experimental results are discussed.

4.1 Experimental set-up

Now let us consider the following cloud service reliability enhancement background. In cloud computing, a service is deployed in the virtual machine. Because there are tens of thousands of service requests, the computing power of a single virtual machine is not strong enough to process the large quantity of service requests. Hence, the service is deployed in several virtual machines. Each virtual machine has a waiting queue. All tasks assigned to the virtual machine are inserted into the queue.

In our experiment, we construct a 16-port fat-tree data centre network. The capacity of the core link and aggregation link is set as 10 Gbps, and the capacity of edge link is set as 1 Gbps. There are eight host servers in each subnet. Each host servers can host four virtual machines at the most. The transfer delay of the core switch, aggregation switch and edge switch are 1, 2 and 3 seconds, respectively. We generate 100 virtual machine failure events and 26,000 tasks. The task size is uniformly distributed between 5 and 10 minutes. We multiple the task size by 6, and add the result to the submit time as the deadline. All tasks are data-processing tasks. The data size is set as 300 MB. We name the number of primary virtual machines as service concurrency. The service concurrency is normally distributed between 10 and 20.

We will demonstrate the capabilities of FTCloudSim by using the following five replication-based cloud service reliability enhancement mechanisms:

- *Cold-Backup*. Suppose the service concurrency of a special service is *m*. After a host fails, the recovery scheduler searches a new virtual machine to provide the interrupted service.
- *Hot-Backup*. Suppose the service concurrency of a special service is *m*. There are *m* number of services providing virtual machines and three hot standby virtual machines. After a virtual machine fails, all the tasks in its waiting queue are reassigned to a backup virtual machine.
- *Head-First*. Suppose the service concurrency of a special service is *m*. There are *m* number of primary virtual machines and three hot standby virtual machines. All tasks are assigned to the (m + 3) service-providing virtual machine. If a virtual machine fails, the scheduler randomly selects a virtual machine for each task, and adds it to the head of the waiting queue.
- *Tail-First*. Suppose the service concurrency of a special service is *m*. There are *m* number of primary virtual machines and three hot standby virtual machines. All tasks are assigned to the (m + 3) service-providing virtual machine. If a virtual machine fails, the scheduler randomly selects a virtual machine for each task, and adds it to the tail of the waiting queue.
- *Random Selection*. Suppose the service concurrency of a special service is *m*. There are *m* number of primary virtual machines and three hot standby virtual machines. All tasks are assigned to the (m + 3) service-providing virtual machine. If a virtual machine fails, the scheduler randomly selects a virtual machine for each task and adds it to a random place in the waiting queue.

To attack the failure of the host servers, the methods keep a copy of backup data on another host server in the same subnet. Therefore, the data can be re-fetched from the data backup server or the host where the failed virtual machine is placed. We will evaluate the five mechanisms by using the following five metrics, and we only count the packets related to data backup and data re-fetching.

- *Failure rate:* the percentage of tasks that does not complete before deadline.
- *Root switch packets processed:* the total size of network packets that has been transferred by the root switches.
- *Aggregation switch packets processed:* the total size of network packets that has been transferred by the aggregation switches.
- *Edge switch packets processed:* the total size of network packets that has been transferred by the edge switches.
- *Total packets processed:* the total size of network packets that have been transferred by all the switches.

In cloud computing environments, all resources are commercialised. The mechanisms need to reduce resource consumption when enhancing the cloud service reliability. The metrics related to packets processed can show the performance in network resource consumption.

4.2 Results

The performance of all the mechanisms is studied. Figure 8 shows the performance of reliability enhancement. Figures 9–12 show the performance of network resource consumption, including the results of root switch packets processed, aggregation switch packets processed, edge switch packets processed and total packets processed.

Figure 8 Failure rate

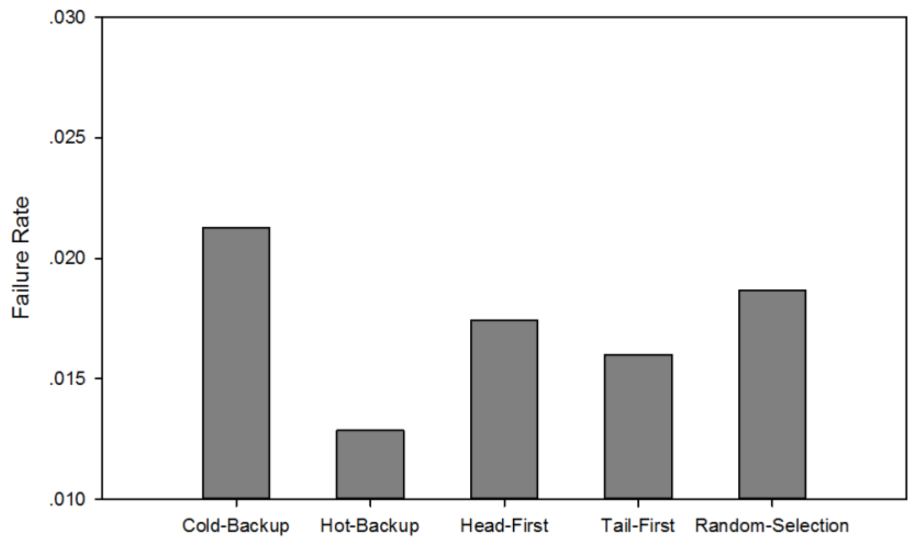
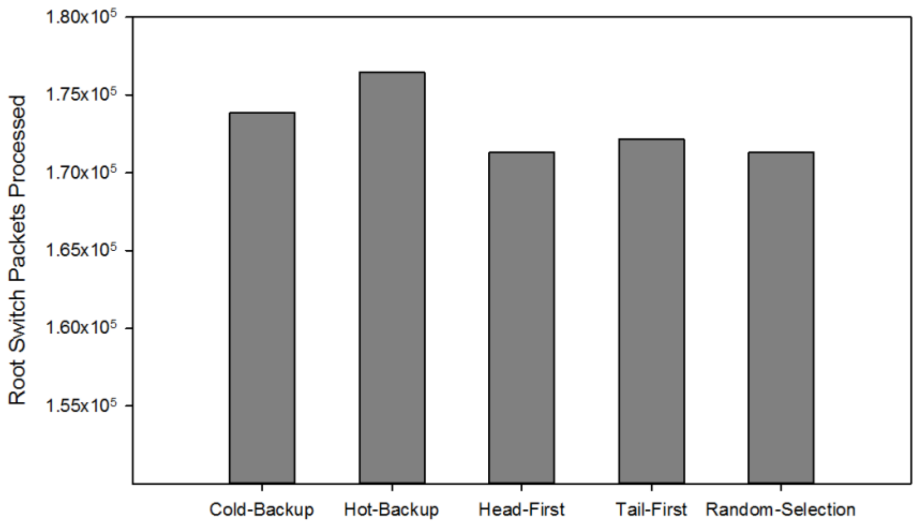


Figure 9 Root switch packets processed (MB)



As shown in Figure 8, the failure rate of Cold-Backup is higher than that of all other mechanisms. Hot-Backup shows the strongest reliability enhancement ability. As shown in Figures 9–12, the core-level network resource consumption and the aggregation-level network resource consumption of Head-First, Tail-First and Random Selection are almost the same. Of all the mechanisms, Hot-Backup consumes the most core-level network resource and aggregation-level network the least resource. However, the edge-level network resource consumption of all the mechanisms is almost the same. All in all, Hot-Backup consumes more network resource than the other mechanisms.

Figure 10 Aggregation switch packets processed (MB)

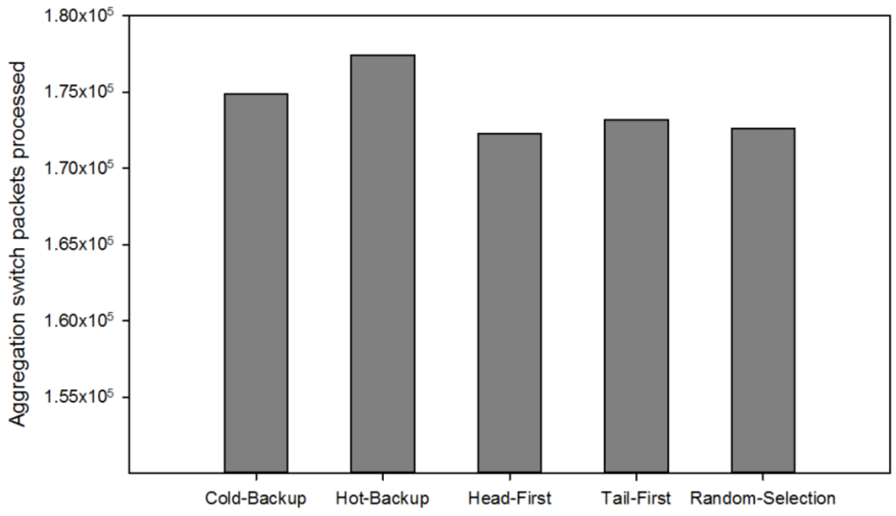


Figure 11 Edge switch packets processed (MB)

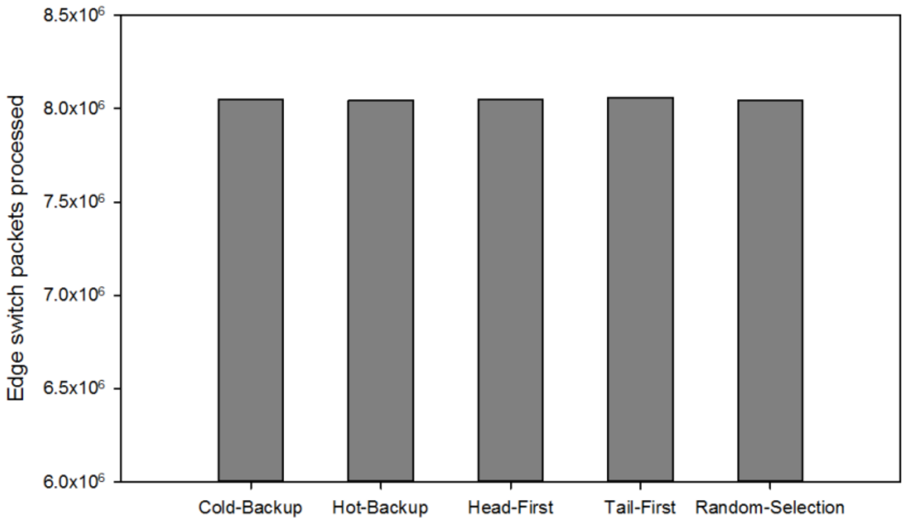
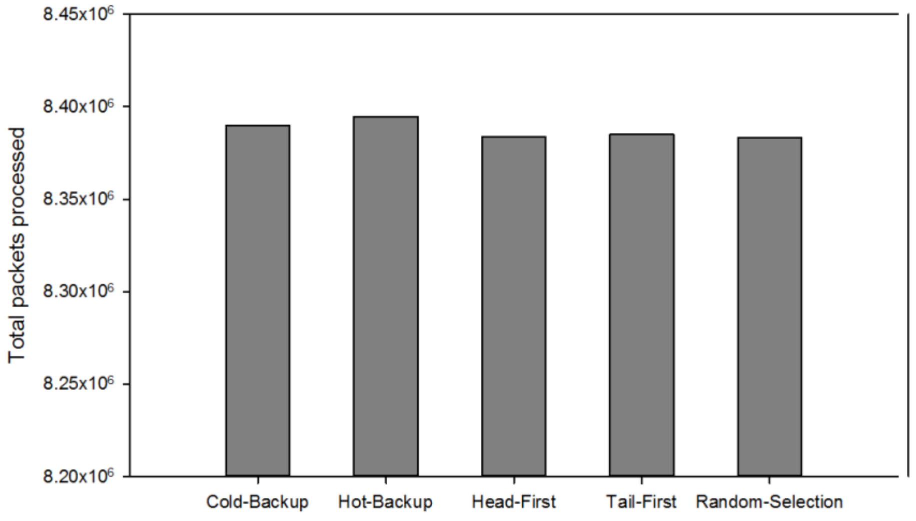


Figure 12 Total packets processed (MB)



Therefore, FTCloudSim can help highlight the advantages and shortcomings of the mechanisms.

5 Conclusion

In this paper, we presented a CloudSim-based system called FTCloudSim for modelling and simulation of cloud service reliability enhancement mechanism. FTCloudSim consists of the new features: (1) fat-tree data centre network construction, (2) failure event triggering, (3) repair event triggering, (4) checkpoint image generation and storage, (5) replication-based virtual machine placement, (6) checkpoint-based service recovery, (7) replication-based service recovery and (8) results generation. We present the new functions by using five cloud service reliability enhancement mechanisms. The results show that our system provides an easy-to-use reliability enhancement mechanism implementation interface. Furthermore, FTCloudSim can also show the advantages and shortcomings of each mechanism. Our future research involves providing easy-to-use user interface and visualisation results display.

Acknowledgements

The work presented in this study is supported by NSFC (61272521), SRFDP (20110005130001), the Fundamental Research Funds for the Central Universities (2014RC1101) and Beijing Natural Science Foundation (4132048).

References

Al-Fares, M., Loukissas, A. and Vahdat, A. (2008) 'A scalable, commodity data center network architecture', *ACM SIGCOMM Computer Communication Review*, Vol. 38, No. 4, pp.63–74.

Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A. and Stoica, I. (2010) 'A view of cloud computing', *Communications of the ACM*, Vol. 53, No. 4, pp.50–58.

Bauer, E. and Adams, R. (2012) *Reliability and Availability of Cloud Computing*, John Wiley & Sons, New York.

Bilal, K., Manzano, M., Khan, S., Calle, E., Li, K. and Zomaya, A. (2013) 'On the characterization of the structural robustness of data center networks', *IEEE Transactions on Cloud Computing*, Vol. 1, No. 1, pp.64–77.

Bux, M. and Leser, U. (2013) 'DynamicCloudSim: simulating heterogeneity in computational clouds', *Proceedings of the 2nd ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*, 22–27 June, New York, pp.1–12.

Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J. and Brandic, I. (2009) 'Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility', *Future Generation Computer Systems*, Vol. 25, No. 6, pp.599–616.

Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A. and Buyya, R. (2011) 'CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms', *Software: Practice and Experience*, Vol. 41, No. 1, pp.23–50.

Dai, Y-S., Yang, B., Dongarra, J. and Zhang, G. (2009) 'Cloud service reliability: modeling and analysis', *Proceedings of the 15th IEEE Pacific Rim International Symposium on Dependable Computing*, 16–18 November, Shanghai, China, pp.1–17.

Do, T., Gunawi, H.S., Do, T., Harter, T., Liu, Y., Gunawi, H.S., Arpaci-Dusseau, A.C. and Arpaci-Dusseau, R.H. (2013) 'The case for limping-hardware tolerant clouds', *Proceedings of the 5th USENIX Workshop on Hot Topics in Cloud Computing*, 25–26 June, San Jose, CA, pp.1–6.

Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A. and Stoica, I. (2009) *Above the Clouds: A Berkeley View of Cloud Computing*, Technical Report No. UCB/EECS-2009-28, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 13pp.

Garg, S.K. and Buyya, R. (2011) 'Networkcloudsim: modelling parallel applications in cloud simulations', *Proceedings of the 4h IEEE International Conference on Utility and Cloud Computing*, 5–8 December, NSW, Victoria, pp.105–113.

Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J. and Ghalsasi, A. (2011) 'Cloud computing: the business perspective', *Decision Support Systems*, Vol. 51, No.1, pp.176–189.

Rausand, M. and Høyland, A. (2004) *System Reliability Theory: Models, Statistical Methods, and Applications*, John Wiley & Sons, New York.

Schwarzkopf, M., Murray, D.G. and Hand, S. (2012) 'The seven deadly sins of cloud computing research', *Proceedings of the 4th USENIX Workshop on Hot Topics in Cloud Computing*, 12–13 June, Boston, MA, pp.1–5.

Undheim, A., Chilwan, A. and Heegaard, P. (2011) 'Differentiated availability in cloud computing SLAs', *Proceedings of the 12th IEEE/ACM International Conference on Grid Computing*, 21–23 September, Lyon, France, pp.129–136.